ÉCOLE POLYTECHNIQUE

Thèse présentée pour obtenir le grade de

DOCTEUR DE L'ÉCOLE POLYTECHNIQUE

spécialité : Mathématiques Appliquées

par

Leila KALLEL

Convergence des algorithmes génétiques : aspects spatiaux et temporels

Soutenue le 5 février 1999 devant le Jury composé de

MM	Kenneth De Jong	Rapporteur
	Alain Trouvé	Rapporteur
	Thomas Bäck	Examinateur
	Josselin Garnier	Examinateur
	Pierre Liardet	Examinateur
	Colin Reeves	Examinateur
	Laurent Younes	Examinateur
	Marc Schoenauer	Directeur de thèse

Remerciements

C'est un grand plaisir pour moi de pouvoir remercier ici mon directeur de thèse Marc Schoenauer. Je lui suis particulièrement reconnaissante pour la patience et l'attention qu'il m'a toujours généreusement accordées. Je passe les détails, oh combien importants, de ses encouragements et de son soutien (je me rappelle encore de ma première conférence!), enfin je voudrais aussi, à ce propos, remercier Michele Sebag qui illumine par sa présence et ses remarques pertinentes nos groupes de travail.

Je remercie vivement Alain Trouvé pour avoir accépté de rapporter sur ma thèse, j'ai beaucoup apprécié sa lecture attentive et ses remarques qui ont permis d'améliorer certaines preuves et d'apporter un nouveau regard critique sur l'ensemble. Merci aussi à Ken De Jong qui est pour beaucoup dans l'amélioration de la version finale de ce manuscrit. C'est un grand honneur pour moi qu'ils aient pris le temps de rapporter sur ma thèse et qu'ils se soient déplacés ici pour ma soutenance.
C'est aussi un grand honneur pour moi que Thomas Bäck, Pierre Liardet, Colin Reeves et Laurent Younes participent à ce jury.

Par ailleurs, je tiens tout particulièrement à remercier Josselin Garnier, pour sa contribution précieuse à ce travail, alliant conseils, énergie et bonne humeur. Je voudrais remercier tout aussi chaleureusement, Bart Naudts, avec qui c'était bien agréable de discuter et de travailler. Josselin et Bart, je vous suis bien reconnaissante d'avoir ainsi animé le cours de ma thèse.

L'environnement particulièrement agréable dans lequel j'ai pu travailler pendant ces quelques années, doit beaucoup à Geo et Jeanne qui étonnent par leur efficacité et disponibilité permanente, mais aussi aux soins de Jean-Claude Nédélec, fondateur de ce laboratoire, puis à Pierre-Arnaud Raviart et Vincent Giovangigli, son nouveau directeur. Ces trois directeurs successifs ont bien voulu m'accueillir au sein du CMAP et je les en remercie beaucoup.

Merci aussi aux autres membres du laboratoire pour leurs qualités personnelles et professionnelles. J'aimerais ainsi adresser un merci particulier à plusieurs personnes avec qui j'ai pu discuter par moments, et qui n'ont pas hésité à me donner de leur temps et conseils : Habib, Pedro, Aldjia, Carl, François Jouve, Jérome, Christophe, Evelyne Lutton, Laurence, Sylvie, Robert, et puis, plus récemment, Rama Cont, étonnant par l'étendue de sa culture scientifique, et enfin, tous ceux que j'oublie et qui sauront me pardonner.

Merci à tous ceux qui m'ont aidé durant ces derniers moments mouvementés : Edmond pour ses conseils et sa clairvoyance, Lise qui a si gentillement accepté de corriger (et en si peu de temps) l'anglais de ma thèse, et pour en arriver à aujourd'hui, merci à ceux que sont venus m'écouter.

J'aimerais enfin remercier ma famille, pour sa présence affective a mes cotés, et plus particulièrement, mes parents d'avoir su guider et accompagner mes pas et m'aider à chaque fois que j'en ai eu besoin, à eux deux je dédie ce travail.

Contents

Overview

This thesis presents theoretical and experimental work in stochastic optimization, specifically devoted to the class of algorithms that fall under the heading of Evolutionary Optimization.

Chapter I of the thesis is mainly dedicated to the comparison of two bitstring mutation operators.
To make the comparison amenable to theoretical analysis, we chose to incorporate these mutations in a (1+1) ES. Within this framework, section 2 presents results concerning the law of the first hitting time of the solution, derived by Markov chain methods. The mutation operators compared are random deterministic bit flipping (one randomly chosen bit is flipped) and uniform non-deterministic mutation (all bits are independently flipped with the same probability). The fitness functions employed in the comparison are the needle in the haystack (yielding a random walk) and the OneMax problem. For both fitness cases, deterministic bit flipping is shown to win over uniform mutation.
In section 3, the counter-intuitive result obtained in section 2 with the cited fitness functions is shown not to hold in the case of the *long k-path* problem of G. Rudolph. Both cases of fixed and variable values of k are considered.
In an effort to extend our theoretical work to algorithms with crossover, we construct in section 4 a long path problem that proves hard for population-based Evolutionary Algorithms incorporating uniform and n-point bitstring crossover.

Chapter II is devoted to statistical indicators which measure the adequation of a given evolutionary algorithm instance to some given fitness landscape.
In section 2 it is shown that the Hamming distance to optimum defines a helper function which can be used to significantly rescale, interpret and explore the convergence behavior of an EA.
In section 3 various recognized statistical indicators (FDG, epistasis, SWO) taken from the literature are employed to identify function classes inducing similar EC behavior.
Section 4 investigates the various types of EC behavior that one can elicit with fitness functions belonging to the same FDC class.
In section 5 two new bitstring initialization procedures are described, and it is shown how the FDC can be used to choose the most effective.
Section 6 compares some known measures of crossover operator efficiency, and introduces a new method called "offspring ranking" which however proves to be a valid predictor only in the early stages of a run.

Chapter III extends the studies of the previous chapters to variable length representations, which in particular requires defining a genotypic distance for these representations. A difficult inverse problem in structural mechanics serves as an experimental testbed.
Section 2 presents the inclusion identification problem : Material defects inside a 2-D elastic structure are to be identified from the measures of edge displacement under load. No known deterministic numerical method is available to find the location of such defects.
Section 3 describes the use of fixed and variable length representations in the above structural mechanics problem. The obvious fixed-length representation, namely a bitstring mapped onto a finite-element mesh, suffers from a quadratic increase of complexity when refining the mesh. Two variable length representations of geometric structure are also described, namely holes

and Voronoï, with the corresponding topological crossover and mutation operators.
Section 4 gives experimental data comparing the quality of solutions and the convergence time for fixed and variable length representations; While the problem is shown to be tractable by GA, even with bitstrings, variable length representations outperform fixed-length genotypes by at least one order of magnitude on three out of four studied test cases.
Section 5 defines genotype distances for both holes and Voronoï representations, in order to apply the estimator methodologies described in chapter II, more specifically FDC. The proposed distances on topological genotypes are qualified by investigating their interaction with the topological mutation and crossover operators.
Section 6 applies the FDC results of II.5 to tune the initialization procedure of the variable-length GA. The actual choice of representation is not immediately derivable from the FDC in this problem. However it can be made with greater confidence thanks to the offspring ranking method of II.6 .

Notations and Definitions

Binary Search Space (chapters 1 and 2)

Genic search space $E = \{0,1\}^l$	an individual is a vector of l genes
	each gene takes its values in the space of alleles $\{0,1\}$
d	Hamming distance
d_n	normalized (to l) Hamming distance
f	fitness function (to maximize)
s^*	global optimum of f
Local optimum s of f (w.r.t. d)	$\forall\ s' \in E\ /\ d(s,s') = 1,\ f(s) \geq f(s')$
Unimodal	f has 1 local optima (which is the global optimum)
Multi-modal	f has at least 2 local optima
Hamming fitness to $s \in E$	$f(x) = l - d(x,s)$

Evolution binary operators

stochastic $\frac{c}{l}$ mutation	flips all bits independently with probability $\frac{c}{l}$
deterministic *k-bit flip* mutation	flips exactly $k \in \mathbb{N}$ uniformly randomly chosen bits
one-point crossover	exchanges parent genes at a uniform random position
uniform crossover	selects either parent alleles uniformly, for each position

Variable Length Search Space (chapter 3)

Allelic search space $\mathcal{G}$	an individual is a variable length unordered list of alleles
$\mathcal{A}$	space of alleles
Δ	distance on $\mathcal{G}$
δ	distance on $\mathcal{A}$
Θ	*Hausdorff* distance
$diam_\Delta(\mathcal{G})$	diameter of $\mathcal{G}$: $\sup\{\Delta(x,y),(x,y) \in \mathcal{G}^2\}$
f	fitness function (to minimize)
Distance fitness to $s \in \mathcal{G}$	$f(x) = diam_\Delta(\mathcal{G}) - \Delta(x,s),\ \forall x \in \mathcal{G}$

Search algorithms

(1+1)-ES: Let $X(0) \in E$. At each generation $t \in \mathbb{N}$, let $X'(t)$ be the mutated string from $X(t)$. If $X'(t)$ fitness is (strictly) better than $X(t)$, then $X(t+1) = X'(t)$ otherwise $X(t+1) = X(t)$.

(1,1)-ES: Let $X(0) \in E$. At each generation $t \in \mathbb{N}$, let $X'(t)$ be the mutated string from $X(t)$, then set $X(t+1) = X'(t)$.

GA: An elitist, generational GA with ranking selection is used in most experiments.

The ranking selection scheme is such that the probability to select individual x is proportional to its fitness rank in the current population.
The elitism is artificially maintained: in case the best fitness of the parent population is not improved by the offspring, the best individual of the parent population replaces the less fit offspring.
The whole process can be described as follows:

1. **Initialization**: a population of m random individuals (points of the search space) is generated
2. **Iteration** of the following steps until the stopping criterion is satisfied, typically a number of generations (iterations) with no fitness improvement:
   (a) **Selection** of individuals from the population to be transformed
   (b) **Transformation**: all selected individuals undergo mutation and/or crossover.
   (c) **Replacement**: The m generated offspring build the next population.

Introduction

Introduction

Ce chapitre présente sommairement les algorithmes d'évolution dans le cadre de l'optimisation de fonctions.
Il donne un aperçu de l'état de l'art du domaine, et des notions essentielles à la compréhension des travaux et de leur impact. Il introduit aussi les notations et le vocabulaire utilisés par la suite.
Les abréviations anglo-saxonnes usuelles dans le domaines des algorithmes d'évolution (EA) sont employées ici pour améliorer la lisibilité.

1 Optimisation et algorithmes d'évolution

L'introduction des algorithmes d'évolution comme outil d'optimisation stochastique, n'est pas une idée récente [Rec73, Hol75, Gol89], cependant seules des puissances de calcul croissantes ont permis leur réelle émergence. Ce type d'algorithmes propose des réponses à des problèmes dont la solution était jusque-là numériquement inatteignable. En effet :

- ils n'imposent aucune contrainte de régularité (méthode d'optimisation d'ordre 0, qui nécessite seulement les valeurs de la fonction à optimiser) ;
- ils sont capables d'évoluer sur des paysages très chahutés (plusieurs optima locaux) ;
- ils peuvent trouver plusieurs solutions quasi optimales en même temps.

Les algorithmes d'évolution ont un mécanisme de base similaire : celui-ci consiste à faire évoluer une population par transformation aléatoire de certains de ses éléments. Cependant, on distingue plusieurs origines et tendances (par ordre alphabétique) :

- La programmation évolutionnaire (EP), est apparue en Californie [FOW66] dans l'espace des automates à états finis.
- Les stratégies d'évolution (ES), sont nées en Allemagne [Rec73, Sch81], pour résoudre des problèmes numériques d'optimisation dans l'espace des vecteurs de réels.
- Les algorithmes génétiques (GA) ont été développés aux Etats Unis par Holland [Hol62], pour étudier, dans le cadre binaire, les mécanismes d'adaptation de populations dans le cadre de la psychologie/biologie. Nous considérons ici les GA comme algorithmes d'optimisation (voir [DeJ92] pour une discussion sur les conséquences de ce changement de point de vue).
- La programmation génétique (GP) [Koz94], consiste à faire évoluer des structures d'arbres, qui représentent un programme informatique (ex. en LISP); le but est de générer par évolution, un programme syntaxiquement correct qui réponde aux spécifications. Initialement sous-domaine des GA, GP se démarque des *GA* par des conférences qui lui sont spécifiquement dédiées.

Les algorithmes d'évolution sont très sensibles à de nombreux paramètres qui déterminent leur comportement. Cette sensibilité est cruciale pour le temps de calcul et la précision des résultats. Les questions de performance en relation au choix des paramètres constituent ainsi un axe de recherche majeur.

1.1 ALGORITHMES D'ÉVOLUTION

Prenons l'exemple de la maximisation d'une fonction f à valeurs dans $\mathbb{R}$, appelée fonction adaptation ou *fitness*.

En pratique, la fonction fitness est définie sur l'*espace des phénotypes* Φ, éventuellement différent de l'*espace de recherche* E (espace *génotypique*), dans lequel la solution est cherchée par l'algorithme. On se donne alors une fonction φ de décodage du génotype au phénotype, ainsi, la fitness de $x \in E$ est définie par $f(\varphi(x))$. Dans le cadre binaire considéré dans les deux premiers chapitres, l'espace des phénotypes est le même que celui des génotypes, φ est donc la fonction identité.

Pratiquement, les algorithmes d'évolution commencent par choisir aléatoirement un p-uplet – *la population* – de points de E – *les individus* –, ensuite le processus suivant est itéré jusqu'à satisfaction d'un critère d'arrêt (ex. nombre maximum d'itérations):

1. Sélection d'individus (plutôt de grande fitness) à transformer (étape Darwinienne, où les meilleurs sont plus aptes à se reproduire).
2. Transformation de chaque individu sélectionné :

   par *mutation* (transformation aléatoire de E vers E), ou

   par*croisement* avec un autre individu (transformation de $E \times E$ vers E où chaque *parent* participe par une partie complémentaire à la construction de *l'enfant*).
3. Remplacement : Sélection de p individus parmi les points initiaux et ceux obtenus par la transformation précédente (étape Darwinienne, où les meilleurs sont plus aptes à survivre).

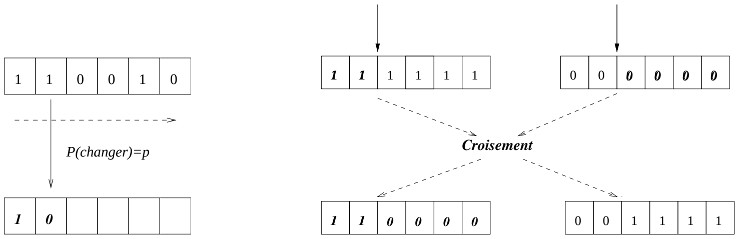
Selon le problème, un individu peut être une chaîne de bits (représentation binaire $E = \{0,1\}^l$), une chaîne de réels (ex. $E = [0,1]^l$), ou, plus généralement, une liste, ordonnée ou non, d'éléments dans un espace donné (cf. chapitre III). Le comportement des EA dépend, de manière cruciale, du choix de cette *représentation* (choix de E), des opérateurs, ainsi que du choix de la fitness.

Étant donné une fitness et une relation de voisinage (une distance d par exemple) sur l'espace de représentation E, on peut définir le *paysage de fitness* par le triplet (E, f, d).

Une représentation est dite génique quand le génotype est un vecteur de *gènes*, prenant leurs valeurs dans un espace d' allèles. c'est le cas de la représentation binaire, ou l'espace des allèles est $\{0,1\}$.

GA classique : l'algorithme génétique est dit classique s'il utilise :

- un schéma de sélection proportionnelle à la fitness (définie plus loin).
- un remplacement "*generational*" (la phase de remplacement consiste à remplacer tous les parents par leurs transformés).



(a) (b)

FIG 1 : *Opérateurs d'évolution dans le cas d'un codage binaire. (a) La p mutation : chaque bit est changé avec une probabilité p ; (b) Croisement à un point : échange de deux parties complémentaires des deux parents.*

Traditionnellement [Gol89], le croisement est prépondérant, la mutation étant de probabilité très faible.
De plus, un EA est dit **élitiste** s'il utilise une procédure de remplacement modifiée : si le meilleur enfant est moins bon (en fitness) que le meilleur des parents, celui-ci est réintroduit dans la population.

Stratégies d'évolution (ES) : On distingue deux types d'ES selon la sélection (déterministe) utilisée :

- $(\mu + \lambda) - ES$: μ parents donnent λ enfants (chaque parent donne λ/μ enfants en moyenne), les μ meilleurs parmi parents et enfants sont choisis pour former la population suivante (il est élitiste par définition).

- $(\mu, \lambda) - ES$: μ parents donnent λ enfants, les μ meilleurs enfants sont choisis pour former la population suivante (la meilleur fitness de la population peut décroître).

Pratiquement, l'opérateur principal des ES est la mutation. Le croisement est utilisé avec pragmatisme.

Opérateurs d'évolution : Les opérateurs de mutation et de croisement (Fig. 1) agissent sur le codage des données. Prenons l'exemple d'un codage binaire. Les mutations les plus utilisées sont :

- 1-*bit mutation* : choisir une position uniformément dans $\{1..l\}$, et changer la valeur du bit correspondant.

- $\frac{c}{l}$ *mutation* : changer chaque bit indépendamment avec une probabilité $\frac{c}{l}, c > 0$.

En codage réel [ES93, Mic96], la mutation consiste à rajouter une perturbation Gaussienne centrée d'écart type à définir. Nous n'entreront pas dans les détails de la mutation adaptative [BS93].
Le croisement(en binaire) met en jeu deux chaînes de bits. Une fois les deux parents choisis (des exemples de sélections sont donnés par la suite), le croisement consiste par exemple à (cf. Fig. 1) :

- *Croisement à un point* : choisir une position de coupure uniformément dans $\{1 \dots l\}$, et construire un enfant à partir de deux parties complémentaires des deux parents.

- *Croisement uniforme* : choisir la valeur de chaque gène au hasard, à partir de l'un où l'autre des parents.

Procédure de sélection : La sélection joue un rôle clef dans l'algorithme, puisqu'elle guide l'évolution vers les zones de meilleure fitness. Sans sélection, un algorithme d'évolution ne ferait guère mieux qu'une recherche aléatoire. Une comparaison de plusieurs procédures de sélection est conduite dans [CDC96]. Les sélections les plus populaires sont :

- *Sélection proportionnelle* : la probabilité de sélectionner l'élément i de la population $X = \{x_1, \dots x_m\}$, $x_i \in E$ est $f(x_i)/(\sum_{k\in\{1\dots m\}} f(x_k))$.
- *Sélection par le rang linéaire* : la même méthode, avec le rang au lieu de la valeur de la fitness dans la population: Si $r(x_i)$ est le rang de la fitness de x_i dans la population alors, la probabilité de sélectionner x_i est $(m - r(x_i))/(m(m+1)/2)$.
- *Sélection tournoi p* : p individus sont choisis au hasard et le meilleur est sélectionné. Cette sélection donne les mêmes espérances (du nombre d'enfants d'un individu donné) que la sélection par le rang, mais permet une plus grande variance [DS95].

La sélection proportionnelle, dans le cadre pratique de populations de petite taille (généralement quelques centaines d'individus) par rapport à la taille de l'espace, peut donner lieu à des convergences prématurées. En effet, le meilleur individu risque d'envahir rapidement la population, dés qu'il est loin devant les autres en fitness. Les sélections par le rang ou le tournoi, suppriment cette dépendance par rapport aux valeurs de la fonction fitness. Voir [GD91, DS95] pour l'étude et comparaison de différents types de sélections.

1.2 THEORIE DES SCHÉMAS *(Holland 1975)*

La théorie des schémas est l'une des premières analyses heuristiques du comportement des algorithmes génétiques, dans le cadre d'un espace binaire $E = \{0,1\}^l$, avec une sélection proportionnelle, un croisement à un-point et une $1/l$ mutation.
Un schémas est une séquence $\mathcal{H} \in \{0,1,*\}^l$, où le symbole $*$ désigne indifféremment 0 ou 1. Par exemple, $(1*11) = \{1011, 1111\}$. Un schémas définit donc un sous-espace de E.
Si on note $N(\mathcal{H}, t+1)$ le nombre de représentants de $\mathcal{H}$ dans la population à l'instant t (X_t), alors, si on néglige les éventuelles créations de nouveaux représentants de $\mathcal{H}$ provenant (de la destruction par mutation ou croisement) des autres schémas dans la population :

$$\mathbf{E[N(\mathcal{H}, t+1)]} \geq \mathbf{N(\mathcal{H}, t)}\,\frac{\mathbf{\bar{f}(\mathcal{H}, t)}}{\mathbf{\bar{f}(X_t)}}\left[\mathbf{1 - p_m o(\mathcal{H}) - p_c \frac{l(\mathcal{H})}{l-1}})\right]$$

où $o(\mathcal{H})$ (ordre de $\mathcal{H}$) est le nombre de bits fixés dans $\mathcal{H}$ et $l(\mathcal{H})$ (longueur de $\mathcal{H}$) est la distance entre la première et la dernière position des bits fixés (dans l'exemple précédent, $o(\mathcal{H}) = 3$ et $l(\mathcal{H}) = 4$). $\bar{f}(\mathcal{H}, t)$ est la fitness moyenne des représentants de $\mathcal{H}$ présents dans la population X_t.
On a donc une progression de type géométrique si $c_t = \frac{\bar{f}(\mathcal{H},t)}{\bar{f}(X_t)} \geq \alpha > 1$. Une conséquence directe est que les schémas de fitness au dessus de la moyenne de la population à l'instant t ont tendance à envahir les futures populations.

Interprétations : Cela a donné lieu à des généralisations qui vont jusqu'à interpréter la convergence des algorithmes génétiques en termes de partitions de schémas : *'les schémas courts et performants (building blocks) croissent et se multiplient.'*
Une fonction est dite trompeuse [Gol89] (*deceptive*) si les schémas les plus performants ne contiennent pas l'optimum. La notion de '*deception*' est alors proposée comme source substantielle de difficulté d'un paysage de fitness, puisqu'elle défavorise l'émergence de l'optimum.

Critiques :

1- La théorie ne tient pas compte de la variance des valeurs de la fitness dans un schéma. Ainsi, la fitness observée à l'instant t dans la population X_t est différente de la fitness moyenne du schéma [Rad91a, Rad91b, RS95a]. D'autant plus qu'en pratique, les populations sont de petite taille (relativement à la taille de E).

2- La fitness moyenne de la population croît au cours du temps. Donc le qualificatif 'schéma performant' dépend du temps. Pour les mêmes raisons, l'hypothèse $c_t >\geq \alpha > 1$ n'est pas réaliste.

3- il existe des exemples de fonctions trompeuses (*deceptive*) faciles à optimiser pour les algorithmes génétiques [Wil91, Whi91]. Par ailleurs, seules les fonctions à variables séparables sont totalement (pour tout ordre de schéma) non trompeuses (*deceptive*) [DW91].

2 Principaux résultats théoriques de convergence

Il existe plusieurs approches pour l'étude de la dynamique des algorithmes génétiques :

1- L'approche physique, qui pour un problème donné, approxime la distribution des fitness dans une population par les '*cumulants*' (moyennes du premier, second et troisième ordre) [PBS97, Bor96, SPB96].
Cette approche permet une meilleure compréhension de la dynamique, et de l'influence des opérateurs sur plusieurs cas de fonctions fitness. Cependant, à ce jour, il n'y a pas résultats de convergence globaux (valables pour toute fonction fitness) par cette approche.

2- L'approche probabiliste, qui utilise une modélisation (exacte) par les chaînes de Markov donne des résultats de convergence globaux :

On se place dans l'espace binaire ($E = \{0,1\}^l$), on considère une mutation irréductible [1] (exemple la $1/l$ ou la 1-bit mutation) et une sélection proportionnelle ou par le rang.
La population à l'instant t est notée $X_t = (x_1, \dots x_m)$, $x_i \in E$. L'espace d'état de la chaîne X_t est donc E^m, et la matrice de transition donne les probabilités de passer d'une population à une autre suivant le schémas :

$$\mathbf{X_t} \overset{Mutation}{\longrightarrow} Y_t \overset{Croisement}{\longrightarrow} Z_t \overset{Selection}{\longrightarrow} \mathbf{X_{t+1}}$$

La chaîne X_t est homogène [2], irréductible [3], sur l'espace fini $\{0,1\}^{lm}$. Elle est donc ergodique. Cela implique qu'il existe un processus limite, ie. que le comportement de la chaîne converge asymptotiquement vers une loi de probabilité indépendante de la population de départ:

$$\lim_{t\to\infty} P[X_t = y/X_0 = x] = g(y),\ avec\, g(y) > 0\ \forall y \in E^m$$

[1] la probabilité de passer d'un état i à j (de E) en un nombre fini (t_i) de mutations est $P^{(t_i)}(i,j) > 0$

[2] La population à l'instant $t+1$ dépend uniquement de la population à l'instant t. Ainsi les probabilités de transition sont indépendantes du temps et des populations antérieures.

[3] On peut trouver une composition finie d'opérations qui permet d'aller de n'importe quel point de l'espace d'état E^m à n'importe quel autre.

Cependant, ce résultat de 'convergence' est insuffisant, vu que l'on s'intéresse plus particulièrement à la convergence vers les optima de f, la fonction à optimiser.

2.1 CONVERGENCE EN PROBABILITÉ DES ES

Rudolph [Rud97b] prouve que dans le cadre de la minimisation d'une fonction f ayant un minimum unique noté f^*, si on note $f(X_t) = min\{f(x), x \in X_t\}$ la meilleure fitness de la population X_t, et $V_t = min(V_{t-1}, f(X_t))$ la meilleure fitness trouvée depuis le début jusqu'à l'instant t. Alors,

1. La séquence $f(X_t)$ d'un processus $(1+1)-ES$ avec une mutation telle que : *la probabilité de passer d'un état i quelconque et en un nombre fini de transitions (t_i), à un élément de X^* est non nulle*, converge en probabilité (eq. 1) et en moyenne (eq. 2) vers f^*.

   De plus cette condition sur la mutation est nécessaire à une convergence globale (indépendante de l'état initial).

2. La séquence V_t des meilleures valeurs de fitness trouvées – depuis le début – par un processus $(1, \lambda) - ES$, $\lambda \geq 1$ avec une mutation positive [4], converge en probabilité et en moyenne vers f^*.

$$\lim_{t \to \infty} P[V_t - f^* = 0] = 1, \forall X_0 \in E^m \tag{1}$$

$$\lim_{t \to \infty} E[V_t - f^* = 0] = 0, \forall X_0 \in E^m \tag{2}$$

3. la chaîne $f(X_t)$ d'un processus $(1, \lambda) - ES$ avec une mutation positive [6] (par exemple la *$1/l$ mutation)*, ne converge pas en probabilité vers f^* indépendamment de l'initialisation, à moins que f ne soit constante.

Ces résultats sont valables avec ou sans opérateur de croisement (avec probabilité de croisement $p_c < 1$). Seule la mutation est nécessaire à la convergence.
Par ailleurs, les résultats ci-dessus sont valables si on remplace la condition 'mutation positive' par : 'la matrice de transition du processus est primitive' (irréductible [5] et ayant une puissance positive [6]). Cette condition est réalisée avec une mutation positive par exemple.

2.2 CONVERGENCE EN PROBABILITÉ DES ALGORITHMES GÉNÉTIQUES

• Rudolph [Rud94a] montre la convergence de l'algorithme génétique classique. Ceci dans le cadre d'une mutation positive et sous réserve que le meilleur soit mémorisé [7] d'une génération à une autre (on considère la chaîne V(t) des meilleures valeurs de fitness trouvées depuis le début).

• Agapie [Aga97] prouve que si la matrice de mutation (M) est irréductible, à diagonale positive et si les matrices de sélection (S) et de croisement (C) sont à diagonale positive, alors

[4] La probabilité de passer par mutation d'un état i à j (de E) est $m_{ij} > 0$
[5] la probabilité de passer d'un état à un autre en un nombre fini de transitions est strictement positive
[6] terme à terme strictement positive
[7] ce n'est pas de l'élitisme : le meilleur n'est pas réintroduit dans la population

la matrice de transition du processus total (P=CMS) est irréductible et à diagonale positive, et donc il y a convergence en probabilité de la chaîne V(t) (des meilleures valeurs de fitness trouvées depuis le début) vers les optima de f.
Ce résultat généralise celui de Rudolph ci-haut.

• Vose étudie l'évolution de populations de taille infinie [VL91, VW95], démontrant la convergence vers une population unique. Cependant le modèle part d'une population initiale contenant une fraction (infinie) de l'optimum, et n'est pas transposable au cas des populations de taille finie.

• Cerf [Cer94, Cer96] démontre (dans le cadre in homogène) que si les opérateurs d'évolution et de sélection évoluent d'une manière donnée en fonction du temps, et si la taille de la population dépasse une taille critique dépendant des paramètres de l'algorithme et de la fonction fitness f, alors il y a convergence vers les optima globaux de f en un temps fini.
Ce résultat ne prouve pas seulement la convergence en probabilité mais, prouve en plus que l'algorithme à partir d'un moment donné, converge sur les maxima de f, et y reste avec une probabilité 1.

$$\forall x \in E \; P[\exists N \; \forall \mathbf{n} \geq \mathbf{N} \; [X_{T_n}] \in f^* / X_0 = x] = 1$$

• C. Meise démontre la convergence dans le cadre du recuit simulé parallélisé [Mei]. Les mêmes techniques s'appliquent aux algorithmes génétiques.

Il existe aussi d'autres résultats de convergence faisant intervenir la modélisation par chaînes de Markov, mais avec une approche plus algébrique et topologique [Mor, Lez].

2.3 VITESSE DE CONVERGENCE

Si les aspects limites du comportement des algorithmes d'évolution sont accessibles par des modèles qualitatifs simples via les meilleures fitness à chaque génération, les études théoriques du temps de convergence (premier temps d'atteinte de l'optimum), sont plus difficiles à mettre en oeuvre. Elles nécessitent l'étude du modèle Markovien exact de l'algorithme concerné.
Seulement les termes de la matrice de transition deviennent plus compliqués dès que la taille de la population dépasse 1. Ainsi, la majorité des résultats sont obtenus dans le cas $\mu = 1$.

• La fonction *OneMax* (Annexe A) introduite par Ackley [Ack87] est définie par $f(x) = l - d(x, 11\ldots1)$. Elle compte le nombre de uns dans une chaîne de bits de longueur l. Cette fonction a été largement étudiées [Bäc92, M9̈2, Bäc93, Rud97b]. Le temps de convergence d'un processus (1+1)-ES, avec une 1-bit mutation, est majoré par $l \ln(l)$. Récemment, Droste [DJW98] a prouvé que cette borne était atteinte pour les fonctions linéaires.

• Les *longpaths* (en binaire) introduits par Horn [HGD94] sont des chemins de longueur exponentielle. Un processus (1+1)-ES converge en un temps exponentiel ($O(2^{l/2})$) avec la 1-bit mutation, et converge en un temps polynômial ($O(l^2)$) avec une $1/l$ mutation [Rud96].

• Les fonctions convexes dans le cadre des ES (codage réel, ex. $f(x) = |x|^2$) ont été largement étudiées [Sch81, BS93, Rud97b, Bey93, Bey95a, Bey95b] notamment dans le cas $\mu > 1$ [Bey94]. Les ES sont capables de converger en un temps log-linéaire [8] sur des fonctions localement

[8]Soit f la fonction fitness à minimiser. Si $\hat{f}_t$ est la plus petite fitness trouvée à l'instant t, alors $\log \hat{f}_t / \hat{f}_0$ se comporte asymptotiquement comme t.

convexes.

2.4 LIMITES PRATIQUES DES RÉSULTATS DE CONVERGENCE

Les résultats globaux de convergence en probabilité, de par leur généralité (f est quelconque), perdent en efficacité. En effet, pour important qu'il est, le résultat de convergence vers les optima de la fonction à optimiser, ne nous donne aucun renseignement sur l'influence relative des opérateurs et des paramètres, ni sur l'ordre du temps de convergence.
Par exemple, le temps de convergence peut être exponentiel (en la taille du problème l) auquel-cas, il dépasse très vite les limites du raisonnable, se chiffrant en siècles.
On peut aisément imaginer qu'un algorithme d'évolution ne fera pas mieux qu'une recherche aléatoire (d'où un temps exponentiel) sur un paysage presque plat, par exemple une fonction constante partout sauf en un point où elle est maximale. Optimiser une telle fonction, revient à chercher une aiguille dans une botte de foin, et l'on peut s'attendre à ce que le temps de convergence soit au moins exponentiel (cf. section I.2.3).
Le cas des paysages plats est toutefois purement illustratif. Un cas plus réaliste est celui des longpaths [HGD94]. Ces fonctions définissent des chemins de longueur exponentielle, tel que le processus (1+1)-ES avec 1-bit mutation, converge en suivant une trajectoire prédéfinie, en un temps exponentiel en la taille du problème. Cependant, avec la 1/l mutation ce temps devient polynômial [Rud96].

Ainsi, deux questions sont posées : existe t-il des fonctions (non constantes) optimisables en un temps exponentiel avec la 1/l mutation ? Existe t-il des chemins contrôlés et prédéfinis pour les algorithme génétiques? (c'est l'objet des sections I.3 et I.4).

Cependant, au-delà de cette question d'étalonnage, si l'on sait que les algorithmes évolutionnaires convergent toujours, il est important d'identifier les classes de fonctions pour lesquelles ils convergent suffisamment vite. Un axe de recherche majeur est ainsi dédié à analyser les comportements des algorithmes d'évolution pour identifier les facteurs qui diminuent ou augmentent leurs performance.

Cependant, l'algorithme ne 'voit' pas le paysage de fitness d'une manière unique : chaque changement des opérateurs d'évolution (croisement et mutation), pire, chaque changement de représentation (codage des données) change complètement la nature du paysage à optimiser. Ainsi, les questions de performances se posent non seulement en relation au choix de la fonction fitness (cf. sections II.3 et II.4), mais aussi au choix des opérateurs et de la représentation (cf. sections II.6 et III.6.3).

Principaux Résultats

Principaux Résultats

Ce chapitre propose un résumé de la thèse. Les propositions et figures sont numérotées comme dans les chapitres respectifs (chapitre.section.numéro).

1 Convergence des algorithmes d'évolution

Le premier chapitre est dédié à l'étude (et étalonnage) des types de convergences des algorithmes d'évolution d'un point de vue temporel (premier temps d'atteinte de la solution) et spatial (trajectoire).

La première partie propose une comparaison de la trajectoire et du temps de convergence, de deux opérateurs de mutation, d'une part avec avec le processus (1,1)-ES, et d'autre part le processus (1+1)-ES sur le problème OneMax [9] (cf. Annexe A) . Les lois asymptotiques du premier temps d'atteinte de la solution sont calculées, indiquant la nette supériorité, de la 1-bit mutation sur la $\frac{c}{l}$ mutation, pourtant plus utilisée en pratique. Les trajectoires des deux processus (avec chaque mutation) sont comparées.

Poussant plus loin la comparaison entre les deux mutations, la deuxième partie démontre l'existence de 'chemins exponentiels' pour la $\frac{c}{l}$ mutation (à l'image des *longpaths* [HGD94] pour la 1-bit mutation):
L'étude de la la distribution statistique du temps de convergence des stratégies d'évolution sur les *long k-path* de Rudolph [Rud96], permet d'identifier une sous classe de ces fonctions telle qu'un processus $(\mu + \lambda)$-ES muni de la $\frac{c}{l}$ mutation converge sans prendre de raccourcis, et donc en un temps exponentiel.

Ceci prouve qu'il existe des chemins exponentiels et contrôlés pour la $\frac{c}{l}$ mutation. Cependant, la question reste ouverte pour les opérateurs de croisement (un-point et complèmentaire), qui prennent des raccourcis conséquents sur ces mêmes *long k-path*.

La dernière partie propose une méthode générique pour construire des 'chemins contrôlés' pour l'opérateur de croisement à partir d'une suite de points aléatoires de l'espace. Ainsi, un algorithme muni des opérateurs de mutation et de croisement, converge en 'passant' dans l'ordre, par les points du chemin prédéfini. Ce résultat, empirique pour le croisement, est aussi démontré dans le cas de la $\frac{c}{l}$ mutation.

[9] La fitness est simplement le nombre de 1

1.1 COMPARAISON DE DEUX MUTATIONS BINAIRES

But : On se donne les points X^0 et X^f de $E = \{0,1\}^l$ tels que $d(X^0, X^f) = k_f$, et on cherche à déterminer les temps d'atteinte respectifs $\bar{T}$ et T du point X^f par le processus (1,1)-ES respectivement avec la 1-bit mutation (chaîne $\bar{X}(n)$) et la 1/l mutation (chaîne $X(n)$).

De toute façon, la recherche à l'aide des chaînes de Markov est une très mauvaise stratégie, en comparaison à une recherche aléatoire ou exhaustive (comme le confirme la proposition I.2.2 ci-dessous).
Cependant, l'objectif de cette étude est d'une part, l'étalonnage du temps de convergence, d'autre part, la comparaison des propriétés d'exploration (trajectoires) des deux mutations en l'absence de pression sélective (donc sans interaction avec le paysage de fitness).

Le temps de convergence est donné par
Proposition I.2.2
Si $d(X^0, X^f) = 1, ..., l$, alors on a :

$$\mathbb{E}[T] \simeq 2^l \frac{e}{e-1}, \qquad \mathbb{E}[\bar{T}] \simeq 2^l,$$

dans le sens où $2^{-l} \times \mathbb{E}[T]$ (resp. $2^{-l} \times \mathbb{E}[\bar{T}]$) tend vers $e/(e-1) \simeq 1.58$ (resp. 1) quand $l \to \infty$.

$\longrightarrow$ Ainsi, la 1-bit mutation est, en moyenne, $\frac{e}{e-1}$ fois plus plus rapide que la $1/l$ mutation avec le processus (1,1)-ES. La proposition suivante donne les lois des premiers temps d'atteinte, et permet de conclure que la 1-bit mutation est aussi plus performante à court terme ($\forall t \in \mathbb{N}$, $\mathbb{P}[T < t] \leq \mathbb{P}[\bar{T} < t]$).

La démonstration de la proposition I.2.2 ci-dessus, repose sur une représentation des lois des processus X et $\bar{X}$ en fonction de la chaîne de Markov auxiliaire $Z(n) = d(\bar{X}(n), X^f)$ (respectivement $d(X(n), X^f)$) à espace d'états $\{0, ..., l\}$. Dans les deux cas, les approches sont similaires. Ainsi, on présente uniquement les arguments utilisés dans le cas de $\bar{X}$:

Le premier instant d'atteinte du point 0 par $Z(n)$ suit la même loi que le premier instant d'atteinte du point X^f par $\bar{X}(n)$. On cherche donc à déterminer $\mathbb{E}_k[\nu_0]$, le premier temps d'atteinte de 0 par Z(n) partant de $Z(0) = k$. Les propriétés de Markov faible et forte permettent de démontrer la formule de récurrence (pour $k > 1$) :

$$\mathbb{E}_{k+k'}[\nu_k] = \mathbb{E}_{k+k'}[\nu_{k+k'-1}] + \mathbb{E}_{k+k'-1}[\nu_k].$$

En itérant cet argument, on obtient pour tout $k' > k$:

$$\mathbb{E}_{k'}[\nu_k] = \sum_{j=k}^{k'-1} \mathbb{E}_{j+1}[\nu_j]$$

D'où l'encadrement de $\mathbb{E}_{k_f}[\nu_0]$: $\mathbb{E}_1[\nu_0] \leq \mathbb{E}_{k_f}[\nu_0] \leq \mathbb{E}_1[\nu_0] + \mathbb{E}_{k_f}[\nu_1]$

Le terme $\mathbb{E}_1[\nu_0]$ s'obtient (par Markov faible) en fonction du premier temps de retour $\mathbb{E}_0[\bar{\nu}_0](= 1 + \mathbb{E}_1[\nu_0])$. Or par la théorie générale des chaînes de Markov, $\mathbb{E}_0[\bar{\nu}_0] = \frac{1}{\mu(0)}$, où $\mu(k) = 2^{-l} C_k^l$ est la mesure de probabilité invariante de la chaîne $Z(n)$.
Il ne reste plus qu'à majorer le terme $\mathbb{E}_{k_f}[\nu_1]$. En utilisant l'expression de $\mathbb{E}_{k'}[\nu_k]$, on a:

$$\mathbb{E}_{k_f}[\nu_1] \leq \sup_{k \leq [l/2]} \mathbb{E}_k[\nu_1] + \sup_{k \geq [l/2]} \mathbb{E}_k[\nu_{[l/2]}]$$

Après majoration des deux termes de droite de cette inégalité, on obtient le résultat de la proposition.

Les lois asymptotiques de T et $\bar{T}$ sont données par:

Proposition I.2.4
Si $d(X^0, X^f) \neq 0$, alors $2^{-l}(1-e^{-1})T$ (resp. $2^{-l}\bar{T}$) converge en loi vers une variable aléatoire de loi exponentielle et de moyenne 1 quand $l \to \infty$.

$\longrightarrow$ Conséquence directe : si on considère N processus (1+1)-ES qui évoluent indépendamment, alors le minimum des premiers temps d'atteinte de ces processus T_N (resp. $\bar{T}_N$), est tel que $2^{-l}T_N$ (resp. $2^{-l}\bar{T}_N$) converge en loi vers une variable aléatoire de loi exponentielle et de moyenne $N^{-1}(1-e^{-1})^{-1}$ (resp. N^{-1}) quand $l \to \infty$.
Ainsi, le gain en temps de convergence est de l'ordre de N. Mais, en terme de nombre total d'opérations de mutation effectuées, il n'y a pas d'avantage à augmenter N.

Une esquisse de la preuve de la proposition I.2.4 est donnée ci-dessous :
D'après la proposition I.2.2 ci-dessus, le premier temps d'atteinte ν_0 de $\bar{X}$ vérifie : $lim_{l\to\infty}\mathbb{E}[\nu_0] = 2^l$, on a donc :

$$\sup_{l \in \mathbb{N}} \mathbb{E}[2^{-l}\nu_0] < \infty$$

ce qui suffit d'après [EK86, théorème II-2-2] à établir la relative compacité de la suite de variables $2^{-l}\nu_0$. Ceci implique qu'il existe au moins une valeur d'adhérence de la suite $2^{-l}\nu_0(l)$.
On montre que cette valeur d'adhérence ν vérifie pour tous $a, b > 0$:

$$\mathbb{P}(\nu \geq b) = \mathbb{P}(\nu \geq a + b/\nu \geq a)$$

D'après le lemme B.3, ν est forcément une variable exponentielle. Comme sa moyenne est 1, on prouve l'unicité de cette limite faible, et par conséquent la convergence en loi de $2^{-l}\nu_0$ vers cette variable exponentielle.

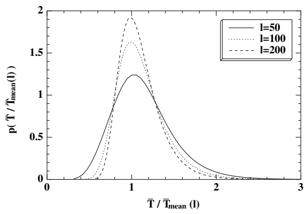
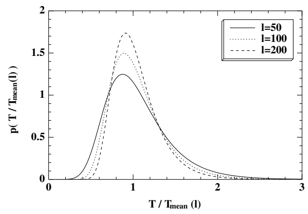
La comparaison (heuristique) des trajectoires des deux processus montre que les deux chaînes coïncident aux indices S_n (cf. Fig. 6) :

Proposition I.2.12
Quand $l \to \infty$ on a:

$$(X(n))_{n\in\mathbb{N}} \overset{loi}{\simeq} (\bar{X}(S_n))_{n\in\mathbb{N}}, \ \ avec \ \lim_{l\to\infty} \frac{S_n}{n} = 1,$$

$S_n = \sum_{j=1}^{n} N_j$, où N_j représente le nombre de bits mutés en une transition du processus X quand $l \to \infty$: $\mathbb{P}(N_j = k) = \frac{e^{-1}}{k!}$

$\longrightarrow$ Comme le montre la figure 6, $X(n)$ ($1/l$ mutation) saute certaines valeurs prises par la chaîne $\bar{X}(n)$. Ainsi, quand la chaîne $\bar{X}(n)$ atteint X^f pour la première fois, alors X le manque avec une probabilité de e^{-1} (pour $l \gg 1$).

<table>
<tr><th>Choix de x_0</th><th>1-bit mutation</th><th>$\frac{c}{l}$ mutation, $c > 0$</th></tr>
<tr><td colspan="3" align="center">Mutation sans sélection: (1,1)-ES</td></tr>
<tr><td rowspan="3">$\mathrm{d}(x_0, x^*) = k$
$\forall k \in \{0 \ldots l\}$</td><td>$\mathbb{E}[\bar{T}] = 2^l$</td><td>$\mathbb{E}[T] = 2^l (1 - e^{-c})^{-1}$</td></tr>
<tr><td>$N\ 2^{-l}\ \bar{T}_N \ \xrightarrow{d} Z$</td><td>$N(1 - e^{-c})\ \ 2^{-l}\ T_N \ \xrightarrow{d} Z$</td></tr>
<tr><td colspan="2">Ecart type $\simeq$ moyenne
$E_k[\nu_1] = O(2^l/l)$ et $E_1[\nu_0] \simeq 2^l$</td></tr>
<tr><td colspan="3" align="center">Mutation et sélection : (1+1)-ES, OneMax</td></tr>
<tr><td>$\mathrm{d}(x_0, x^*) = k$</td><td>$\mathbb{E}[\bar{T}] = l\ \ln k$</td><td>$\mathbb{E}[T] = l\ \frac{e^c}{c}\ \ln k$</td></tr>
<tr><td rowspan="4">uniforme dans E</td><td>$\mathbb{E}[\bar{T}] = l\ \ln l$</td><td>$\mathbb{E}[T] = l\ \frac{e^c}{c}\ \ln l$</td></tr>
<tr><td>$\dfrac{\bar{T}_N - l \ln l}{l} \xrightarrow{p} -\ln 2 - \ln Z_N$</td><td>$\dfrac{T_N - \frac{e^c}{c} l \ln l}{l} \xrightarrow{p} -\dfrac{e^c}{c}(\ln Z_N + C) + R(c)$</td></tr>
<tr><td></td><td></td></tr>
<tr><td colspan="2">Variances $\to 0$ à la vitesse $1/ln(l)$</td></tr>
</table>

où Z est une variable aléatoire exponentielle de paramètre 1, et $E_k[\nu_j]$ est le premier temps d'atteinte du point j par la chaîne $Z(n) = d(X(n), X^f)$ partant de $Z(0) = k$.
T_N est le minimum des temps d'atteinte de N processus indépendants. Z_n une variable aléatoire de densité $p_N(z) = N(1 - e^{-z})^{N-1}\, e^{-z}\, \mathbb{1}_{z \geq 0}$, $C \simeq 1.58$ est la constante d'Euler, et $R(c)$ une fonction réelle de c.

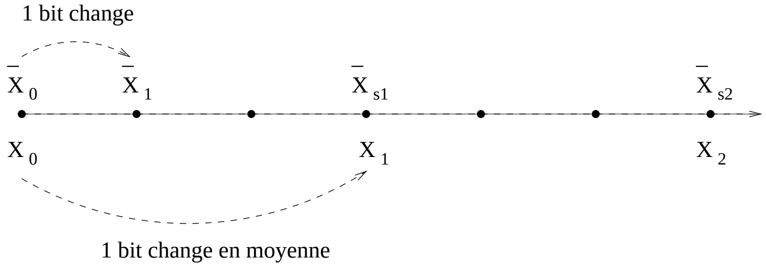


FIG 6 Les deux chaînes coïncident aux indices S_n : quand la chaîne $\bar{X}$ atteint X^f pour la première fois, alors X le manque avec une probabilité e^{-1}

Enfin, le tableau (1.1) récapitule l'essentiel des résultats obtenus, à la fois dans le cas (1,1)-ES présenté ci-dessus, et dans le cas du processus (1+1)-ES sur le problème OneMax. De plus, ces résultats impliquent entre autres que :

1. Dans le cas du processus (1,1)-ES, il n'y a aucun avantage (en nombre d'évaluations) à augmenter la taille de la population, ou à considérer $N = O(l)$ processus indépendants.
Par ailleurs, le temps nécessaire à la convergence partant d'un voisin de la solution ($\simeq 2^l$), est largement supérieur à celui nécessaire pour atteindre un voisin de la solution à partir d'un point initial quelconque ($O(2^l/l)$).

2. Dans le cas du problème OneMax, $O(l)$ processus indépendants permettent de réduire le temps de convergence de $l \ln \ln l$, ce qui ne change pas l'ordre du temps de convergence (en nombre de générations) asymptotiquement, et augmente le nombre d'évaluations. La seule manière d'avoir un gain significatif est de prendre un nombre exponentiel de processus.
On peut aussi noter que la distribution du temps de convergence est asymétrique, ainsi, en pratique, le temps de convergence est plutôt proche ou supérieur à la moyenne.

1.2 TEMPS DE CONVERGENCE SUR LES *LONG k-PATHS*

But : Étude de la distribution statistique du temps de convergence des processus $(\lambda+\mu)$-ES sur les *long k-paths* de Rudolph [Rud96] (décrits dans l'annexe A), dans les cas : k constant (améliorant les résultats de Rudolph) et $k = l^\alpha$.

On note T le temps de convergence du processus (1,1)-ES avec la $1/l$ mutation (chaîne $X(t)$), avec $X(0) = 0.....0$, le premier point du chemin.

Le *long k-path* est un chemin de longueur exponentielle ($O(k2^{\frac{l-1}{k}})$) construit récursivement sur l à partir d'une séquence d'individus ayant des fitness croissantes. Un exemple de séquence (ou chemin) est présenté ci-dessous, pour $l = 7$ et $k = 3$ (la fitness est proportionnelle à l'indice dans le chemin) :

$$\begin{array}{lll}
0000000_1 & 0011110_7 & 1111110_9 \\
0000001_2 & 0111110_8 & 1111111_{10} \\
0000011_3 & & 1110111_{11} \\
0000111_4 & & 1110011_{12} \\
0001111_5 & & 1110001_{13} \\
0001110_6 & & 1110000_{14}
\end{array} \qquad (3)$$

La seule manière pour le processus $X(t)$ de faire un "saut" de plus de k positions sur le chemin des *long k-path* est de muter k bits à la fois. On introduit deux familles de "raccourcis" (*shortcut*) $\mathcal{S}$ et $\mathcal{S}'$:

- $\mathcal{S}$ consiste en $\frac{l-1}{k}$ raccourcis notés $\mathcal{S}_j$ pour $j = 1, ..., \frac{l-1}{k}$. Le raccourci $\mathcal{S}_j$ mute les k bits aux positions successives $[(j-1)k+1, jk+1]$.

- $\mathcal{S}'$ consiste en tous les raccourcis qui mutent au moins les k bits aux positions successives $[(j-1)k+1, jk+1]$, $j = 1, ..., (l-1)/k$.

Ainsi, $\mathcal{S}'$ contient tous les $\mathcal{S}_j$. Cependant, le temps d'attente d'un événement de $\mathcal{S}$ ($T_{\mathcal{S}}$) ou de $\mathcal{S}'$ ($T_{\mathcal{S}'}$) est asymptotiquement du même ordre (loi exponentielle de moyenne de l'ordre de l^k).

On considère alors deux événements complètementaires concernant le type de convergence, i.e. avec ou sans raccourcis :
- $\mathbf{G_1}$: le processus ne contient pas de raccourci de $\mathcal{S}'$.
- $\mathbf{G_2}$: le processus contient au moins un raccourci $\mathcal{S}'$.

• **Premier cas :** $k = l^\alpha$

Proposition I.3.2

$$\begin{aligned}
&\ln \mathbb{E}[T] = l^\alpha \ln(l), && \textit{pour } k = l^\alpha, \textit{ avec } \alpha < 1/2,\\
&\ln \mathbb{E}[T] = l^{1-\alpha} \ln(2), && \textit{pour } k = l^\alpha, \textit{ avec } \alpha \geq 1/2,\\
&\lim_{l\to\infty} \mathbb{P}(G1) = 1.
\end{aligned}$$

où $\mathbb{P}(G_1)$ *désigne la probabilité de l'événement* G_1.

$\longrightarrow$ Conséquence directe de cette proposition : les raccourcis accélèrent la convergence uniquement dans le cas $\alpha < 1/2$. Autrement ($\alpha \geq 1/2$), presque sûrement, la solution est atteinte avant qu'aucun raccourci n'ait pu avoir lieu. Cela implique que les *long* l^α*-path* ($\alpha \geq 1/2$) définissent des chemins exponentiels pour la $1/l$ mutation.

La démonstration se fait en estimant les 4 membres à droite dans l'égalité :

$$\mathbb{E}[T] = \mathbb{E}\left[T/G_1\right]\mathbb{P}[G1] + \mathbb{E}\left[T/G_2\right]\mathbb{P}[G2]$$

Sous G1 Sans raccourcis, le processus visite au moins $2^{(l-1)/k}$ points du chemin, et au plus tous les points du chemin, donc

$$l2^{(l-1)/k} \leq \mathbb{E}\left[T/G_1\right] \leq e(k+1)l2^{(l-1)/k}$$

Sous G2 Au mieux un seul raccourci est nécessaire à la convergence, sinon la séquence $\mathcal{S}_1$, puis $\mathcal{S}_2$, etc jusqu'à $\mathcal{S}_{(l-1)/k}$ garantit la convergence, donc :

$$\frac{k}{l-1}l^k \leq \mathbb{E}\left[T/G_2\right] \leq e\frac{l-1}{k}l^k$$

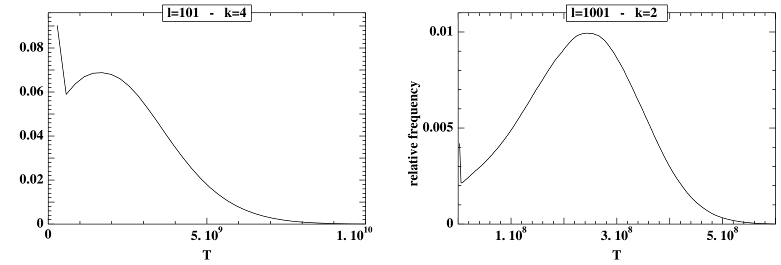


FIG 12 : *Histogramme théorique de T pour différentes valeurs des paramètres k et l.*

Estimation de $\mathbb{P}(G_1)$ pour $\alpha \geq 1/2$ On note $T_{ws} = T/G_1$. T_{ws} correspond au temps de convergence d'un processus sans raccourcis.
Comme T_{ws} et $T_{\mathcal{S}'}$ sont indépendants, et que $T_{\mathcal{S}'}$ suit une loi exponentielle, on obtient :

$$\mathbb{P}(G_1) = \mathbb{P}(T_{\mathcal{S}'} > T_{ws}) = \mathbb{E}\left[\exp -\frac{T_{ws}}{l^k k/(l-1)}\right]$$

On applique alors l'inégalité de Jensen [Fel71] :

$$\mathbb{P}(G_1) \geq \exp -\frac{\mathbb{E}[T_{ws}]}{l^k k/(l-1)}$$

Si $\alpha \geq 1/2$, alors, le fait que $elN_l \sim el2^{l^{1-\alpha}} \ll l^{l^\alpha} l^{\alpha-1}$ implique :

$$\mathbb{P}(G_1) \simeq 1, \text{ pour } k = [l^\alpha], \ \alpha \geq 1/2 \text{ et } l \gg 1$$

• **Second cas : k constant**

Rudolph démontre dans ce cas, que le temps de convergence T est polynomial, et ainsi, que la $1/l$ mutation prend des raccourcis sur les long k-path.
On peut alors décomposer T en fonction du nombre de raccourcis nécessaires à la convergence :

L'expression de la densité $p(\bar{t})$ de $\bar{T}$ (avec $T = el^k\bar{T}$) est obtenue en écrivant $\bar{T} = \sum_{j=1}^{M_L} \tau_j$, où τ_j (de loi exponentielle de moyenne 1) caractérise le temps d'attente entre deux raccourcis, M_L le nombre de raccourcis nécessaires à la convergence.
Or, la convergence nécessite que l'événement : 'raccourci S_j arrive avant S_1 (et pas les autres)', soit itéré (S_{j+i} arrive avant S_{1+i}) jusqu'à ce que la somme des indices j soit supérieure à $L = (l-1)/k$. Or la probabilité conditionnelle de ce type d'événements, sachant que le temps d'attente entre deux raccourcis est τ, est $e^{-(n-1)\tau}(1 - e^{-\tau})$. Ce qui donne :

$$p_{\bar{T}}(t) = \sum_{m=1}^{\infty} \sum_{\substack{n_1,\ldots,n_m=1 \\ n_1+\ldots+n_{m-1}<L \\ n_1+\ldots+n_m \geq L}}^{\infty} \int_0^\infty \ldots \int_0^\infty d\tau_1 \ldots d\tau_m \prod_{j=1}^{m} e^{-n_j\tau_j}(1-e^{-\tau_j})\delta_{\tau_1+\ldots+\tau_m=t},$$

$\longrightarrow$ L'histogramme théorique de T (Figure 12) présente un pic anormal proche de 0, qui donne lieu à un temps de convergence en dessous de la moyenne. Ce pic correspond au premier terme de $p(\bar{t})$ ($m = 1$), or la seule manière d'atteindre la solution en un seul raccourci, est que le raccourci $\mathcal{S}_1$ arrive en premier.
Ainsi, ce pic est associé à l'événement Ω_1 : raccourci $\mathcal{S}_1$ arrive en premier, i.e. avant $\mathcal{S}_2$, ... , $\mathcal{S}_{(l-1)/k}$ (donc $\mathbb{P}[\Omega_1] = 1/L$, $L = (l-1)/k$ étant le nombre de raccourcis possibles).

Proposition I.3.13 *Il existe un ensemble de réalisations Ω_1 avec la probabilité*

$$\mathbb{P}(\Omega_1) \underset{l\to\infty}{\simeq} kl^{-1}$$

tel que, sous Ω_1, la distribution conditionnelle du premier temps d'atteinte est exponentielle de moyenne $2el^{k-1}k$:

$$\mathbb{P}\left(T \geq a/\Omega_1\right) \underset{l\to\infty}{\simeq} \exp(-\frac{a}{2el^{k-1}k}).$$

Ceci montre que le temps de convergence sous Ω_1 $(2el^{k-1}k)$ est en gros, un facteur l^2 plus petit que le temps de convergence global $(\frac{el^{k+1}}{k\ln(l/k)}$ dans la Proposition I.3.7).

$\longrightarrow$ Le (minimum des) temps de convergence de $N = O(l)$ processus indépendants sera l^2 fois plus petit que le temps de convergence d'un seul processus, ceci, avec une forte probabilité : la probabilité que l'un des processus réalise l'événement Ω_1 (raccourci $\mathcal{S}_1$ arrive en premier), à savoir $1 - e^{-\frac{N}{l/k}}$.

$\longrightarrow$ Seconde conséquence, non moins importante : l'utilisation d'une population n'accélère pas la convergence (en nombre d'évaluations), par rapport au cas d'une population composée d'un seul individu ($N = 1$). En effet, le premier individu de la population qui réalise un raccourci $\mathcal{S}_j$, $j \neq 1$, colonise toute la population en moins de $O(N)$ générations à cause de l'élitisme. Ce qui anéantit toute chance de réaliser l'événement Ω_1. Ainsi, une population d'un seul individu donne les meilleurs résultats en temps de convergence.

Cependant, dans le cas $k = l^\alpha$, $\mathbb{P}(\Omega_1) = l^{\alpha-1}$ tend vers 0 quand l tend vers l'infini. Donc, le pic disparaît asymptotiquement.

Cette étude permet de conclure quant à l'existence de chemins exponentiels pour la $\frac{c}{l}$ mutation, à savoir les *long l^α-path* ($\alpha \geq 1/2$). Cependant, la question reste posée pour l'opérateur de croisement, qui prend des raccourcis conséquents sur ces chemins.

1.3 CHEMINS CONTRÔLÉS POUR LES ALGORITHMES GÉNÉTIQUES

But : A défaut de construire des chemins exponentiels pour l'opérateur de croisement, on propose de commencer par définir la notion de chemin pour un algorithme à base de populations de plus d'un individu ; puis on construit une famille de fonctions telle que, pour converger, l'algorithme génétique (avec mutation et croisement) soit obligé empiriquement de *suivre* un chemin arbitrairement prédéfini.

Dans cette partie l est supposé pair. Pour tout $s \in E$, on note s_{L} $(= s_1 \dots s_{\frac{l}{2}})$, la première moitié de s et s_{R} $(= s_{\frac{l}{2}+1} \dots s_l)$, la seconde.

Notion de chemin : On dit qu'un algorithme d'évolution suit le chemin défini par les points successifs : $(p^1, \dots p^i, p^{i+1} \dots)$, si le meilleur individu à chaque génération passe successivement par les schémas : $s_L = p^1_L$, $s_R = p^1_R$, $\dots$ $s_L = p^i_L$, $s_R = p^i_R$, $s_L = p^{i+1}_L$ $\dots$

Chemin contrôlé : Étant donné un ensemble de clefs aléatoires : $\{key^i\}_{i=1}^r$, avec $key^i \in \{0,1\}^{l/2}$, on construit une fonction fitness telle qu'un algorithme génétique soit amené à passer par les points successifs : $(key^1 key^1, \dots key^i key^i, key^{i+1} key^i, \dots key^r key^{r-1})$:

- Si $s_{\mathrm{L}} = key^i$, la fitness est définie sur s_{R} comme une *fitness de Hamming* vers key^i. L'algorithme génétique est donc guidé vers le point $key^i key^i$. Symétriquement, si $s_{\mathrm{R}} = key^j$, l'algorithme est guidé vers la chaîne $key^{j+1} key^j$. Plus formellement :

$$f(key^i s_{\mathrm{R}}) = 2i - 1 + \mathrm{d}_n(s_{\mathrm{R}}, key^i), \qquad i \in \{1, \dots, r-1\}.$$

$$f(s_{\mathrm{L}} key^j) = 2j + \mathrm{d}_n(s_{\mathrm{L}}, key^{j+1}), \qquad j \in \{1, \dots, r-1\}.$$

- Si $s = key^i key^j$ alors l'équation qui donne la plus grande fitness est utilisée, ainsi l'équation ci-dessus est appliquée si $j \geq i$.

- Si aucune des clefs n'apparaît dans la moitié droite ou gauche de s, la fitness $f(s)$ est définie comme une *fitness de Hamming* sur s_L vers la première clef :

$$f(s_{\mathrm{L}} s_{\mathrm{R}}) = \mathrm{d}_n(s_{\mathrm{L}}, key^1).$$

L'expérience montre que l'algorithme génétique [10] avec la $1/l$ mutation et un croisement (uniforme ou un-point) n'est pas plus performant qu'un processus (1+1)-ES sur des chemins contrôlés, construits avec 1001 clefs aléatoires uniformément choisies dans $\{0,1\}^{l/2}$ pour l=100.

L'incapacité du croisement (et de la population) à améliorer le temps de convergence est probablement due au manque de diversité génétique dans les populations courantes. En effet, une séléction par le rang, avec une *fitness de Hamming*, amène très vite tous les individus au voisinage de la prochaine clef.
Ceci motive l'utilisation de nouvelles procédures de séléction, moins fortes, ou dépendantes de critères autres que la seule fonction fitness. Il serait aussi intéressant d'étudier l'influence du *sharing* [11] sur la convergence des GA sur ces chemins contrôlés.

Application au *long k-path* Si on considére les positions $1 + \alpha(k+1)$, $\alpha \in \mathbb{N}$ sur le *long k-path* de Rudolph, on obtient un sous-chemin $S(l,k)$ de longueur $|S(l,k)| = \lfloor 2^{(l-1)/k} \rfloor$. Les points de $S(l,k)$ sont deux à deux distants d'au moins k bits.
On considère maintenant un chemin contrôlé (fonction f) construit à partir des points successifs du sous-chemin $S(l/2, \sqrt{(l)})$:

En combinant les résultats des deux dernières parties, concernant la convergence d'un processus (1+1)-ES sur les *fitness de Hamming* et les *long k-paths*, on démontre qu'un processus (1+1)-ES, presque sûrement, suit ce chemin contrôlé sans prendre de raccourcis (moyennant toutefois une petite modification de la fonction (*fitness de Hamming*) qui mène d'une clef à une autre).

[10] GA avec une population de taille 100 et une séléction par le rang

[11] Le *sharing* consiste à diminuer artificiellement la fitness des individus trop proches en distance les uns des autres (dans un rayon choisi par l'utilisateur).

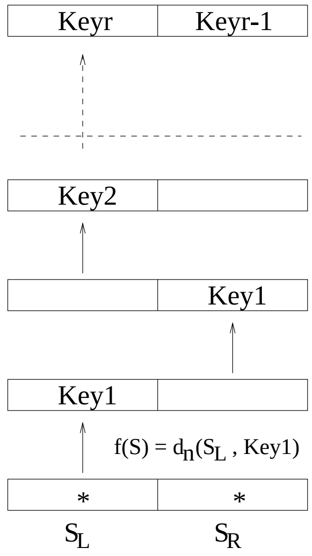


FIG 2 : Chemin contrôlé pour les algorithmes génétiques. Le chemin 'commence' par *fitness de Hamming* sur s_L vers key_1.

2 Notion de difficulté

Le second chapitre propose plusieurs études comparatives des performances des algorithmes génétiques.

La première partie est dédiée à la comparaison *a posteriori* des résultats des algorithmes génétiques, et précise la notion de *comportements similaires des algorithmes génétiques*: Les *fitness de Hamming* sont proposées comme base commune de comparaison, permettant de relativiser le biais induit par les opérateurs, initialisation et autres paramètres. Cela permet d'une part, de comparer les résultats du même algorithme sur des fonctions fitness ayant des distributions et des solutions différentes ; d'autre part, de comparer les résultats de différents algorithmes et représentations.

Toute la suite concerne les outils de comparaison *a priori*. Ces outils ont été développés dans le but de reconnaître les fonctions faciles ou difficiles pour les algorithmes génétiques, avant de voir ses résultats.

Plusieurs mesures statistiques des paysages de fitness (corrélation, épistasie …) sont comparées et généralisées. Les classes de fonctions décrites par ces mesures sont identifiées et comparées à des classes connues de fonctions *faciles* pour les algorithmes génétiques. Cela montre les limites de ces mesures pour caractériser –ne serait ce que– les fonctions faciles pour ces algorithmes. Dans la même logique, l'étude d'une classe de fonctions fitness partageant le même graphe de corrélation (relation 'Fitness $\times$ Distance à la solution'), révèle une grande sensibilité des algorithmes génétiques à la distribution des points dans le graphe de corrélation et à l'initialisation.

Ayant vu que le comportement des algorithmes génétiques ne peut être prédit par des propriétés sommaires des paysages de fitness, les deux dernières parties proposent une étude comparative de différents opérateurs de croisement et de plusieurs (nouvelles) procédures d'initialisation. Des pistes sont proposées pour choisir un opérateur, et une initialisation *a priori*, en fonction du problème à optimiser (la section III.6.2 illustre également ceci dans le cas des représentations de longueur variable).

2.1 BASE DE COMPARAISON DES RÉSULTATS DES ALGORITHMES GÉNÉTIQUES

But : En l'absence d'une définition rigoureuse de la notion de 'comportements similaires des algorithmes génétiques', une base relative de comparaison est proposée, et distingue notamment le point de vue temporel de celui de la trajectoire spatiale de l'algorithme.

Quand il s'agit de comparer les résultats de l'algorithme génétique sur des fonctions fitness ayant des distributions et des solutions différentes, un problème d'échelle se pose : la quantité d'amélioration en fitness [RVSK97] n'est pas significative. Par ailleurs, le temps de convergence [SCED89] ou taux de succés [Tan89, CJ91] ne tient pas compte du biais introduit par l'initialisation ou autres paramètres de l'algorithme.

Ainsi, le recours à une **base de référence** semble inévitable. Les *fitness de Hamming* sont proposées dans la suite comme base de comparaison relative : le comportement d'un algorithme génétique sur une fonction donnée, est relativisé par rapport à son comportement sur la *fitness de Hamming* correspondante.

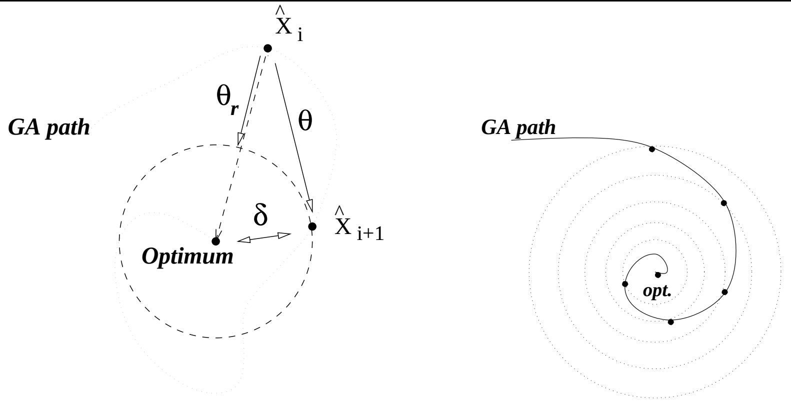
Étant donné une fitness f, la méthode consiste à :

1. Lancer l'algorithme (GA) avec la fitness f, et noter le X_{N_c} le meilleur trouvé.
2. Relancer le même GA, avec la même population initiale, mais avec une *fitness de Hamming* vers X_{N_c} $(\mathcal{H}_{X_{N_c}}(x) = l - d(x, X_{N_c}))$.
3. Pour chaque mesure m (exemples ci-dessous), calculer le rapport des valeurs obtenues pendant l'exécution des étapes 1 et 2 $m(f)/m(\mathcal{H}_s)$.

La suite propose un certain nombre de mesures pour caractériser le comportement de l'algorithme génétique, en séparant notamment le point de vue temporel de celui de la (longueur de la) trajectoire spatiale. On note $\hat{X}_i$ le meilleur individu à la génération i.

- **Temps de convergence :** N_c
- **Distance moyenne à** $\hat{X}_{N_c}$ **:** $\delta = \frac{\sum_{1,N_c} d_n(\hat{X}_i, \hat{X}_{N_c})}{N_c}$
- **Longueur de la trajectoire :** $\theta = \sum_{1,N_c-1} d_n(\hat{X}_i, \hat{X}_{i+1})$
- **Longueur de la trajectoire radiale :**
  $\theta_{\mathbf{r}} = \sum_{1,N_c-1} |d_n(\hat{X}_i, \hat{X}_{N_c}) - d_n(\hat{X}_{i+1}, \hat{X}_{N_c})|$

Comme illustré dans la figure 20, θ caractérise la distance entre les meilleurs individus à chaque génération, θ_r leur différence en distance à $\hat{X}_{N_c}$, et δ leur proximité à $\hat{X}_{N_c}$.



(a) (b) θ_r Minimum

FIG 20 : *Les cercles représentent des points équidistants de l'optimum. (a) Points de vue sur la trajectoire de l'algorithme génétique : θ caractérise la distance entre les meilleurs individus à chaque génération, θ_r leur différence en distance à l'optimum, et δ leur proximité à l'optimum. (b) θ_r est minimale, ssi la trajectoire de l'algorithme se rapproche de l'optimum à chaque amélioration en fitness.*

θ et θ_r ne sont pas forcément corrélées : si θ est minimale (peu réaliste), alors θ_r l'est aussi. Mais, θ_r peut être minimale, avec un θ quelconque: comme le montre la Figure 20 (b), θ_r est minimale (et vaut $d_n(\hat{X}_1, \hat{X}_{N_c})$), ssi la trajectoire de l'algorithme se rapproche de $\hat{X}_{N_c}$ à chaque amélioration en fitness. C'est le cas avec les *fitness de Hamming*, où typiquement θ vaut 5 à 10 fois θ_r.

Cette méthode peut facilement être **généralisée à différentes représentations**, simplement en remplaçant la *fitness de Hamming* par la fitness de distance génotypique de la représentation concernée. Ainsi, on peut aisément comparer des résultats obtenus sur des problèmes différents, chacun dans une représentation différente, et éventuellement avec un algorithme différent.

Cependant, le choix des *fitness de Hamming* est plutôt arbitraire, et probablement trop sommaire. En effet, il permet uniquement de mesurer la différence du comportement de l'algorithme génétique de celui qu'il a sur la fonction de base. Il serait plus constructif de comparer le comportement de l'algorithme sur **une série de fonctions de base**. Ainsi, on peut identifier la base qui donne le plus proche comportement de l'algorithme génétique. Les *NK-landscapes* [Kau89], par exemple, peuvent fournir une suite de fonctions base indexées par la valeur K (une *fitness de Hamming* correspond d'ailleurs à K=0).

$\longrightarrow$ Ceci permettrait de classer les fonctions fitness selon le type de comportements dynamique d'un algorithme donné, et peut orienter le choix des paramètres de l'algorithme pour améliorer la convergence sur une fitness inconnue.

$\longrightarrow$ La question cruciale est alors de choisir une famille générique de fonctions de base.

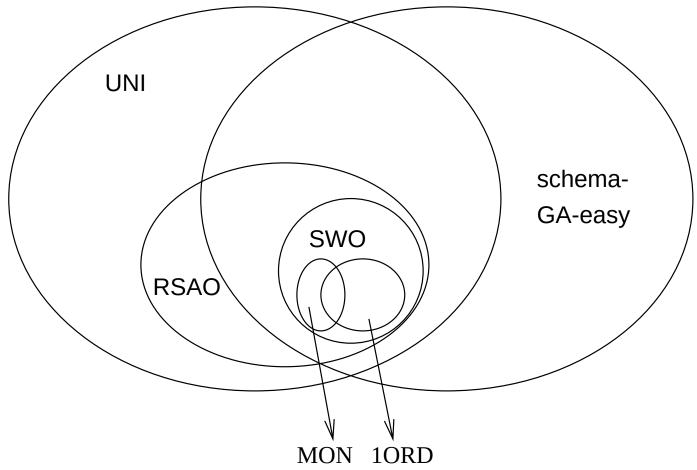


FIG 25 : Classes de fonctions 'faciles' pour les GA. MON : monotones en fonction de la distance à l'optimum (ex. OneMax). 1ORD : à variables séparables (pas d'épistasie). SWO : optimisables site par site (II.3.3.3). RSAO : optimisables par un algorithme de descente (local HC) en moins de l pas. UNI : pas d'optima locaux en plus du global (ex. [HGD94]). Schéma-facile : à chaque ordre, le schéma de meilleure fitness contient l'optimum.

2.2 MESURES STATISTIQUES DES PAYSAGES DE FITNESS

But : Étude des mesures statistiques des paysages de fitness, en identifiant et comparant les classes de fonctions qu'elles caractérisent. Ceci permet de conclure quant à la signification et limite de ces mesures (en relation au comportement de l'algorithme génétique), mais aussi de toute autre mesure potentielle.

Des observations empiriques ont identifié différentes sources de difficulté pour un AG dans un paysage de fitness. Une étude intensive des limites et propriétés de ces sources de difficulté, révèle une totale décorrélation entre les mesures qui en découlent (corrélation, épistasie) et les résultats (qualité, vitesse) de convergence de l'AG sur différentes classes de fonctions fitness (ex. monotones, unimodales, non-*deceptives*).

Signification des mesures : Ce que mesure l'épistasie (resp. la corrélation) c'est – par définition – la distance (resp. l'écart) d'une fonction à une classe de référence : la classe des fonctions du premier ordre (resp. de *hamming*). Or ces classes sont loin de contenir toutes les fonctions faciles à optimiser pour l'algorithme génétique, d'où l'absence manifeste de tout lien entre les valeurs de ces mesures et la difficulté d'un paysage de fitness pour l'algorithme génétique :

⟶ Pour construire une mesure significative, il faudrait d'abord identifier la classe de toutes les fonctions faciles pour l'algorithme génétique, et espérer que la difficulté soit proportionnelle à la distance à cette classe !

La figure 25 montre les relations entre plusieurs classes de fonctions faciles, en majorité [12] (Def. II.3.7), pour l'algorithme génétique (relations démontrées dans II.3.5). La classe *RSAO*

[12] les *longpaths* (dans *UNI*), peuvent être très longs à optimiser par un GA sans croisement, (temps polynômial à exponentiel en l, cf. I.3). Il faudrait donc mieux définir les limites du terme 'facile pour l'algorithme génétique'. De plus, les fonctions schéma-faciles ne sont pas forcément faciles pour l'algorithme génétique (cf. II.3).

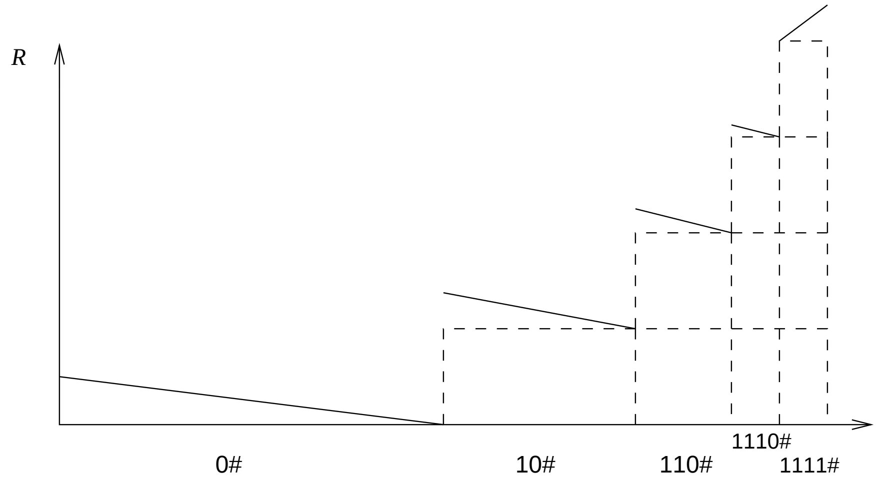


FIG. 26 Description schématique de l'association trompeuse des problèmes OneMax et 4 ZeroMax.

(optimisables site par site), facile pour l'algorithme génétique, contient les classes de référence de toutes ces mesures, mais contient aussi des fonctions épistatiques et mal corrélées:
Ci-dessous est construite une famille de fonctions de $RSAO$, faciles pour l'algorithme génétique, épistatiques et mal corrélées :

<u>Exemple : Association trompeuse des problèmes OneMax et ZeroMax</u>

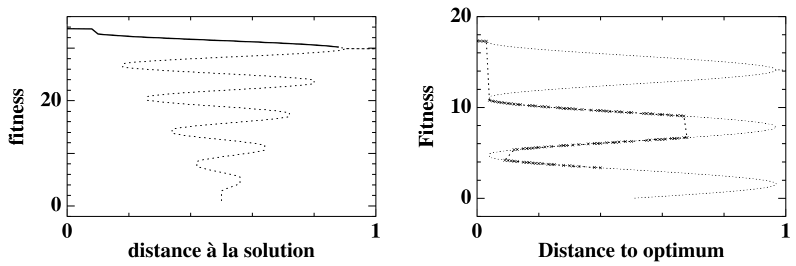
Soit $n \in \mathbb{N}$, tel que $0 < n < l$, un paramètre de contrôle. La fonction f_n est construite comme suit :
Sur le schéma $0\#^{l-1}$, f_n correspond à un problème ZeroMax défine sur $l-1$ sites. Sur le schéma $10\#^{l-2}$ f_n correspond à un problème zeromax (sur $l-2$ sites) majoré de $2l$ ... Sur le schéma $1^{n-1}0\#^{l-n}$, f_n correspond à un ZeroMax (sur $l-n$ sites) majoré de $2nl$.
La fonction f_4 est représentée Fig. 26, et définie ci-dessous :

$$\begin{aligned}
f_4(0s_1 \dots s_{l-1}) &= \text{ nombre de zéros dans } s_1 \dots s_{l-1} \\
f_4(10s_2 \dots s_{l-1}) &= \text{ nombre de zéros dans } s_2 \dots s_{l-1} + 2l \\
f_4(110s_3 \dots s_{l-1}) &= \text{ nombre de zéros dans } s_3 \dots s_{l-1} + 4l \\
f_4(1110s_4 \dots s_{l-1}) &= \text{ nombre de zéros dans } s_4 \dots s_{l-1} + 6l \\
f_4(1111s_4 \dots s_{l-1}) &= \text{ nombre de zéros dans } s_4 \dots s_{l-1} + 8l
\end{aligned}$$

L'expérience montre qu'un GA simple converge systématiquement rapidement (même ordre que le temps de convergence sur une *fitness de Hamming* vers la solution).

Autre point de vue sur les mesures : Le problème de ces mesures peut aussi être expliqué par d'autres considérations :

• La variance d'épistasie proposée par Davidor [Dav91a] détécte uniquement l'absence d'interactions entre les allèles. Pourtant, si l'on considère le modèle épistatique en fonction de l'ordre des interactions, comme proposé dans [RW95b] :



(a) (b)

FIG 3 : *Deux relations FD (fitness × distance à la solution). (b) Visualisation de trajectoire de l'AG sur le graphe de corrélation.*

$$\begin{aligned} f(s) = \quad & constante + \sum_{i=1}^{l} (\text{ effet de } s_i) \\ & + \sum_{i=1}^{l-1} \sum_{j=i+1}^{l} (\text{ interaction entre } s_i \text{ et } s_j) \\ & + \qquad \ldots \\ & + (\text{ interaction entre } s_1\ s_2 \ldots \text{ et } s_l) + \text{ erreur aléatoire} \end{aligned}$$

Alors, l'information la plus importante du point de vue de la difficulté des paysages, n'est ni la présence, ni la quantité ou l'ordre des interactions entre allèles, mais le **type d'interaction**. des fonctions monotones (en fonction de la distance à la solution) illustre parfaitement un cas d'interactions positives (du premier ordre), qui est facile à optimiser. L'exemple des problèmes NNI (fitness ci-dessous) illustre un cas d'interactions du second ordre faciles ou difficiles pour l'AG, selon le choix des paramètres h_i, k_i, l_i (pour améliorer la lisibilité, on pose $h_l \equiv h_0$ et $s_l \equiv s_0$) :

$$f(s) = \sum_{i \in \{1,\ldots,l\}} h_i(1 - s_i)s_{i+1} + \sum_{i \in \{1,\ldots,l\}} (k_i s_i + l_i(1 - s_i)).$$

• La corrélation entre fitness et distance à la solution, réduit l'information que l'on a sur le paysage aux couples $(f(x), d(x, opt))$, $x \in E$, ie. au graphe de corrélation. Ainsi, il faudrait d'abord se poser la question : Dans quelle mesure la relation entre la fitness et la distance à la solution peut-elle caractériser les types de comportements des GA? Plus concrètement, la section II.4 donne des éléments de réponse à la question :

Combien de comportements différents de l'algorithme génétique peut-on observer sur des fonctions partageant le même graphe de corrélation ?

2.3 RELATIONS FITNESS × DISTANCE :

But : Etude d'une classe de fonctions fitness partageant le même graphe de corrélation (Relation Fitness × Distance à la solution). Inventaire des comportements de l'algorithme génétique au sein de cette classe, et de sa sensibilité à de petites modifications de la relation FD initiale.

L'interprétation du graphe de corrélation est une entreprise hasardeuse. En effet, deux points x et x' de E, contigus sur le graphe de corrélation, ne sont pas forcément proches en distance : si $d(x, opt) = d(x', opt) - 1 = l/2$ alors x et x' peuvent être complémentaires (en $d \simeq l/2$). Par conséquent, à partir des couples (fitness,distance) on peut difficilement savoir si il existe des chemins faisables ou non : La distribution des points dans le graphe est capitale pour déterminer le comportement des GA.

Ces phénomènes sont concrètement illustrés par l'étude de deux relations particulières (Fig. 2.3). Une classe de fonctions partageant une même relation FD est construite dans la section II.4. Au sein de cette classe le comportement des GA varie du facile à l'impossible à résoudre.

Plus généralement, le comportement des GA dépend d'un nombre de détails indétectables par l'analyse du graphe de corrélation :

• Une simple translation de la relation FD initiale le long de l'axe des distances peut complètement faire changer le type de convergence.

• Une redistribution des points transforme la ligne droite marquée en gras dans la figure 2.3-a, initialement facile à parcourir pour le simple GA, en une ligne contenant un optimum local infranchissable (présence de points quasiment complémentaires à proximité les uns des autres en $d = l/2$).

• Une perturbation insignifiante (relativement à l'écart minimum entre les valeurs de fitness) de la relation initiale, transforme un problème difficile en un problème facile.

Le graphe de corrélation a cependant un rôle certain de visualisation, par exemple pour choisir une procédure d'initialisation, facteur important de rapidité et d'efficacité pour les GA.

2.4 CORRÉLATION ET INITIALISATION

But : Deux nouvelles procédures d'initialisation basées sur différents critères d'uniformité dans l'espace de recherche sont proposées, et comparées sur plusieurs fonctions fitness. Le graphe de corrélation est testé comme outil potentiel pour choisir une initialisation *a priori.*

Procédure uniforme (BU) : La majorité des GA utilisent une initialisation dite uniforme, où chaque bit prend la valeur 0 ou 1 avec une probabilité $p = 0.5$.
L'argument majeur en faveur de cette procédure étant que chaque point de l'espace $\{0, 1\}^l$ a la même probabilité $1/2^l$ d'être généré. En plus, la diversité bit par bit est encore maximale : en chaque position il y a autant de chance d'avoir 1 que 0.

Cependant, si on considère le nombre de uns (noté $\sharp 1$) dans chaque individu $s \in E$, on a

$$P[\sharp 1 = k] = C_l^k p^k (1-p)^{l-k}$$

Or, pour l grand (e.g. $l \geq 30$), cette loi binomiale peut être approximée par une Gaussienne $N(l, p)$, ce qui implique que $\sharp 1$ est dans un domaine très restreint. Par exemple, pour $l = 900$, 99% des individus ont un nombre de uns dans $[411, 489]$.

Procédure de densité uniforme (UC) : Pour chaque individu,

1. Choisir r la densité de uns, uniformément dans $[0, 1]$
2. Pour chaque bit, choisir la valeur 1 avec la probabilité r

Cette fois-ci, la distribution de $\sharp 1$ est uniforme. De plus, la diversité bit par bit est encore maximale.

Cependant, la probabilité d'avoir une séquence de k uns successifs à partir d'une position donnée [13] est plutôt faible : $1/(k+1)(=\int_0^1 r^k dr)$ (contre $1/2^k$ pour BU).

Procédure de blocs homogènes (HB) : Cette procédure est définie par la superposition de blocs de uns tirés (centre, longueur du bloc).
Par exemple, le bloc de uns de caractéristiques (c, n) met à 1 les bits des positions j avec $max(1, c - n/2) \leq j < min(l, c + (n+1)/2)$. Les autres bits ayant par défaut la valeur 0.
Une densité de uns est choisie uniformément dans $[0, 1]$, les blocs sont ajoutés tant que cette densité n'est pas atteinte, de la manière suivante :

1. L_{loc} est choisi uniformément dans $[0, L_{max}]$, (L_{max} typiquement $l/10$).
2. Le centre c et longueur n de chaque bloc est choisi uniformément respectivement dans $[0, n]$ et $[0, L_{loc}]$.

Comparaison des procédures : Dans tous les cas de fitness considérés, les nouvelles procédures donnent des résultats au moins équivalents à ceux de l'initialisation uniforme (BU). Même dans les cas où la solution est typiquement bien adaptée à la procédure uniforme (BU), i.e. faite de uns et de zéros alternés, les deux autres procédures montrent des performances au moins équivalentes (en temps de convergence du simple GA correspondant).

Le graphe permet d'évaluer la diversité d'un échantillon aléatoire, de par l'étendue et la distribution des distances et des fitness. Ainsi, il peut aider à choisir l'initialisation la plus adaptée au problème, le but étant de trouver le bon compromis entre diversité et meilleure fitness (dans l'échantillon aléatoire).

De plus, les distances peuvent être calculées relativement au meilleur point de l'échantillon. Cela en fait un outil pratique, puisque l'optimum est rarement connu quand on cherche à optimiser une fonction. Les résultats les plus spectaculaires sont obtenus avec les représentations de taille variable, où la procédure d'initialisation dépend de plusieurs paramètres interdépendants (chapitre III).

2.5 COMPARAISON DES OPÉRATEURS DE CROISEMENT

But : Confrontation des mesures connues pour caractériser la performance des opérateurs de croisement. Une nouvelle méthode, plus efficace pour choisir un opérateur *a priori*, est proposée.

[13] Ce n'est pas la probabilité qu'un individu contienne une séquence de k uns.

Si le comportement des GA dépend directement du paysage de fitness, on ne peut ignorer le rôle crucial des opérateurs sur ce paysage. Pour comparer les opérateurs de croisement ou de mutation, plusieurs méthodes ont été proposées, suite à des observations empiriques, mais sur des problèmes différents à chaque fois. Ces méthodes ne sont pourtant pas forcément compatibles. On peut d'ailleurs distinguer deux tendances. A partir d'un échantillon de parents (aléatoire), et de leurs enfants, la comparaison des opérateurs se fait par :

1. Étude de la distributions des fitness des enfants indexées par les fitness de leurs parents respectifs.
2. Étude de la distributions des fitness des enfants, indépendamment de leurs parents.

Ainsi, dans le premier cas, un opérateur est dit meilleur si il maximise : la différence moyenne des fitness entre parents et enfants (positive ou négative), l'amélioration moyenne en fitness [FG96] (positives) ou le taux de corrélation [MdWS91] (si les meilleurs parents donnent les meilleurs enfants). Dans le deuxième cas, un opérateur est dit meilleur si il maximise : la fitness moyenne des enfants ou l'amélioration moyenne au dessus d'une certain seuil [FG96].

Une confrontation de ces mesures sur un même ensemble de problèmes binaires, et deux opérateurs de croisements différents (uniforme et un-point), révèle d'une part des contradictions entre les résultats de ces méthodes, et d'autre part, des complémentarités intéressantes entre certaines. Cependant, aucune de ces méthodes n'est efficace sur tous les problèmes à la fois.

La qualité essentielle d'un opérateur semble être le potentiel d'un opérateur à produire quelques enfants de très bonne fitness. Ceci, indépendamment des parents respectifs de ces enfants, de la quantité ou nombre d'améliorations en fitness, apportées par ces enfants. Ainsi, comparer **les valeurs maximales** des fitness des enfants, et l'évolution de celle-ci quand la fitness des parents augmente, serait une mesure plus efficace.

Cependant, cette conclusion, est basée sur une logique de maximisation du profit (en fitness) à court terme. Et cette situation n'est pas toujours compatible avec une logique d'optimisation globale, pour laquelle le seul impératif est de trouver le meilleur global à la fin, quelle que soit la progression au début. Dans ce cas, les qualités d'un opérateur dépendraient probablement de sa capacité d'exploration (notion qui reste à définir par ailleurs).

Il est important de noter la dualité représentation/opérateurs : changer de représentation revient aussi à changer d'opérateur. La conclusion ci-dessus a donc pu être utilisée de manière particulièrement efficace pour comparer les différentes représentations dans le cadre de l'optimisation des formes en mécanique des structures, présenté ci-dessous.

3 Représentations de longueur variable

L'identification d'inclusions est un problème théoriquement mal posé, pour lequel des méthodes numériques déterministes se révèlent instables. Ainsi, les résultats obtenus sur ce problème inverse par application des algorithmes génétiques vont bien au-delà de l'état de l'art du domaine : la connaissance d'un nombre suffisant de déplacements des points

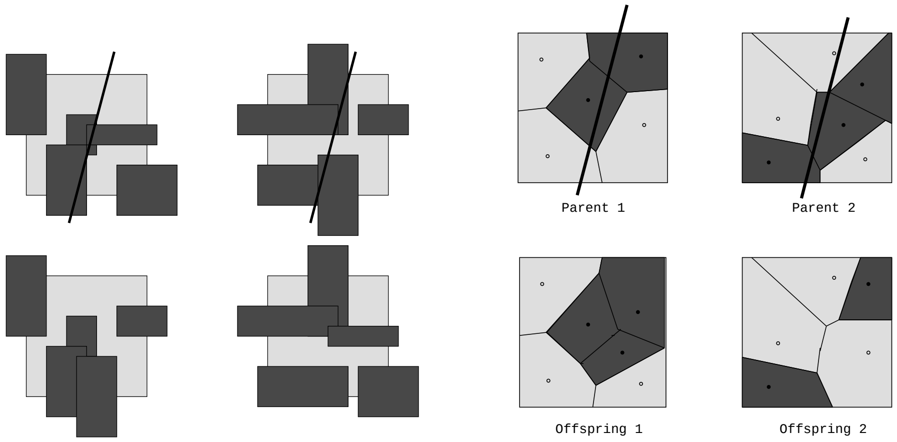


FIG 4 : *: Croisement par échange* géométrique *de rectangles ou de sites*

de la surface d'une structure faite de deux matériaux connus, sous l'action d'un ensemble de chargements, permet d'identifier la répartition de ces deux matériaux à l'intérieur de la structure.

Cependant, l'application des algorithmes évolutionnaires à ce problème doit faire face au problème crucial du choix de la représentation des solutions, ie. de l'espace de recherche. Dans la suite, deux nouvelles représentations sont proposées et comparées sur ce problème avec la représentation, plus classique, en matrice binaire.
Dans ces nouveaux espaces de représentation, un individu est une liste non ordonnée et de longueur variable. Une approche générale pour construire des distances génotypiques sur ces nouveaux espaces de représentation est proposée, ainsi que des outils d'analyse et de comparaison de ces distances, en relation avec les opérateurs d'évolution.

Finalement, à l'aide de ces distances génotypiques, les méthodes développées dans le chapitre II dans le cadre binaire (comparaison des opérateurs et des procédures d'initialisation), sont mises en œuvre pour choisir une initialisation et une représentation *a priori* sur le problème d'identification d'inclusions.

3.1 APPLICATION EN MECANIQUE DES STRUCTURES

Le problème Une structure mécanique est composée de deux matériaux connus, dont la répartition dans la structure est inconnue. On dispose cependant de données expérimentales concernant le déplacement de certains points à la surface de la structure, sous l'action de forces ponctuelles appliquées sur cette surface. Il s'agit alors de trouver la répartition géométrique des deux matériaux à partir de ces données expérimentales.
L'identification est un problème inverse. En effet, on peut calculer par éléments finis les déplacements simulés d'une structure connue sous l'action des mêmes forces surfaciques. La fitness d'une structure est alors le moindre carré des différences entre les déplacements simulés et expérimentaux (sur la structure inconnue). Le problème d'identification d'inclusions consiste à minimiser cette fitness.

Les représentation L'espace des **phénotypes** est celui des partitions d'un domaine donné Ω dans $\mathbb{R}^2$, en deux sous ensembles à frontières continues. La représentation la plus classique est la représentation binaire où une matrice de bits définit la répartition des deux matériaux comme étant respectivement les points à 1 et à 0. Cependant, une faiblesse majeure de la représentation binaire est que sa complexité dépend directement du maillage. En effet, si la taille du maillage augmente, la taille du problème suit.

En alternative, on considère de nouvelles représentations indépendantes du maillage, où le **génotype** est une liste de rectangles (ou polygones quelconques), ou un diagramme de Voronoï :

• Une liste de n rectangles (Fig. 4) établit une partition du plan : la partie recouverte de rectangles, et celle qui ne l'est pas. Un individu x de l'espace de recherche $\mathcal{G}$, est alors défini par :

$$x = (n, R_1, \dots, R_n),\ 1 \leq n \leq N_{max}$$

On obtient alors deux représentations : selon que les rectangles correspondent à la valeur 1 ou 0, ces représentations sont notées respectivement **1-rectangles** et **0-rectangles**.

• Un diagramme de Voronoï est une liste de n sites (Fig. 4) (points du plan dans Ω). A chaque site S_i est associée une cellule Voronoï (c_i), contenant l'ensemble des points les plus proches de ce site. En associant à chaque site un label de 0 ou 1, on obtient une partition du plan en deux parties.

$$x = (n, (S_1, c_1), \dots, (S_t, c_t)),\ 1 \leq n \leq N_{max},\ c_i \in \{0, 1\}$$

Résultats La représentation binaire donne des résultats de qualité inférieure à ceux obtenus avec les autres représentations. La figure 53 montre les courbes de convergence avec plusieurs représentations. L'initialisation binaire uniforme, inadaptée à ce problème [14], est remplacée par une initialisation identique à celle des 1-rectangles. La figure 54 montre les structures trouvées avec chaque représentation.

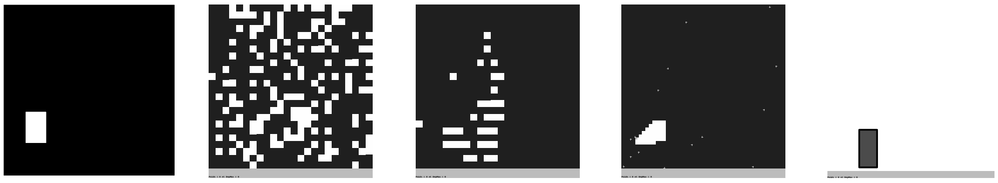
- Les représentations par rectangles sont plus performantes quand les rectangles représentent le matériau le moins présent (en densité) dans la structure. Autrement, on observe une dégradation sensible des performances qui restent cependant au-dessus des résultats binaires, mais en dessous de ceux de Voronoï .

- Quand les deux matériaux de la structure solution ont des densités similaires, la performance des diagrammes de Voronoï dépasse celle des rectangles.

- Bien sûr la forme des matériaux dans la solution peut biaiser les résultats. Par exemple, une forme triangulaire favorise la représentation par triangles. Ainsi, choisir une représentation a priori (entre rectangles, triangles et Voronoï), sans avoir d'information sur la solution, est l'un des enjeux majeurs en optimisation des formes.

Seulement, ces informations ne sont accessibles qu'*a posteriori*, une fois le problème résolu. D'où l'interêt de développer des méthodes d'analyse des paysages de fitness, en vue d'obtenir des informations *a priori*. Dans cette perspective, on se propose de construire des distances génotypiques sur ces espaces.

[14] L'initialisation uniforme défavorise la représentation binaire (population initiale très loin de la solution) et donne des temps de convergence jusqu'à 10 fois supérieurs à ceux des autres représentations.



Solution | Binaire Init. uniforme | Binaire Init. 1-rect. | Voronoï | 1-Rectangles

FIG 54 Meilleure structure après 300 générations sur le problème dissy, avec un maillage 48×48 *(les couleurs sont inversées pour les rectangles). La représentation binaire est beaucoup moins performante que les autres.*

3.2 DISTANCES GENOTYPIQUES

Proposition III.5.1
On note $\mathcal{G}$ l'espace des listes non ordonnées d'éléments d'un espace $\mathcal{A}$.
Etant donné δ, une distance bornée dans $\mathcal{A}$, Δ définit une distance dans $\mathcal{G}$

$$\Delta(x,x') = \min_{\mu \in M(x,x')} \Big(\sum_{1}^{n'} \delta(a_i, a'_{\mu(i)})\Big) + (n-n')diam(\delta)$$

où $x = (a_i)_{i\in[1,n]}$, $x' = (a'_i)_{i\in[1,n']}$, *et $M(x,x')$ l'ensemble des permutations sur [1,n'].*

Dans le cas où $\mathcal{A}$ est un ensemble de rectangles,
Une distance possible sur $\mathcal{A}$, est la distance de *Hausdorff* entre rectangles ou cellules Voronoï . Seulement, étant donné d une distance sur $\mathcal{A}$, la proposition ci-dessous propose une construction générale de distances sur δ, où d est un cas particulier.

proposition III.5.2
Soit $Sim : \mathcal{A} \to \mathbb{R}_+$ une fonction sur $\mathcal{A}$. Pour tout $(a,b) \in \mathcal{A}^2$, on note $|a-b| = Sim(a,a) - Sim(a,b)$.
Si il existe g et h tel que

$$Sim(a,b) = g(A_a, A_b)\, h(d(a,b)),$$

avec
d une distance sur $\mathcal{A}$, A_a (notation pour $A(a)$) une fonction strictement positive sur $\mathcal{A}$ (ex. l'aire des rectangles ou cellules Voronoï),
$h : [0,+\infty] \to [0,1]$ une fonction décroissante telle que $h(0) = 1$ et

$$1 + h(d+d') - h(d) - h(d') \geq 0 \;\; \forall (d,d') \in \mathbb{R}_+^2$$

$g : [0,\infty] \to \mathbb{R}_+$ satisfait l'inegalité

$$1 + g(\alpha,\beta) \geq g(\alpha,1) + g(1,\beta) \;\; \forall (\alpha,\beta) \in \mathbb{R}_+^2$$

ainsi que l'une des deux conditions ci-dessous, pour tout $(s,s') \in \mathbb{R}_+^2$:

$$\left\{ \begin{array}{l} g(s,s) = s \\ g(ks,ks') = k\, g(s,s')\, \forall k \in \mathbb{R}_+ \\ 0 \leq g(s,s') \leq min(s,s') \end{array} \right. \quad ou \quad \left\{ \begin{array}{l} g(s,s) = 1 \\ 0 \leq g(s,s') \leq 1 \end{array} \right.$$

alors

$$\delta(a,b) = (|a-b| + |b-a|)/2$$

est une distance sur $\mathcal{A}$.

Six expressions possibles pour δ sont données ci-dessous, respectivement en échelle linéaire et exponentielle :

$$\delta(a,b) = \begin{cases} \dfrac{d(a,b)}{ßC} \\ \dfrac{A_a + A_b}{2} - min(A_a, A_b)(1 - \dfrac{ßd(a,b)}{ßC}) \\ 1 - \dfrac{2\,min(A_a, A_b)}{A_a + A_b}(1 - \dfrac{ßd(a,b)}{ßC}) \end{cases}$$

$$\delta(a,b) = \begin{cases} 1 - e^{\frac{ß - d(a,b)}{ßC}} \\ \dfrac{A_a + A_b}{2} - min(A_a, A_b) e^{\frac{ßd(a,b)}{ßC}} \\ 1 - \dfrac{2\,min(A_a, A_b)}{A_a + A_b} e^{\frac{ßd(a,b)}{ßC}} \end{cases}$$

L'analogie avec la distance de *Hamming* en binaire est évidente : si on pose $Sim(a,b) = 1 - abs(a-b)$, $(a,b) \in \{0,1\}^2$, abs étant la valeur absolue, on obtient la distance de Hamming.

Chaque choix d'une distance δ sur $\mathcal{A}$, peut aussi donner lieu à une distance de *Hausdorff* sur $\mathcal{G}$. Définie, ci-dessous, la distance de *Hausdorff* est une distance entre ensembles, qui écarte naturellement le problème de la taille variable des génotypes. Cependant, ces génotypes ne sont pas des ensembles, mais des listes où un élément peut apparaître plusieurs fois. Cet événement peut être négligé dans le cas des listes de coordonnées réelles et de petite taille ($N_{max} = 50$ ici).

Distance de *Hausdorff* : *Soit d une distance sur E, $d(a,B) = min\{d(a,b), b \in B\}$ avec $a \in E$, $B \subset E^n$. La distance de* Hausdorff *(Θ) entre deux sous ensembles de E, $A \subset E^{n'}$, $B \subset E^n$ est définie par:*

$$\Theta(A,B) = max\{d(a,B) \ : a \in A\} + max\{d(b,A) \ : b \in B\}$$

Comparaison des distances :

Notations : Dans la suite, on considère par défaut, les distances δ avec h en échelle exponentielle (peu différentes en pratique des distances en échelle linéaire), et l'on compare les distances dans le cas de la représentation 1-rectangle : En reprenant les notation de la proposition III.5.2 au-dessus, on note respectivement Δ_H, Δ_A et Δ_{HA} les distances correspondant à un choix de $g(A_a, A_b)$ égal à 1, $\min(A_a, A_b)$ et $2\ \min(A_a, A_b)/(A_a + A_b)$. A_a étant l'aire d'un rectangle a.

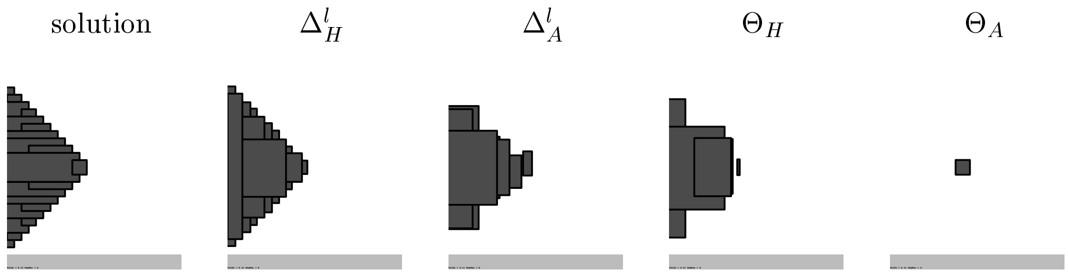


FIG 57 Génotypes (médiane de 21 essais) trouvés par l'algorithme génétique, avec différentes fonctions de distance : Aucune des distances de Hausdorff *(Θ), ne trouve la solution.*

erreur relative pourcentage (%)	$\mathbf{\Delta_H}$	$\mathbf{\Delta_A}$	$\mathbf{\Theta_H}$	$\mathbf{\Theta_A}$
Dégénérescence $(1-\tau_{DDC})$ (*100)	1-**11**	1-**19**	6-**22**	1-5
Paysage de distance $d(X_{N_c}, X^*)$ (*100/576)	0-1.5	1-**11**	2-**13***	7-**45***
Mutation δ_{Mut} $(*100/D)$	2	2	**20**	4
Croisement $(1-\tau_{PO})$ (*100)	1	2	**53**	8
D=diam(G)	50	50*576	2	2*576

*Table 11 : Récapitulatif des critères de comparaison des distances. Paysage de distance : min et max de 4 problèmes test de la distance phénotypique entre le meilleur trouvé par l'algorithme et la solution. Dégénérescence : min et max de 4 problèmes test de l'erreur de corrélation. Mutation : fluctuation moyenne (entre un individu et son muté) relative à l'échelle de la distance. Croisement: erreur de corrélation ($(1-\tau_{PO})$*100). Une * indique que le nombre de rectangles exact n'est pas trouvé. Les chiffres en gras indiquent des écarts importants.*

Outils de comparaison : Plusieurs points de vue peuvent être considérés (section III.5.5), essentiellement motivés par l'analogie avec les propriétés de Hamming en binaire. Dans la suite quelques commentaires pour chaque critère de comparaison, ainsi qu'un tableau récapitulatif (Fig. 11) qui donne des pourcentages d'erreur (écart aux résultats qu'on obtiendrait avec les mêmes mesures en binaire), relativement aux l'échelles respectives des mesures.

1. Paysage de distance : On part du principe qu'une fitness qui est simplement la distance à la solution, doit être facile à optimiser pour l'algorithme génétique. En binaire, par exemple, un simple algorithme de descente (local HC) résout une *fitness de Hamming* en $O(l \ln l)$ générations. Pour tester et comparer les distances sur l'espace des listes de longueur variable, on considère des *fitness de distance* [15] avec chaque distance génotypique, et on compare les résultats de l'algorithme génétique (qualité des solutions) sur ces fonctions.

Ce critère a permis, de distinguer concrètement les distances sur ces espaces. La figure 57 montre les résultats de l'algorithme génétique, sur une *fitness de distance* à un génotype de coin à 11 rectangles : La qualité des résultats varie sensiblement d'une distance à une autre. De toute évidence, la *fitness de Hausdorff* n'est pas facile à optimiser.

On peut faire un classement des fitness de distance, des plus faciles à résoudre aux plus difficiles. Globalement, les distances en échelles linéaire et exponentielle donnent des résultats semblables. Cependant, quand le nombre de rectangles est grand, les distances en échelle exponentielles donnent de meilleurs résultats. Un classement relativement à la qualité des résultats serait :

$$\Delta_H > \Delta_{HA} > \Delta_A > \Theta_H > \Theta_{HA} >> \Theta_A$$

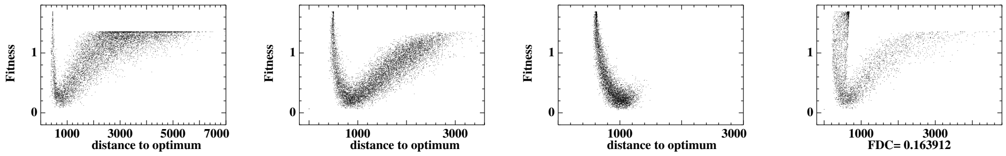
2. Corrélation : La corrélation entre les distances phénotypiques et génotypiques est calculée sur un échantillon aléatoire.
La distance de *Hausdorff* donne des graphes plus compacts que ceux obtenus, sur les mêmes échantillons, avec les distances par appariement. Les distances par appariement montrent plus de diversité sur l'axe des distances, donc permettent de mieux différencier les individus.

3. Mutations : Sur un échantillon aléatoire, on calcule la moyenne (et variance) de la distance entre un individu et son transformé par mutation élémentaire. Cela donnerait 1 en binaire avec la 1-bit mutation :
Si on raisonne de manière relative à l'échelle des distances (c'est en relatif que les GA perçoivent les changements), alors les distances de *Hausdorff* sont 20 à 50 fois plus sensibles aux effets des mutations élémentaires que les autres. Les paysages de *Hausdorff* ne respectent donc pas la propriété de "forte causalité" (une petite variation du phénotype implique une faible variation en fitness).

4. Croisements : En binaire, un enfant (o) est à mi-chemin entre ses parents ($p1$ et $p2$). Les quantités $d(p1, p2)$ et $d(p1, o) + d(p2, o)$ sont égales. Ainsi on mesure pour des couples pris dans un échantillon aléatoire : δ_{PO} différence moyenne, et τ_{PO} la corrélation statistique entre ces quantités. En binaire $\delta_{PO} = 0$ et $\tau_{PO} = 1$ pour tout opérateur de croisement :

Globalement les distances de *Hausdorff* donnent des résultats nettement moins bons que les distances par appariement (avec des corrélations de 0.5 par exemple). Ceci, à l'exception de Θ_{HA} en échelle linéaire, qui donne des corrélations au moins aussi bonnes que celles de Δ_H.

[15] $f(x) = diam(d) - d(x, x^*)$: la fitness est proportionnelle à la distance à la solution



FDC=0.72	FDC=49	FDC=-0.78	FDC=0.16
(a) 3 points	(b) 0.5-un point	(c) 0.25-un point	(d) γ uniform

FIG 69 : Graphes de corrélation avec la distance Δ_H^{φ} (variante de Δ_H) pour différentes initialisations sur le problème du damier, avec la représentation par triangles. Le choix des trois sommets uniformément dans la structure (a) donne beaucoup de triangles presque dégénérés, (limite supérieure du graphe).

Cependant, l'algorithme génétique est incapable d'optimiser la *fitness de distance* Θ_{HA}. Ceci montre que cette propriété d'exploitation des opérateurs de croisement (bonne corrélation entre parents et enfants), n'est pas forcément suffisante pour guider la recherche sur un paysage de fitness.

5. Dégénérescence : Pour étudier la stabilité au changement de génotype, on considère deux génotypes g_1 et g_2 (du même phénotype) et on calcule, sur un échantillon aléatoire $\{x_i\}$, la corrélation entre les distances $d(x_i, g_1)$ et $d(x_i, g_2)$. une corrélation de 1 signifie que ces distances sont proportionnelles.
Les distances par appariement montrent globalement une bonne robustesse à la dégénérescence (corrélation proche de 1), mais uniquement quand il s'agit de génotypes ayant des nombres de rectangles proches. Les distances de Hausdorff montrent des figures différentes, globalement plus compactes (faible diversité sur l'axe des distances).

3.3 ANALYSE DES PAYSAGES DE FITNESS

Choix d'une initialisation *a priori*

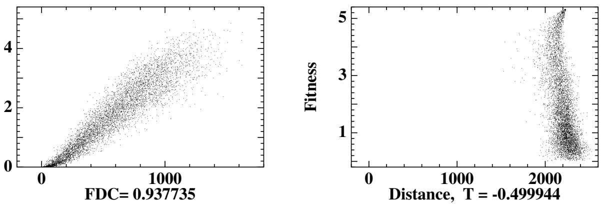
Sur ces représentations, les distances génotypiques permettent de caractériser quelques propriétés du paysage de fitness. L'un des aspects les plus frappants est celui de l'initialisation : Le graphe 'fitness $\times$ distance au but' permet de comparer les procédures d'initialisation, donc de régler leurs paramètres (nombre, centres, et tailles des rectangles par exemple). Ceci, d'autant plus que le réglage de ces paramètres interdépendants est assez délicat.

La figure 69 montre les graphes de corrélation (entre fitness et distance au meilleur individu dans un échantillon aléatoire) d'échantillons aléatoires obtenus avec diverses procédures d'initialisation pour la représentation par listes de triangles :

1. La **procédure 3 points** génère les trois sommets uniformément dans un domaine plus large que Ω.

2. La **procédure γ-un point** génère un sommet S uniformément dans Ω ; les autres sommets sont alors obtenus par une perturbation Gaussienne des coordonnées de S d'écart type γL_x, γL_y.

3. La **procédure γ-uniforme** est la même que **γ-un point**, avec la différence, que pour chaque individu, γ est d'abord choisie uniformément dans $[0, 0.8]$.



$1 - rect.$ $0 - rect.$

FIG 70: Graphes de corrélation (locale) : fitness (axe y) $\times$ distance à l'optimum (axe x), avec la distance Δ_H^{φ} (variante de Δ_H) pour le problème dissy*. La représentation 0-rect est clairement à éviter. Non seulement la corrélation est négative mais en plus, la diversité est faible.*

Choix d'une représentation *a priori*

La corrélation permet aussi d'orienter le choix de la matière qui correspond à la valeur par défaut, permettant ainsi de choisir entre les représentations 1-rectangle et 0-rectangle. L'expérience montre que la représentation 0-rectangle est à éviter quand il y a peu d'inclusions dans la solution. Or cette information ne peut être connue avant de résoudre le problème.

Dans le cas du problème "dissy" (cf. Figure 70), la différence de corrélation (entre fitness et distance au meilleur individu dans un échantillon aléatoire) permet de faire le bon choix : avec la représentation 0-rect, non seulement le taux de corrélation est négatif, mais en plus, la diversité est faible.

Cependant, s'il s'agit de comparer différentes représentations (Voronoï et rectangles, ou rectangles et triangles ...), la différence de corrélation ne peut suffire à expliquer la différence de performance des représentations sur un problème donné.

Le dernier recours pour choisir une représentation est de comparer leurs opérateurs respectifs. La méthode proposée dans la section II.6 permet en effet de comparer les opérateurs de croisement, mais aussi de manière analogue, les opérateurs de mutation. Ceci permet de choisir la représentation la mieux adaptée au problème, sans aucune connaissance *a priori* sur la forme ou la densité des inclusions dans la solution.

Summary of Results

Summary of Results

Evolutionary Algorithms (EAs) are a class of stochastic optimization/adaptation techniques that have been successfully applied in multiple areas such as machine learning, combinatorial problems, and numerical optimization. This success has initiated the development of different EA variants, and stimulated the theoretical research in order to describe and predict the behavior of EAs. It is a challenging task to develop a sound theory for EAs, as they use several probabilistic operators, such as mutation, recombination (also named crossover) and selection. The study of the properties of these operators for particular problems and algorithms is a first step to improve our understanding of the dynamics of EAs. One of the important issues is to compare EA convergence time (first hitting time of problem solution) when using different operators.

In that streamline, the first chapter is dedicated to the theoretical investigation of some mutation and crossover properties. Both first chapters are restricted to the binary search space, though some of the methods and conclusions are easy to extend to other search spaces. The first chapter compares two mutations in two different situations, and demonstrates longpaths existence for crossover operators. The second chapter is devoted to experimental comparison of the performance of EAs. It proposes a common basis to *a posteriori* compare EA results, and investigates the reliability of different statistical indicators of fitness landscapes to *a priori* predict EA performance. The last chapter proposes a case study of some variable length representations (unordered lists of real-valued *alleles*) that were designed for solving some difficult inverse problems of Structural Mechanics, the elastic inclusion identification.

Chapter I: EA convergence

1.4 Comparison of binary mutations

The first part (sections I.2 and I.3) of this chapter compares two types of binary mutation operators, namely the stochastic c/l mutation (all bits are flipped independently with the same probability c/l where l is the string length) and the deterministic 1-bit mutation that, for each bitstring, flips exactly 1 bit uniformly chosen. These operators yield the same average mutation rate for $c = 1$, but only the stochastic mutation allows simultaneous bit flips.
This study aims at comparing first hitting times of these mutations when applied in

- random walks directed by mutation (i.e. without selection), by considering a (1,1)-ES algorithm.
- two extreme cases of unimodal functions (respectively Onemax and *long k-path*), using the elitist (1+1)-ES.

The second part (section I.4) of the chapter addresses analogous issues for the crossover operator, and demonstrates the existence of controlled[16] paths for population based EA instances using crossover and mutation.

• **The first comparison case** presents a study of the each mutation in the absence of selection (i.e., inside a simple (1,1)-ES algorithm). Such algorithm can also be viewed as a random walk directed by the corresponding mutation. This should allow to compare both mutations without bias due to the interaction with the selection scheme.

[16]A fitness is a controlled path for a process if at each run, the process converges through a user-predefined path, in the sense that it takes no *shortcuts* ...

<u>PREVIOUS WORK:</u> Note that Analogous methods in the literature also avoid such a bias by considering the effects of each operator alone: crossover without selection or selection without operator (see the studies of time to convergence of crossover only algorithms [Ank91, LR93], *takeover time* of selection only algorithms [GD91, CDC96]), and *innovation time* –or time necessary to generate fitter individuals– of crossover or mutation in the absence of selection [TG93, Gol98])
Note also that the Markov chain corresponding to the deterministic *1-bit-flip* mutation, is known in the mathematical and physical literature under the term "Ehrenfest model" [Ios, Section 4.6]. One can find in [KS, Section 7.3] the computation of the expectation of the first hitting time when starting at distance $l/2$ from the target point, so that Lemma 3.6 of the expected convergence time for the deterministic mutation is not really new. Nevertheless, the intermediate results derived in its proof are used to exhibit the complete distribution of the hitting time. As discussed above, the complete distribution is necessary to solve the case of several independent processes.

<u>THE MAIN RESULT</u> in the absence of selection is that the expectation of the first hitting time for the stochastic mutation is larger by a factor $(1-e^{-c})^{-1}$ than that for the deterministic *1-bit-flip* mutation. Further, the hitting time of the $\frac{c}{l}$ mutation (resp. *1-bit-flip* mutation) obeys an exponential distribution with mean $\frac{2^l}{(1-e^{-c})}$ (resp. 2^l).
A direct consequence is that the first hitting time of N independent $\frac{c}{l}$ mutations (resp. *1-bit-flip* mutations) obeys an exponential distribution with mean $\frac{2^l}{N(1-e^{-c})}$ (resp. $\frac{2^l}{N}$). Hence, there is no gain in number of mutation applications.
Another consequence that follows from the exponential law properties, is that the deterministic mutation is more efficient than the stochastic mutation in the short-term: The probability that convergence occurs before time t is higher for the deterministic mutation than for the stochastic mutation.
Finally, if we compare the sequences of visited points by both mutations, we see that these sequences coincide at instants that follow a Poisson distribution, in the sense that the $\frac{c}{l}$ mutation jumps over some points visited by the *1-bit-flip* mutation. Hence, when the *1-bit-flip* process reaches the target point for the first time, the $\frac{c}{l}$ mutation process misses it with probability $1-\frac{1-e^{-c}}{c}$.
Note that this latter result is interesting by itself since it gives a way to compare the trajectories of the processes. Further, it allows a better understanding of the difference in expected convergence time with both mutations.

Remark: *Another point to consider, when trying to analyze these theoretical results in the light of practical work, is that for large values of l, a fraction e^{-c} of the stochastic mutations do not modify the bitstring at all (about one third for $c = 1$): Considering only active mutation steps, the laws of the hitting times with either mutation are identical in the asymptotical framework.*

• **The second comparison case** involves the coupling of the operators with deterministic selection, the (1+1)-ES algorithm, for the OneMax problem: a single parent generates a single offspring, and the fittest (i.e, closest in Hamming distance to all ones string) of both becomes the next parent.

<u>PREVIOUS WORK</u> prove that complexity of the (1+1)-ES on the Onemax problem for both the deterministic and stochastic mutations is $O(l\log(l))$. However, these works cannot be used to actually compare the $\frac{c}{l}$ and the *1-bit-flip* mutations operators, as they provide order of magnitude results without expliciting the multiplicative constants.

<u>MAIN RESULTS:</u> We significantly improve the above cited results by providing the expansion of the Onemax hitting times up to terms of order $o(l)$. This expansion shows that the dominant term of the hitting time of the $\frac{c}{l}$ mutation (resp. the *1-bit-flip* mutation), is deterministic and is equal to $\frac{e^c}{c} l \ln l$ (resp. $l \ln l$). Further, the normalized variance (that is to say the variance divided by the mean) is of the order of $(\ln l)^{-1}$ for both mutations.
We then give an expansion of the first hitting time T_N of N independent processes with the $\frac{c}{l}$ mutations (resp. $\bar{T}_N$ for the *1-bit-flip* mutation), and show that there is very little gain when using several independent processes: In fact, the gain amount is of the order of l, so that the dominant term of the hitting time expansion of N processes is the same as that of one process, ie. $\frac{e^c}{c} l \ln l$ (resp. $l \ln l$). This is due to the low variances of the distributions of T and $\bar{T}$ ($(\ln l)^{-1}$ goes to zero when lgoes to infinity). However that this convergence rate is rather slow for practical l values. The normalized variance is actually of order 1 for a large band of values of l. In order to get $\bar{V}_\mu < 20\%$ (resp. $\bar{V}_\mu < 10\%$), it is necessary that $l > 7.10^2$ (resp. $l > 4.10^5$).

Remark: *As recalled in the previous remark, the* 1-bit-flip *and the* $\frac{c}{l}$ *mutations require the same effort to reach their goal in the absence of selection. This is not the case here in the presence of selection: Convergence time of the stochastic* **active mutation** *is still a factor* $\frac{e^c - 1}{c}$ *greater than the deterministic mutation in the case of the (1+1)-ES on the OneMax problem.*

$\longrightarrow$ <u>TO SUM UP</u>, a first outcome of the results of this study is a clear advantage of the *1-bit-flip* mutation over the $\frac{c}{l}$ mutation, both in the absence of selection, and when using the (1+1)-ES selection on the OneMax problem. In the Onemax case, the result holds even if we restrict to comparing the number of active mutations (that flip at least one bit of a string). Further, when considering N independent processes, there is no gain in number of mutation steps in the absence of selection, and very little gain in the Onemax case.
However, we don't claim that the deterministic *1-bit-flip* mutation should be preferred to the stochastic mutation, as it is clear that the *1-bit-flip* mutation with non site wise optimizable (SWO) functions cannot avoid any local optimum. Rather, we point out the fact that a balance should be found between the ability of a mutation to escape from local optima, and its efficiency in terms of convergence speed. The results of this study suggest that positive[17] mutations can be outperformed by irreducible[18] mutations such as the deterministic *1-bit-flip* mutation.

On the other hand, the results obtained here are strongly related to this lack of deception and interaction of the Onemax problem. One can wonder whether such results may hold for more challenging problems? Note that for the *long k-path*, the results are the opposite of those of the Onemax case: Rudolph [Rud96] proves that the (1+1)-ES with stochastic $\frac{1}{l}$ mutation is a lot quicker (see next section) than the deterministic *1-bit-flip* mutation.

1.5 Long k-path problems

Last section gives exact hitting time for the stochastic mutation on the straightforward unimodal onemax problem. Still within unimodal[19] functions, we now consider the *long k-path* problem. Compared to the Onemax problem, the *long k-path* probably situates at the other end of the class of unimodal fitness functions. This study should allow to put some

[17] A positive mutation links any two points of the space with non-zero probability
[18] There exists a finite chain of mutations linking any two points of the space with non-zero probability.
[19] No local optima but the global one.

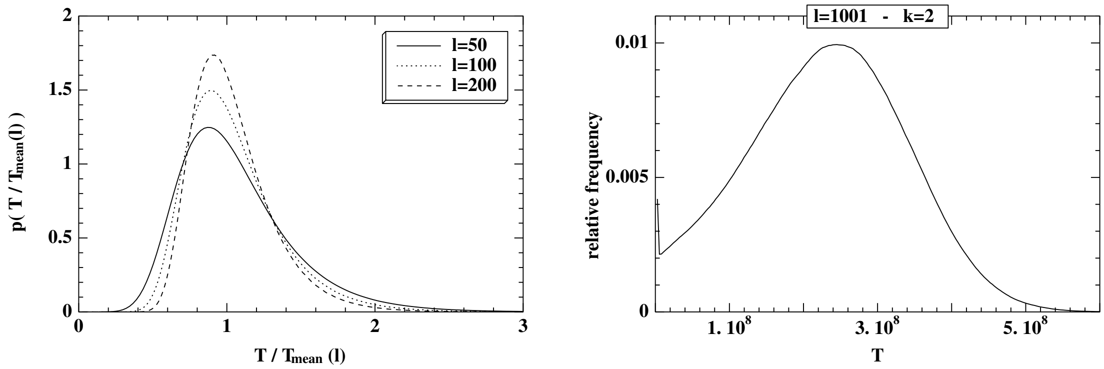


FIG 5 : *Theoretical histograms of (1+1)-ES first hitting time (normalized to its mean) using the stochastic* $1/l$ *mutation: In the Onemax case (left picture), note that the histogram is asymmetric. In the* Long k-path *case (right picture), note the anomalous pick close to zero.*

theoretical bounds on the efficiency of the stochastic mutation within the class of unimodal of fitness functions.

An extensive study of *long k-path* problems is proposed: we first extend Rudolph results (which gives the order of the expectation of hitting time) by giving the expected hitting time and its variance in the case of fixed k. Finally, we investigate the unsolved case of variable $k = l^{\alpha}$ and address the challenging issue of the existence of longpaths[20] for the (1+1)-ES using the $\frac{c}{l}$ mutation.

Remark: *The case of the (1+1)-ES using deterministic* 1-bit-flip *mutation is not addressed here. In fact, by construction of the* (k,l)*-path, this process visits every string of this path before it ends at the global optimum. Hence, convergence time is equal to the path length, which is exponential*[21] *in problem size* l.

<u>PREVIOUS WORK:</u> Rudolph [Rud94b] proves $O(l^{k+1}/k)$ as an asymptotic upper bound for the expected time spent by a (1+1)-ES using the stochastic $1/l$ mutation rate on a long k-path. For a fixed value of k, this bound is polynomial in terms of string length. This implies that for large values of l the (1+1)-ES with $1/l$ mutation is a lot quicker than the 1-bit-mutation Hill Climber in optimizing the path. The case of variable k value has been left unsolved, leaving the the existence of longpaths [22] for the stochastic mutation an open question.

• **In the case of fixed k,** <u>THE MAIN RESULT</u> claims that $\mathbb{E}[T]$ the expected first hitting time is $el^{k+1}/k\ln(l/k)$. This result is not an estimate but the dominant term of the expansion of $\mathbb{E}[T]$ with respect to l. Further, we show that although the normalized variance of T goes to 0 as $l \to \infty$, the decay rate is so slow that the normalized variance actually remains of order 1 for a large band of values of l.

We also give the dominant term of the expansion of complete probability density of T. As shown in Figure 5, the histogram of T shows an anomalous peak close to zero, corresponding to the event *shortcut* $\mathcal{S}_1$[23] *happens first* , ie. before the other types of shortcuts. The probability of this event is kl^{-1}. It yields a convergence time, a factor of l^2 smaller than the expected

[20] A fitness is a longpath for some process if: convergence time is exponential or super-polynomial in problem size

[21] Unless the growth rate is explicitly stated, any asymptotic behavior beyond a polynomial growth in string length will be called *exponential.*

[22] A fitness function is a *longpath* for a given iterative search procedure when each convergence time of each run of the procedure is at least an exponential of the problem size. Moreover, we exclude the procedure from spending an exponential time trying to escape from a local optimum.

[23] flips the first k bits of a string

value $\mathbb{E}[T]$.
This shows that with probability kl^{-1}, the process hits the final string a factor of l^2 quicker than the expected value of T. Therefore, the best strategy to take advantage of this distribution, is to perform $O(l)$ independent (1+1)-ES processes, so that one of these processes is very likely to realize the exceptional event. On the other hand, population based processes limit diversity in shortcuts occurrence, drifting all individuals to blindly follow the first occurring shortcut. Hence, a $(\mu + \lambda)$-ES strategy for example yields the same number of fitness evaluations as the (1+1)-ES on *long k-path* problems.
Finally, the expected convergence time distribution is given for any mutation rate d/l, $d > 0$. As expected, the smallest convergence time is obtained with a k/l mutation probability.

We now turn to the case of variable k values, yet unsolved. Recall that the case of $k = O(l)$ is not interesting because it yields a path length of $O(l)$ and an expected number of trials of no more than $O(l^2)$.

• **In the case of $k = l^\alpha$,** MAIN RESULT: The expected convergence time – yet completely unknown– with the stochastic $\frac{c}{l}$ mutation is exponential. Besides, shortcuts speed up the convergence only if $\alpha < 1/2$. Otherwise (if $\alpha \geq 1/2$), the process reaches the solution by following the whole path, taking no shortcuts.

$\longrightarrow$ A number of interesting issues are detailed below:

- The difference of deterministic and stochastic mutations efficiency on *long k-path* problems implicitly illustrates a key phenomenon that is also suggested in the second chapter: $\rightarrow$ Long time behavior of Markov chains on such big search spaces, is not reflected by typical short time behavior (for example, expected behavior of both mutations), but is due to the deviant atypical convergence behavior. The study in the second chapter of –the reliability of– landscape and operator indicators (defined as averages), obviously leads to the same conclusion. A direct consequence is that the fitness landscape of EA operators is related in a non trivial manner to the exploration mechanism.

- The singularity in ES behavior around $\alpha = 0.5$ can be explained in an intuitive manner: When α increases beyond 0.5, the shortcuts require a simultaneous flip of a too big number of bits (l^{l^α}); hence shortcuts are no longer taken by the process (this is to combine with the fact that path length decreases when $k = l^\alpha$ increases).

- The existence of longpaths for a (1+1)-ES using the stochastic mutation is proved in the case $k = l^\alpha$ and $\alpha \geq 1/2$, the process follows the path (in the sense that, with probability 1, it takes no shortcuts), which implies that convergence time is exponential. This situation is somehow counter-intuitive, as in theory this mutation is a positive[24].

- Independent parallel (1+1)-ES processes perform a lot better with a null migration rate than they perform when there is migration (of fit individuals) between the processes. Increasing the population size also slows down the convergence speed.

1.6 Controlled longpaths for EAs

A natural direction for the last part of the chapter is the extension of the above result (existence of longpaths for mutation Hill Climbers) for other types of algorithms, for example for population based algorithms involving crossover operators.

[24] In the sense that, in one transition, it can reach any point of the space with a non null probability

However, before investigating population based algorithms, the notion of "following a path" has to be extended for populations:

<u>DEFINING CONTROLLED PATHS FOR POPULATIONS:</u> A path of an Evolutionary Algorithm (EA), for example, can be represented as a sequence of populations. However, by definition, EAs are stochastic population-based algorithms. It makes little sense to exactly control the composition of the successive populations. We then propose the following definition of the term "follow a path". An EA is said to follow a *controlled path* when the best individual of the population visits a number of predefined *areas* in the search space in a predefined order during the course of each run.

<u>PREVIOUS WORK:</u> There is no previous work in the context of populations, but only for simple algorithms: As explained in section I.3, the *long k-path* problem has been constructed in previous work to prove the existence of longpaths for the deterministic *1-bit-flip* mutation (1+1)-ES. Section I.3 extends this result for the stochastic mutation in the case of variable $k = l^\alpha$ with $\alpha > 0.5$.

<u>MAIN RESULT:</u> We propose a generic way for constructing controlled paths for EAs. EA populations are lead through a number of consecutive hyperplanes of order $l/2$ of the search space. While we have no formal proof yet of this controlled path being a possible longpath for population based EAs, experiments with uniformly chosen hyperplane keys show no shortcuts to fitter areas. This empirical observation holds for different EA instances using one point or uniform crossover and deterministic or stochastic $1/l$ mutation. Moreover, we prove that the controlled path with keys from the *long k-path*, defines a longpath for the (1+1)-ES, in the sense that convergence time is exponential and that the solution is reached before any shortcut (to fitter areas) happens.

To construct a controlled path, the main idea is to force the populations to go successively towards a number of predefined areas, by using a sequence of modified Onemax functions. An area typically is a hyperplane determining one half of the string. For example the first area is made of strings $s = s_1..s_l$ of $\{0,1\}^l$ such that $s_1..s_{l/2} = key1$, where key_1 is a (user defined) fixed string of length $l/2$. The keys are chosen far enough from each other (uniformly in $\{0,1\}^{l/2}$ so that the expected distance between a pair of key is $l/4$), in order to avoid simultaneous bit flipping shortcuts.

$\longrightarrow$ Controlled paths provide an example of smooth and unimodal[25] fitness landscapes, on which EA trajectory can be controlled (empirically for commonly used EA instances). An interesting but currently unanswered question is whether <u>real world problems</u> can present such a longpath structure? The analogy with spin glasses shows that there exist physical systems which evolve very slowly towards an equilibrium state, through successive unstable attractors.

$\longrightarrow$ Finally, while controlled paths improve our practical knowledge about possible EA dynamics, it also appeals for the construction of EA instances (choice of operators and selection schemes) that don't follow the controlled path functions proposed here, and this without any *a priori* knowledge on hyperplanes structure.
Note finally that selection schemes that maintain diversity in the population (e.g. crowding[DeJ75], sharing [GR87]) coupled with adaptive operators may help to design more powerful EAs. This shall be left for future work: "Can we break the inherent symmetry of a Controlled Path problem without *a priori* knowledge on its structure".

[25] functions with no local optimum but the global one

Chapter II: Indicators of the performance of EAs

One of the major challenges in the EC field is to characterize which fitness functions are easy for a given instance of EAs and which are not. Previous work aim at finding statistical indicators of EA performance, from the study of fitness landscapes. The ultimate goal of such approaches is to predict EA performance on a given landscape.

PREVIOUS WORK: A number of statistical indicators of fitnesses (resp. operators) landscapes –originated from empirical observations–, have been proposed to characterize classes of fitness functions with similar GA behavior. We mention fitness distance correlation [JF95], epistasis [Dav91a], correlation length (resp. correlation [MdWS91], [FG96] and fitness improvement for operators).
Studies of the above indicators have mostly been limited to the construction of examples and counterexamples ([RVSK97, Alt97, QVSG98]). An interesting direction, scarcely taken, is to construct and study classes of fitness functions which share the same properties (for example functions with the same indicator value) in order to draw a precise picture of the reliability of these indicators (in which cases they may be most reliable).

This meets the main motivations for this chapter: Sections II.3, II.4, and II.5 investigate some static fitness landscapes indicators, as Fitness Distance Correlation (FDC), epistasis variance (Epiv), fitness improvement or correlation for crossover operators ... Particular attention is given to Fitness-Distance relations, and FD plots.

Nevertheless, before being able to *a priori* predict EA performance, one should be able to *a posteriori* compare algorithms' behaviors (to compare the behavior of algorithm A on problem 1 with the behavior of algorithm B on problem 2).

Hence the first part of this chapter (section II.2) addresses the challenging problem of comparing the behaviors of two different algorithms when applied respectively to two different fitness functions. The aim of this part is not properly to define an order relation in the set of possible EA behaviors, but simply to extract *scalable* information that may be used later on (in a manner dependent on the user's prerequisites) to construct a total or partial EA-performance relation that suites a particular problem domain.

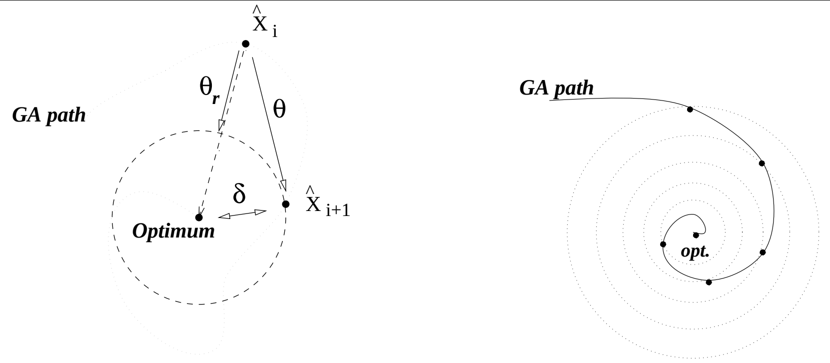
II.2 Method to describe and compare EA behaviors

We first define a number of *behavioral measures* (independent from fitness distribution) to characterize EA behavior and results on a given fitness function: In addition to convergence time, Figure II.2 illustrates some geometrical quantities derived from the trajectory of the algorithm. It is not easy to compare two sets of such behavioral measures, as convergence time is very sensitive to initialization bias (initial population with close to optimal individuals), and trajectory measures are directly dependent on the choice of the genotypic distance.

Since EA behavior on *distance fitness* functions (the fitness is the distance [26] to optimum), is well understood, we propose to use it as an absolute basis to *scale* the behavioral measures, following the question: *What would this same algorithm have done –starting from the same population– if for each x the fitness function were simply the distance to optimum of f (or to the best individual found by the algorithm).*

More precisely, the comparison method consists in performing the following steps for fitness

[26] We suppose that the genotypic distance is implicitly *related* to EA operators, as for example the *Hamming* distance is related to the *1-bit-flip* mutation operator. The relevance of a distance in relation to genetic operators is addressed in the third chapter.



(a) (b) Minimal θ_r

FIG 2: **(a) Behavioral measures** *of EA trajectory:*
Trajectory distance θ, *sums the distance between best individuals at successive generations,*
Radial trajectory θ_r, *sums the difference in distance to optimum of fittest individuals ...*
Distance to optimum δ, *is the expected distance to optimum.*
Naturally, if the optimum is not found the behavioral measures are computed with respect to the best found by the EA.
(b) *Typical GA path with a* Hamming *distance fitness:* θ *is minimal, not* θ_r.

f (resp. g), then in comparing $\rho_m(f)$ and $\rho_m(g)$, where m is a behavioral measure of the algorithm as illustrated in Figure II.2:

1. `Run EA with fitness` $f \longrightarrow$ `compute` $m(f)$

   Note s be the best found by the EA.

2. `Run the same`[27] `EA with fitness` $\mathcal{H}_s$[28] $\longrightarrow$ `compute` $m(\mathcal{H}_s)$

3. `Compute ratios` $\rho_m = m(f)/m(\mathcal{H}_s)$

$\longrightarrow$ The *Hamming fitness* are proposed as an absolute basis to scale EA behavioral measures (convergence time and trajectory measures), and to take into account eventual parameters biases (such as over-fitting initialization that generates close to optimal individuals). Note that the method can be used

1. to compare different representations, simply by replacing the *Hamming* distance with the corresponding genotypic distance (see chapter 3 for the choice of the distance). Therefore it can be used to compare landscapes obtained with different representations for the same fitness function.

2. to identify functions with EA-like behavior. This can be achieved by considering a sequence of selected basis functions: The systematic comparison of EA results to the results it would have if the fitness were an NK-fitness (for different K values) for example, may lead to a partial classification of some typical EA-similar-behavior fitness functions.

3. to guide the choice of the better strategies (representation/operators choice) to improve the optimization task, by detecting similarity –in terms of EA behavior– between the fitness function of the problem at hand and a family of well-known reference functions. This issue is detailed in the section of future work.

[27] EA with the same initial population and parameters as for f in step 1
[28] $\mathcal{H}_s(x) = l - d(x, s)$ is the Hamming fitness to s

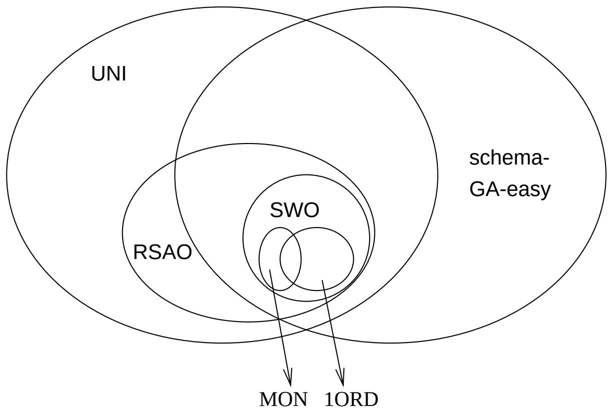


FIG 3: Pictorial description of the relationship between four classes related to GA-easiness in the case of functions with unique global optimum.
1ORD: *The fitness is the sum of the contributions of each bit, with no interaction terms (Epiv=0).*
Linear: *($\subset$ 1ORD) non constant linear funcitons w.r.t. to the distance to optimum (FDC=-1).*
MON: *Monotone functions with respect to the distance to optimum.*
SWO: *Site wise optimizable functions are slightly more general than MON functions (swo=0).*
RSAO: *functions optimizable by a 1-bit-flip mutation steepest ascent in less than l steps.*
Unimodal: *functions with no local optimum but the global one.*
Schema easy: *the average fitness of any hyperplane containing the optimum, is better than the average fitness of hyperplanes of the same order that doesn't contain the optimum.*

Note however that this method is still not completely scale free. For instance the ratios of behavioral measures still heavily depend on the genotypic distance scale. Hence its extention to comparing different representations still requires some more work in order to find more robust statistical comparison tools than simple ratios.

II.3 Statistical indicators of landscapes

Having precised how EA behavior and performance can be described, we now investigate some *a priori* predictors of EA performance. For the sake of clarity, this section and the next ones use a very rough notion of EA hardness, mainly defined as non-EA-easiness. EA-easiniess means that convergence time on fitness f has the same order than on a *Hamming* fitness to the optimum of f, hence meeting the guidelines of last section.

MAIN RESULTS: This section reviews, further investigates and extends some statistical indicators like epistasis variance[29] and Fitness Distance Correlation (FDC [30]) proposed in the literature. First, the reference classes (the class the indicator measures a distance or a difference from) of the indicators are identified and compared. These reference classes are respectively Linear functions for FDC, First Order (1ORD) functions for epistasis variance, and site wise Optimizable (SWO) functions for the new swo indicator. The respective definitions are summarized in Figure II.3.

The swo indicator generalizes both epistasis and FDC indicators, and overcomes their

[29] Epistasis variance computes the distance between a function and its projection on the class of first order functions, hence it computes the distance to the class of 1ORD functions.

[30] The statistical correlation between the fitness and the distance to optimum of points of the search space.

sensitivity to fitness scaling, as well as the constantness problem of epistasis: Epistasis variance reports a 0 value for both linear and constant functions (no interaction between the bits). Hence, by construction, epistasis variance cannot make the difference between easy (linear) functions and difficult (constant) functions for EAs.

We then construct a class of parametrized site wise optimisable fitness functions, for which the FDC and Epiv indicators take any arbitrary value, blinded by a large proportion of completely irrelevant information present in the random sample. Note that the reliability of an indicator is necessarily limited as it is a statistical summary ; However, we may notice that FDC and Epiv cannot characterize a host of functions straitforward for EAs among the SWO class.

A final remark is that it is not difficult to find functions that completely blind the swo indicator: swo also takes any arbitrary value when presented with parametrized fitness functions from the class of Nearest Neighbors functions[31] (NNI), such as $f(s) = \sum_i \alpha_i \ (1 - s_i)s_{i+1}$. These functions have adjacent second order interactions only and they range from easy to difficult for EAs, according to the α_i values.

$\longrightarrow$ As a conclusion, all the average indicators of fitness landscapes studied here are easily misleaded by both *ad hoc* and real world function cases. This appeals for future design of alternative indicators, more robust, and eventually based on more relevant samples (as those obtained from EA populations or metropolis sampling ...).

II.4 Fixed Fitness Distance relations

Of course, the whole Fitness Distance plot (FD plot) gives much more information than the summary correlation value (FDC). This section invetigates the sensitivity of EA behavior to strings distribution inside the FD plot by studying particular fixed FD relations: We build different functions which share the same longpath-shaped FD relation, then we investigate EA behavior on these functions.

Main Results: Different instances of GA and ES algorithms show a convergence behavior very sensitive to the invisible strings distribution inside the FD plot. For example, a **small translation** of the FD relation along the distance axis, completely changes EA behavior. Further, a **small perturbation of the fitness values** can significantly increase convergence speed. On the other hand FD plots that cover the entire fitness–distance space, may contain easy paths to the optimum, eventually resulting in a OneMax like behavior.

The main conclusion is that interpretation of the FD plot can be dangerous, since a local optimum in the fitness landscape is scarcely visible in the FD plot. Further, two neighboring points in the FD plot are not necessarily close in Hamming distance to each other. We give examples where the stochastic $1/l$ mutation cannot escape (in reasonable time) from a point situated on a locally linear relation between fitness and distance to optimum (the upper line in Figure II.4).

II.5 Choice of Initialization procedures

Last section shows that interpreting the FD plot in order to predict EA behavior can be very misleading, as various types of functions (easy or difficult for EAs) can share the same FD relation.

[31] The origin of these functions lies in statistical physics. The functions correspond to the instances of the anti-ferromagnetic one-dimensional nearest neighbor interaction model.

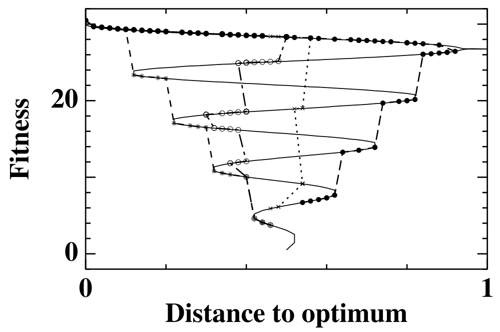


FIG 3: Different EA trajectories (all dotted lines) visualised in the Fitness Distance plot (the plain line shows the FD relation): According to strings distribution in the plot, convergence time (normalized to problem size l) varies from 2 to 1000, and some fitness cases are intractable for GA instances using stochastic $1/l$ *mutation and crossover.*

On the other hand, if we consider a fixed fitness function, the *fitness* $\times$ *distance to optimum* plot may be useful to tune some parameters of EAs:

MAIN RESULTS A robust heuristic for choosing among different initialization procedures, consists in selecting the procedure that yields a good dispersion of the two-dimensional distribution in the space "fitness×distance to optimum" of random points generated with the procedure. Besides, since the optimum is scarcely known (!), the best individual found in the random sample can be used to compute the distances. This practical heuristic is validated for a number of *ad hoc* test functions, and also used successfully for a real world problem in the third chapter: FD plots guide the tunning of the interdependent parameters of initialization procedures for degenerate variable length representations.

II.6 Statistical indicators for evolution operators

Having investigated the limitations (II.3) and the benefits (II.5) of the landscape based indicators, we turn to study some operator based indicators proposed in the literature.

As with landscape indicators, operator indicators are usually computed from a random sample of the search space: Couples of parents are selected in a random sample and mated. This allows to compute operators indicators like the expected improvement of offspring fitness over parents fitness (resp. over some target fitness), and the statistical correlation between the mean (resp. best) fitness of a couple of parents and the mean (resp. best) fitness of their offspring ...

MAIN RESULTS: A thorough comparison of these indicators on various test functions (including linear, deceptive and difficult functions) suggest that EA-behavior is not reflected by average typical properties of operators. Rather, the atypical and rare events such as operators potential to generate –even scarcely– a high fit offspring, accurately reflect actual EA-behavior. The same phenomenon is implicitely illustrated in the first chapter where two mutations that have the same expected behavior, yield a big difference in convergence time on different problems.

In light of these results, the *offspring ranking* method is suggested to compare genetic operators. This method however, has been experimented on random sampling which limits its efficiency to the first generations of the run. Further work may address more representative sampling schemes, for example derived from EA populations.

Note that applications of this heuristic go beyond the choice of operators for a given fitness landscape. Indeed, it also fits to compare representations, by comparing their respective operators (see application to a real world problem in the third chapter).

Chapter III: Variable Length Representations

This chapter proposes a case study of some variable length representations (unordered lists of real-valued *alleles*) that were designed to solve some difficult inverse problems of Structural Mechanics, the elastic inclusion identification [Sch95, Sch96]. The phenotype space is a subspace of the space of two-dimensional "black-and-white" shapes (i.e., partitions of some given initial domain in $\mathbb{R}^2$ into two subsets with continuous boundaries).
Section III.2 introduces the mechanical problem of inclusion identification in Structural Mechanics. The variable-length representations for two-dimensional partitions are introduced in Section III.3, then compared in Section III.4 on some instances of the inclusion problem. This points out the limits of the bitarray representation, which is largely outperformed by the variable-length representations.

The aim of the last sections of the chapter is to design and use distances on the genotype space (Section III.5), allowing a further analysis of the fitness landscape, tuning of algorithm's parameters (choice of representation and initialization) using some results of chapter II (Section III.6).

III.2 An inverse problem

A structure (the *design domain*) is made of two different known materials, but their respective repartition in the structure is unknown. The available data consist in experimental records of the displacements of some points on the structure under known loadings (point-wise forces). The goal is to identify the geometrical repartition of both materials from these experimental data.

For a given repartition of both materials, the computation of the simulated mechanical behavior of the structure is straightforward, using any Finite Element Analysis (FEM) package. The identification can then be viewed as an inverse problem.

Note that in structural mechanics, this is a difficult problem mathematically ill-posed and resisting up-to-date numerical methods [LRN93, Con94]. Yet, in steel manufacturing plants for example, it is of vital importance to check if coal scoriae are included in steel parts, and if their repartition does not dangerously weaken the whole part.

III.3 Representations for 2D partitions

A straightforward representation from the GA point of view is an array of bits (termed *bitarray*), based on a regular discretization (*mesh*) of the design domain into $n \times n$ *elements* (see also [Jen92, CSJ94]). Each bit corresponds to one element, and indicates to which of the two subsets this element belongs.
Nevertheless, the bit-array representation encounters very severe restrictions when refining the mesh, because bitarray complexity (n^2) is dictated by size of the mesh.

Therefore two other representations have been designed in order to separate the complexity of the representation from the accuracy of the fitness evaluation (dictated by the size of the mesh): The *Holes* and *Voronoï* representations exhibiting fairly high degeneracy and epistasis. In both cases, an individual X of the seach space $\mathcal{G}$ is variable-length unordered list of alleles:

Holes representation: An individual (point of the search space) is a list of holes $(H_1, H_2, \dots, H_l)$, $l \leq L_{max}$, of elementary shape (rectangles or triangles here) with variable dimensions. The design domain receives a default value (e.g. subset 0), and the list of "holes" describes the topology of the other subset (e.g. subset 1).

Voronoï Representation: An individual is a list of Voronoï sites (points of the design domain Ω) $(V_0, \dots, V_l)$,, $l \leq L_{max}$ in which each site is labeled 0 or 1. To each site V_i is associated the set of all points of the design domain for which the closest Voronoï site is V_i, termed *Voronoï cell.* The *Voronoï diagram* is the partition of the design domain defined by the Voronoï cells.

Crossover operator: Based on a geometrical choice, the crossover operator exchanges the holes (resp. Voronoï sites) whose centers are on one side of a random line drawn across the design domain.

Mutation operators: Two types of mutation are proposed:

- A Gaussian mutation on the – real-valued – parameters of a randomly chosen hole (resp. Voronoï site), such as center/site coordinates, or hole size.
- Addition and destruction of some holes (resp. Voronoï sites).

III.4 Numerical results

All numerical results for the problem at hand are obtained on a two-dimensional square structure held fixed on its left-side, and normal point-wise forces are applied at given points of the three other boundaries.
The aim is to identify the repartition of two materials into the square: a hard material ($E = 1.5$ and $\nu = 0.3$) and a soft material ($E = 1$ and $\nu = 0.3$). A fixed mesh of size 24×24 will be used used throughout the experiments, unless otherwise mentioned.
The fitness is then the average of the error over all loading cases. Needless to say, the total computational cost increases with the number of loading cases N:

$$f(x) = \frac{1}{N} \sum_i^N |d_i(x) - d_i(x^*)|,$$

where x is a two dimensional partition of Ω, $d_i(x)$ is the displacement at the i^{th} measurement point of structure x and $d_i(x*)$ is the displacement at the i^{th} measurement point of the optimal structure.
The evolutionary method demonstrates powerful on the simplified problem of linear elasticity involving two materials of known characteristics, widely outperforming the to-date results of specialized numerical methods.
However, the bitarray representation of structures is large;y outperformed by the variable length representations.

III.5 Genotypic distances

In order to get a better understanding of the dynamics of EAs with these representations, and in order to use the results of chapter II and guide the choices when tuning the parameters of the algorithm, some tools have to be designed, the first of which are some genotypic distances. For instance, most methods to study the behavior of evolutionary algorithms on the fitness landscape rely on a metric structure of the genotype space, as illustrated in the second chapter. Note also that the different niching mechanisms (e.g. crowding [DeJ75], sharing [GR87]) aim at controlling the diversity of the population in the search space, and hence rely upon the notion of genotypic proximity.

We now propose a general method to design genotypic distances on search spaces of variable length unordered lists of alleles, based on a similarity measure on the allele space. However, different choices of allele similarity can lead to different distances, and raise the issue of the choice among distances on the same search space. This leads to design some qualifying tools for genotypic distances, related to their behavior with respect to either the evolution operators and EA dynamics, or the degeneracy of the representation. These tools are motivated from the analogy with the binary *hamming* distance, and are fitness independent.

PREVIOUS WORK: Previous work suggests different ways for constructing distances related to the evolution operators. Genotypic distances can be defined in relation to the probability of the most likely sequence of operators that transforms an individual into another, or in relation to the minimal number of operator applications that link two points of the search space. However, in the case of variable length representations, the evolution operators yield a non-bounded neighborhood, and it is not possible to compute such distances (some operators link any two points of the space with non-null probability).

Matching based distances in $\mathcal{G}$: Let $\mathcal{G}$ be the space of unordered lists of *alleles*, all alleles taken from a space $\mathcal{A}$, and let δ be a distance over the space $\mathcal{A}$.
If δ is a bounded distance in $\mathcal{A}$, then

$$\Delta(x,x') = \min_{\mu \in M(x,x')} \big(\sum_{1}^{n'} \delta(a_i, a'_{\mu(i)})\big) + (n'-n) diam_\delta(\mathcal{A}) \tag{4}$$

is a distance in $\mathcal{G}$, where $x = (a_i)_{i\in[1,n]}$ and $x' = (a'_i)_{i\in[1,n']}$ are two individuals of $\mathcal{G}$ such that $n \geq n'$. Further, $M(x,x')$ denotes the set of all matchings between x and x' between x and x' (a matching being an injective application from $[1,n]$ in $[1,n']$).

δ distances in $\mathcal{A}$: Given a distance d in $\mathcal{A}$, we construct a class of distances in $\mathcal{A}$, of which d is a particular case. Next section proposes a comparison between the performances of these distances with respect to evolution.

The above table gives three possible expressions for δ, in exponential scales. Replacing the term $\exp(-d(a,b))$ by $(1-d(a,b))$ yields three other expressions for δ in linear scale. In the following, A_a denotes the area of hole (resp. Voronoï cell) a.

Distance in $\mathcal{G}$ *Holes representation* *(resp. Voronoï representation)*	**corresponding distance in $\mathcal{A}$** *(in exponential scale)*
Hole based (resp. Site based)	$1 - e^{-d(a,b)}$
Area based (resp. Cell based)	$\frac{A_a + A_b}{2} - min(A_a, A_b) e^{-d(a,b)}$
Hole/Area based (resp. Cell/Site based)	$1 - \frac{2\, min(A_a, A_b)}{A_a + A_b} e^{-d(a,b)}$

Analogy with other distances: A distance in $\mathcal{G}$ (variable length unordered lists of alleles) is the sum of the distances between matched alleles. Analogously,

- the *Hamming* distance in $\{0,1\}^l$ (viewed as ordered lists of alleles) is the sum of distances between alleles (bits).

- The minimal number of elementary[32] mutations that link two individuals, is very close to the definition of the holes based (resp. Site based) Δ distance, in the context of fixed length integer valued genotypes.

- the ***Hausdorff*** distance in $\mathcal{G}$ is also based on the choice of a distance in A, but it requires no allele-matching: *Given d a distance in E, let $d(a,B) = min\{d(a,b), b \in B\}$ where $a \in E$, $B \subset E^n$. The* Hausdorff *distance between two subsets of E, $A \subset E^{n'}$, $B \subset E^n$ is given by,*

  $\Theta(\mathbf{A},\mathbf{B}) = \mathbf{max}\{\mathbf{d}(\mathbf{a},\mathbf{B}) : \mathbf{a} \in \mathbf{A}\} + \mathbf{max}\{\mathbf{d}(\mathbf{b},\mathbf{A}) : \mathbf{b} \in \mathbf{B}\}$

θ is of course a distance between sets, and not lists. Voronoï diagrams naturally fit in this context, but not the Holes representations. Though, from a practical point of view, it's rather unlikely that a same –real valued– hole appears twice in the same (bounded size) list.

An intuitive dissimilarity measure in $\mathcal{G}$: An intuitive and non-metric 'dissimilarity' measure has been designed in order to focus on alleles difference and diversity rather than alleles number. This distance happens to be almost equal to the minimal number of elementary[18] mutations, defined above, in the context of a discrete version of the representations.
An unmatched allele of x' increases the Δ^φ distance according to its difference from alleles of x, rather than by adding a fixed amount ($diam_\delta(\mathcal{A})$ for the Δ distance). Note finally that Δ and Δ^φ coincide in the space of fixed length alleles.

$$\Delta^\varphi(x,x') = \sum_1^n \delta(a_i, a'_{\mu(x,x')(i)}) + \sum_{\mu(x,x')^{-1}(i)=\emptyset} \min_{j\in[1,n]} |a'_i - a_j|, \qquad (5)$$

where $\mu(x,x')$ is the matching that minimizes the expression of $\Delta^\varphi(x,x')$.
The non-metric Δ^φ demonstrates to be better suited than the actual distances Δ to study the diversity of genotypes in the FD plot, leading to the design of more efficient initialization procedures.

Comparison tools for genotypic distances: An important issue after a distance has been designed, and before using it to analyze the behavior of an Evolutionary Algorithm is to ensure that there is indeed some link between the distance and the operators at hand (crossover, mutation): different adequacy tools are proposed to characterize the adequation between the genotypic distances and the genetic algorithm. These tools are used in next section to select the distances that are most suitable to study the fitness landscapes on the variable length representations at hand.
Some of the adequacy tools are summarized in the following, showing large differences between the matching distances and the *Hausdorff* distance, especially when EA optimization of *distance fitness functions* is involved:

[32]Elementary mutation steps (as adding 1 unit to the coordinates of the rectangle), in the context of a discrete (integer valued) version of the holes or Voronoï representations.

1. **Distance landscapes:** We propose to consider GA performance on each *genotypic distance fitness* (the fitness is proportional to the distance to solution) as a measure of the adequacy of the distance to describe the similarity in the genotypic space. This criterion have been found very instructive: Experiments show that a population based GA was unable to optimize some *Hausdorff fitness* functions (as well as some area based distance functions), whereas it succeeded in finding the optimum genotype when a matching based distance is used for the fitness (in this latter case, even a simple (1+1)-ES scheme was successful).

2. **Phenotypic-Genotypic correlation:** Given a point $s \in \mathcal{G}$ in the search space, we compute the correlation between the genotypic distance to s and the phenotypic distance to s, on a random sample of points of $\mathcal{G}$. Experiments exhibit important differences in genotypic diversity (on the x axis) according to the distance at hand. The smallest dispersion in the FD-plot are obtained with the *Hausdorff* distance.

3. **Distances and mutation:** The genotypic distance between and individual and its mutated is computed, when a minimal mutation is used (smallest modification that changes the phenotype). Experiments with the matching distances show that a small modification of the genotype yields a small change in distance (to the initial genotype). At the opposite, the *Hausdorff* distance can be very sensitive to small genotype change.

4. Other criteria have been designed to investigate the distances in relation to crossover operators and to the degeneracy of the representations. But no striking differences between the distances were found.

$\longrightarrow$ The ability of EAs to optimize distance landscapes (within a time of bounded complexity in problem size to be compared to $l \ln l$ of chapter I) is an important criterion for the choice of a genotypic distance. Other criteria are given by the dispersion of random points in the scatter 'phenotypic $\times$ genotypic distance' to optimum plot, or by the strong causality between small genotype changes and genotypic distance.

III.6 Landscape based analysis

The tools for distance comparison presented above, allow to choose the distances that seems best suited to study the Holes and Voronoï based genotypic spaces for evolutionary optimization. For instance, the Holes (resp. Site) based distances. No major difference was found between linearly and exponentially scaled distances, however, a clear advantage in favor of the exponential scales has been observed, with the distance-landscapes criterion.

Using the holes (resp. Site) based genotypic distances, some new sampling procedures are proposed to generate problem dependent "stable" samples, and efficient initialization procedures are designed in light of FD-plot analyses, as in Section II.5. Further, FD-plot demonstrates to be of potential use to chose the material to be represented with holes.

We also give further evidence that the operators can give more useful hints on the convergence than the correlation of the fitness landscape witnessed by FDC: The *offspring ranking* method demonstrates to be useful to compare representations. Yet, the use of different initializations weakens the comparisons, on the long term. Again this calls for on-line sampling and on-line measures.

Nevertheless, some general guidelines can be derived from the above studies. In the specific case of inclusion identification problem class, a heuristic for the *a priori* choice of a representation is: The Voronoï representation seems the most robust choice in the absence of any a priori information. Otherwise, if the solution contains non-equal quantities of both materials, the Hole representation is also efficient (when the holes correspond to the minority material). The FD-plot can guide the choice of the default and holes materials. On the other hand comparing the representations, necessarily involves the genetic operators.

IV Discussion

Aiming at analyzing EA behaviors, one key conclusion of both the theoretical first chapter and the experimental second chapter is that the long time behavior of EAs on such big search spaces is not reflected by typical properties (expected behavior of both mutations, or expected fitness improvement of operators, or average indicators), but is due to the deviant **atypical properties**. A direct consequence is that the fitness landscape of EA operators is related in a non trivial manner, to the exploration mechanism.

Further, aiming at predicting EA behavior, the key issue for the design of a useful landscape indicator is to find a characterization of a broad class of fitness functions on which the EA behaves similarly. Clearly, the second chapter shows that the class of monotone[33] fitness functions is too small to satisfy this requirement. Note that finding a good definition of "**similar behavior**" is also a major problem, as it is particularly EA dependent. As a first step to meet this requirement, a method of comparison of EA behaviors (trajectory and convergence) has been proposed and experimented, showing that **genotypic distances** are particularly useful to scale and compare EA behaviors.

However, the choice of genotypic distance, relevant with respect to the evolution process, is a crucial and non-trivial issue. The third chapter suggests that the ability of EAs to optimize **distance landscapes** –within a time of bounded complexity in problem size– is a good criterion for the choice of a genotypic distance. Other criteria are given by the dispersion of random points in the scatter 'phenotypic $\times$ genotypic distance' to optimum plot, or by the strong causality between small phenotypic changes and genotypic changes (in distance).
Finally, the genotypic distances prove useful to compare initialization procedures, and to tune their parameters (better procedures yield random samples with a big dispersion in the fitness $\times$ distance space).

On the other hand, the choice of a representation for a given problem cannot be decided from the static analysis of the landscape related to the genotypic distances (even designed in relation to mutation). Rather, the atypical behaviors of evolution operators are responsible for the difference in EA performance with different representations. The *offspring ranking* method demonstrates useful to compare atypical operators properties in both *ad hoc* function and real world function cases.

[33]wrt. to the distance to optimum

Chapter I

EA Convergence

Plan du Chapitre I

Ce chapitre est dédié à l'étude de la convergence des algorithmes d'évolution, du point de vue quantitatif. La trajectoire et le temps de convergence sont comparés pour différents opérateurs et algorithmes. Le but de ce chapitre est d'identifier les comportements limites des algorithmes d'évolution.

I.2 Temps de convergence d'un processus $(1,1)-ES$: Comparaison de deux mutations

Cette première partie est dédiée à la comparaison de la trajectoire et temps de convergence d'un processus rudimentaire sans sélection, avec deux mutations différentes. Les lois asymptotiques du premier temps d'atteinte de la solution sont calculées, indiquant l'écrasante supériorité, en moyenne et à court terme, de la 1-bit mutation sur la $\frac{c}{l}$ mutation, pourtant, plus utilisée en pratique. Les trajectoires des deux processus (avec chaque mutation) coïncident à des instants, qui suivent asymptotiquement une loi de Poisson ...

I.3 Temps de convergence d'un processus $(1+1)-ES$: Comparaison de deux mutations

Ayant obtenu les temps de convergence du processus $(1,1)-ES$, en l'absence de toute pression sélective, il s'agit maintenant de reconduire la comparaison entre les mêmes mutations en présence de selection, avec le processus $(1+1)-ES$ sur la fonction OneMax. Les lois asymptotiques du premier temps d'atteinte de la solution sont encore en faveur de la 1-bit mutation par rapport à la $\frac{c}{l}$ mutation, non seulement en nombre de générations, mais aussi en nombre d'évaluations de la fonction fitness. En effet, la $\frac{c}{l}$ mutation modifie 1 bit en moyenne par individu, certains individus ne sont donc pas mutés.

Cependant, il existe au moins une classe de problèmes où la $\frac{c}{l}$ mutation prend, et de loin, le dessus: Les *longpath* de Horn sont construits de telle manière qu'un processus $(1+1)-ES$ avec la 1-bit-mutation, ne peut que suivre un chemin de taille exponentielle pour converger. La $\frac{c}{l}$ mutation peut prendre des raccourcis, ramenant le temps de convergence à un ordre polynomial.
Existe-t-il des cas où la $\frac{c}{l}$ mutation, elle aussi, soit amenée à 'suivre' un chemin donné, avec un temps de convergence exponentiel ?

I.4 Temps d'un processus $(\mu+\lambda)$ convergence sur les long k-path

Les *long k-path* de G. Rudolph sont une géneralisation des *longpath* de Horn. L'étude de la distribution statistique du temps de convergence d'un processus $(\mu+\lambda)$ sur ces *long k-path*, permet d'identifier une sous-classe de ces fonctions telle qu'un processus $(\mu+\lambda)$-ES muni de la $\frac{c}{l}$ mutation converge sans prendre de raccourcis, donc en un temps exponentiel en la taille du problème.
D'autre part, l'étude de l'influence de la taille de la population révèle une situation caricaturale pour laquelle, contrairement aux cas usuels, la taille de population idéale est égale à un.

I.5 Chemins controlés pour l'AG

La section précédente prouve qu'il existe des chemins exponentiels et contrôlés pour la $\frac{c}{l}$ mutation. Cependant, la question reste ouverte pour l'opérateur de croisement, qui prend des raccourcis conséquents sur ces mêmes *long k-path*.

Etant donné une suite aléatoire de points différents de l'espace, cette partie propose une méthode générique pour construire des 'chemins contrôlés' à partir de cette suite de points. L'expérience montre qu'un AG muni des opérateurs de mutation et de croisement, converge en 'passant' dans l'ordre, par les points de la suite prédéfinie. Ce résultat, empirique pour le croisement, est aussi démontré dans le cas des 1-bit et $\frac{c}{l}$ mutations, sur un sous-chemin des *long k-path*.

Remerciements Je tiens tout particulierement à remercier Josselin Garnier, pour sa contribution précieuse à ce chapitre. Sa maitrise technique et son investissement personnel ont permi d'obtenir ces résultats, qui je l'espère contriburont à une meilleure compréhension des mécanismes complexes des algorithmes stochastiques.

Par ailleurs, je voudrais remercier tout aussi chaleureusement, Bart Naudts co-auteur pour la troisième section de ce chapitre.

Publications: The two first sections are made in collaboration with Josselin Garnier, and are available as technical reports [8,9]. The last section is an edited version of a joint (published) paper with Bart Naudts [5].

Overview of Chapter I

This chapter presents quantitative results about the convergence of some evolutionary algorithms. In the binary framework, the first hitting times of the solution are compared for different mutation operators and two types of evolutionary strategies, asymptotically in terms of the the size of the bitstring. The last section, extends some results to the case of population based Genetic Algorithms using crossover operators.

I.2.3 First hitting times of a (1,1)–ES for two mutation operators

The first subsection considers the oversimplified framework of a $(1,1)$–ES, equivalent to an evolutionary process without any selection, or to a completely flat landscape with a unique isolated peak (needle in the haystack). The asymptotic laws of the first hitting time of the solution are derived exactly for both the stochastic $\frac{c}{l}$ mutation (all bits are flipped independently with probability $\frac{c}{l}$) and the deterministic *1-bit* mutation (exactly one bit is flipped, randomly chosen in the bitstring). Surprisingly, the result proves that in this framework, the *1-bit* deterministic mutation outperforms the stochastic mutation.

I.2.4 First hitting times of a (1+1)–ES for two mutation operators for the OneMax problem

The same study is carried over to the case of the $(1+1)--ES$ optimizing the well-known OneMax function, with the same asymptotic advantage for the *1-bit* deterministic mutation over the stochastic $\frac{c}{l}$ mutation, even when only considering the number of active moves (as the stochastic $\frac{c}{l}$ mutation sometimes does not flip any bit).
These two original theoretical results (only the order $l\log(l)$ of the complexity of these hitting times was known) tend to modify the common understanding of those binary mutation operators. Moreover, an interesting consequence is derived from the variances of those processes: in those contexts, there is no benefit in considering a large number of runs in parallel.

I.3 First hitting times of $(\mu+\lambda)-ES$ on long k-path problems

Rudolph's *long k-paths*, a generalization of Horn's *longpath*, are built in such a way that the $(1+1)$–ES using a deterministic *1-bit* mutation can only "follow the path" to reach the solution. This path has a length increasing exponentially in terms of the string length. On the other hand, the stochastic $\frac{c}{l}$ mutation can sometimes take shortcuts, hence achieving polynomial convergence time, as proved by Rudolph.
Considering the case where k increases with l as l^α, it is proved that the *long k-path* landscape is also of exponential difficulty for the stochastic $\frac{c}{l}$ mutation. The case of fixed k is also investigated. Examining the probability that very rare events take place, it is proved that the best population size of the $(\mu+\lambda)$–ES is ... 1, and that the best strategy is to run multiple $(1+1)$–ES in parallel.

I.4 Candidate Longpaths for Genetic Algorithms

Does there exist longpath-landscapes for the crossover operator? This question is addressed in the last part of this chapter, which proposes a generic way to build such landscapes from

a series of "landmarks" distributed over the search space.
The "long path" characteristic of theses landscapes is empirically demonstrated using different instances of Genetic Algorithms, while it is proved for both deterministic and stochastic mutation operators when the landmarks are taken from Rudolph's *long k-path*.

Publications: The two first sections are made in collaboration with Josselin Garnier, and are available as technical reports [8,9]. The last section is an edited version of a joint (published) paper with Bart Naudts [5] (References from List of papers after the Appendix).

Chapter I
EA Convergence

I.1 Introduction

One known drawback of Evolutionary Algorithms as function optimizers is the amount of computational efforts they require to reach their goal. Hence, a lot of work has been devoted to the study of the theoretical properties of these algorithms, aiming at bounding their complexity, or time-to-solution.
In real-valued spaces, an exhaustive set of results has been obtained for some particular functions (e.g. the sphere and the corridor models) since the early days of Evolution Strategies [Rec73, Sch81]. The most recent and powerful results provide bounds for the Euclidean distance to the solution in different situations: Beyer [Bey93, Bey94, Bey95a, Bey95b] handles sophisticated variants of standard ES (i.e. based on Gaussian mutations, with or without self-adaptive mutation parameters), while Rudolph [Rud97a, Rud97b] studies the local convergence speed for a larger class of mutation operators.

The situation is not so clear when it comes to optimization in the discrete bitstring space. Though some results prove the global convergence of evolutionary algorithms, very few works have succeeded in addressing the issue of convergence time.
Initial investigations about Genetic Algorithms (GAs) concentrated on a constructive view of the algorithms, based on the Schema Theorem and the Building Blocks hypothesis [Hol75, Gol89]. Though the Schema Theory has been widely discussed and commented upon [Ant89, Wil91, DW91], it has also been usefully generalized [Rad91a], leading to the heuristic approaches to the design of efficient algorithms [SR96]. But, it has never led to any strong theoretical result about convergence, or complexity, of the algorithm itself.

The main theoretical results in the field of evolutionary binary function optimization have been obtained using Markov chain analyses. Though considered a marginal operator in the GA paradigm [Hol75, Gol89], the mutation operator is at the core of all convergence results obtained so far in that domain, as mutation is responsible for the ergodicity of the Markov chain representing the evolutionary process. Early works [EAH91, DP93, NV92, Suz93] consider simplified models; Rudolph convergence results [Rud94b] directly rely on the positivity [34] of the mutation operator, while their recent extension by Agapie [Aga97] relaxes the strong hypothesis of positive mutations to the existence of a finite chain of mutations linking any two points of the space with non-zero probability (irreducible and diagonal positive transition mutation matrix).
Also in the Markov framework, but using completely different techniques (the Friedlin-Wentzell theory of stochastic perturbation of dynamical systems), Cerf [Cer94, Cer96] obtains a lower bound for the population size, allowing him to prove finite time convergence results for an inhomogeneous Genetic Algorithm. His results are also primarily based on mutation (but he shows that an acceleration of convergence is clearly brought by crossover).
None of these results makes any particular assumption [35] on the fitness function. But on

[34] there exists a mutation that links any two points of the space with non-zero probability
[35] The only prerequisite, is that the fitness is supposed to be totally known.

the other hand, they do not give any indication about the complexity of the algorithm: the hitting time of the associated Markov chain (i.e. the expectation of the number of generations required to reach the optimum) remains unknown.

The first results about complexity [Bäc92, M9̈2, Bäc93, Rud97b] have been obtained on the simple, well-studied, OneMax problem (the fitness is the number of ones in the bitstring, as introduced by Ackley [Ack87]), though some other particular functions were studied as well (e.g. in [Rud96, DJW98]). And again, most of these works concentrated on the mutation operator[36].

In that same line, the first part of this chapter investigates two types of binary mutation operators, namely the standard bitwise stochastic bit flipping $\frac{c}{l}$ mutation (all bits are flipped independently with the same probability) and the deterministic 1-bit flipping mutation that flips exactly 1 bit of each bitstring, randomly chosen with uniform probability.
The efficiency of these operators are compared through the derivation of exact hitting times of the corresponding Markov chains. First, the operators are applied on an arbitrary fitness landscape, without selection, on a single bitstring, e.g. in a (1,1)-ES algorithm (section I.2.3). Second, the coupling with deterministic selection, the (1+1)-ES algorithm, is investigated for the OneMax problem (section I.2.4).
In both cases, the exact expectations of the average hitting times, as well as their standard deviations, are calculated up to terms of order l (the size of the bitstrings). Moreover, the first study goes beyond complexity considerations, by comparing the respective trajectories obtained with each mutation.

The second part of the chapter answers the challenging question of the existence of longpaths for the stochastic $\frac{c}{l}$ mutation: section I.3 proposes an extensive study of $(\lambda + \mu)$-evolution strategies with the stochastic $\frac{c}{l}$ mutation on long k-path problems [Rud96]. Both cases of a fixed and variable k (a function of l) are addressed, and the influence of population size is investigated.

Finally, the last part of the chapter addresses the same question of longpaths existence, but for crossover operators and population based stochastic algorithms. In section I.4, a family of *controlled paths* functions is constructed, based on an arbitrary choice of keys which define a path in the search space. Experiments show that dynamic GA trajectory goes successively through each point along this path before it ends at the global optimum.
Moreover, we prove that controlled paths, with keys extracted from the long k-path, also define longpaths for the stochastic $\frac{c}{l}$ mutation.

Framework From now on, the search space will be the binary space $E = \{0,1\}^l$ for some $l \in \mathbb{N}$, and we shall develop our theory in the asymptotic framework $l \gg 1$. We shall consider the Hamming distance d defined between two strings $X, X' \in E$ by: $d(X,X') = \sum_{i=1}^{l} \mathbb{I}_{X_i \neq X'_i}$.

[36] though the influence of crossover has been investigated by Culberson [CL96], who shows, on the OneMax problem, that the number of local optima in CX landscape increase exponentially in problem size. Nevertheless, to circumvent these local optima, it is essential that crossover takes place between fairly close parents[HR96b].

I.2 Rigorous Hitting Times for Binary Mutations

Overview In the binary evolutionary optimization framework, two mutation operators are theoretically investigated. For both the standard (stochastic bit flipping) mutation, in which all bits are flipped independently with the same probability, and the deterministic 1-bit-flip mutation, which flips exactly one bit per bitstring, the statistical distribution of the first hitting times of the target are thoroughly computed (expectation and variance) up to terms of order l (the size of the bitstrings) in two distinct situations: without any selection, or with the deterministic (1+1)-ES selection on the OneMax problem. In both cases, the deterministic 1-bit-flip mutation convergence time is smaller by a constant (in terms of l) multiplicative factor. These results extend to the case of multiple independent optimizers.

I.2.1 INTRODUCTION

Global convergence results of GAs give mainly sufficient conditions that rely on the irreducible property of mutation operators. While both deterministic and stochastic $\frac{c}{l}$ mutations are irreducible, most results concern the case of the $\frac{c}{l}$ positive mutation. Agapie [Aga97] goes a step further by providing a necessary and sufficient condition for global GA convergence. Yet this result is only valid for the convergence of GAs whose mutation matrix has a diagonal with positive terms only, which excludes the 1-bit-mutation case.

The main global convergence result that includes the deterministic 1-bit-flip mutation (and all irreducible mutations), is, as far as we know, due to Cerf [Cer94, Cer96]. However, in the case of an inhomogeneous GA, where nearly all theoretical results only consider the homogeneous case (i.e. mutation rate do not change along evolution), Cerf's results [Cer94, Cer96] emphasize a decreasing mutation rate (tightly coupled to an increasing selection pressure) to reach finite time convergence. On the other hand, a decreasing mutation rate along the generations is a common practice, even though very few theoretical results support it. (in [BS96], an hyperbolic decreasing scheme is advocated). Once again, practice is one step beyond theoretical studies.

Cerf's results, however, give no estimation of hitting times. The first results about complexity comparing the two mutations [Bäc92, M$\ddot{9}$2, Bäc93, Rud97b, DJW98] have been obtained on the simple well-studied OneMax problem (a detailed state of the art is given in section I.2.4). They prove that the complexity of the (1+1)-ES with 1-bit-flip and $\frac{c}{l}$ mutation is in $O(l\log(l))$. Still, these works cannot be used to actually compare the stochastic and the deterministic bit flipping mutations operators, as they provide order of magnitude results without expliciting the multiplicative constants.

Some hints have been given, though, in the work of Rudolph. In [Rud96], he proves that for large values of l the (1+1)-ES using stochastic $\frac{1}{l}$ mutation is a lot quicker than the single-bit-flip Hill Climber in optimizing long k-path problem with fixed k values (polynomial versus exponential time in l). The difference between the two algorithms is that the (1+1)-ES with the stochastic $\frac{1}{l}$ mutation can do more than a one bit-flip at the same time: The stochastic $\frac{1}{l}$ mutation flips one bit per bitstring *in average*.

However, an immediate consequence is that the stochastic $\frac{1}{l}$ mutation (or more generally the stochastic $\frac{c}{l}$ mutation) is also "able" to do no bit-flip at all on some individuals. Rudolph also mentions that the distribution of the random variable K representing the number of flipped bits per bitstring with the stochastic $\frac{c}{l}$ mutation is well-known to be

$$P[K=k] = (C_k^l)(\frac{c}{l})^k(1-\frac{c}{l})^{l-k} \simeq \frac{e^{-c}}{k!}c^k$$

A straightforward consequence is that, for large values of l, a fraction e^{-c} of the stochastic mutations do not modify the bitstring at all (about one third for $c = 1$). This raises a serious question about the efficiency of stochastic mutations. Such a big amount of non-mutated individuals causes a consequent waste of time.
In the following, the efficiency of these operators is compared in two situations. First, the operators are applied on an arbitrary fitness landscape, without selection, on a single bitstring, e.g. in a (1,1)-ES algorithm. Second, the coupling with deterministic selection, the (1+1)-ES algorithm, is investigated for the OneMax problem.
In both situations, the exact expectations of the average hitting times, as well as their standard deviations, are calculated up to terms of order l (the size of the bitstrings).

This section is organized as follows. In subsection I.2.2, results with both deterministic and stochastic bit flipping mutation operators and both selection cases are summarized.
Subsection I.2.3 introduces the main results followed by their proofs, in the case of the (1,1)-ES (i.e. without fitness based selection), while section I.2.4 derives the hitting times of the two mutation operators coupled with selection. The final subsection presents some hints for future directions of research. Some basic results in probability theory, intensively used throughout the demonstrations, are recalled in the Appendix.

I.2.2 SUMMARY OF RESULTS

The first part of this study is restricted to the study of both the stochastic and the deterministic bit flipping mutations in the absence of selection (i.e. inside a simple (1,1)-ES algorithm). Such an algorithm can also be viewed as a random walk directed by the corresponding mutation. The purpose of this study is then to calculate the statistical distribution of the first hitting time of a given target point.
The main result (see Section I.2.3) in that case is that the expectation of the first hitting time for the stochastic $\frac{c}{l}$ mutation is larger by factor $(1 - e^{-c})^{-1}$ than that of the deterministic 1-bit-flip mutation. The variances and the complete statistical distributions of the first order with respect to l of both hitting times are also derived. It is found that the hitting time of the stochastic $\frac{c}{l}$ mutation (resp. deterministic 1-bit-flip mutation) obeys an exponential distribution with mean $\frac{2^l}{(1-e^{-c})}$ (resp. 2^l). Furthermore, if we compare the sequences of visited points by both mutations, we see that the stochastic $\frac{c}{l}$ mutation jumps over some points visited by the deterministic *1-bit-flip* mutation. Hence, when the first process reaches the target point for the first time, the second process misses it with probability $1 - \frac{1-e^{-c}}{c}$.
Finally, the case of several independent processes is addressed. The first hitting time of N independent stochastic $\frac{c}{l}$ mutations (resp. deterministic 1-bit-flip mutations) obeys an exponential distribution with mean $\frac{2^l}{N(1-e^{-c})}$ (resp. $\frac{2^l}{N}$).

The second part of this study brings together the mutation operators and the simple deterministic (1+1)-ES selection (a single parent generates a single offspring, and the fittest of both becomes the next parent).
It has already been argued [Rud97b] that it is easier to prove useful results on the OneMax problem than on any other binary function optimization problem, as increasing the fitness is equivalent to decreasing the Hamming distance to the optimum.
This study indeed buys this argument, as all computations of Section I.2.4 make intensive use of that fitness-distance equivalence to precisely study the statistical distribution of the first hitting times of the global maximum (all ones) when using either mutation operator. It is proved that the hitting time $\bar{T}$ of the stochastic $\frac{c}{l}$ mutation (resp. the hitting time

T of the deterministic *1-bit-flip* mutation) can be expanded up to terms of order $o(l)$ as: $\bar{T} = \frac{e^c}{c} l \ln l - \frac{e^c}{c} l \ln Z + lC(c)$ (resp. $T = l \ln l - l(\ln Z + \ln 2)$), where Z obeys an exponential distribution with mean 1 and density $p(z) = e^{-z} \mathbb{I}_{z \geq 0}$, and $C(c)$ is some c-dependent constant. Therefore the dominant term is deterministic and is equal to $\frac{e^c}{c} l \ln l$ (resp. $l \ln l$), and the normalized variance (that is to say the variance divided by the mean) is of the order of $(\ln l)^{-1}$. We then address the case of N independent processes, and prove that the first hitting time $\bar{T}_N$ of N independent stochastic $\frac{c}{l}$ mutations (resp. the first hitting times T_N of N independent deterministic *1-bit-flip* mutations) can be expanded as $\bar{T}_N = \frac{e^c}{c} l \ln l - \frac{e^c}{c} l \ln Z_N + lC(c)$ (resp. $T_N = l \ln l - l(\ln Z_N + \ln 2)$), where Z_N obeys a distribution with density $p_N(z) = N(1-e^{-z})^{N-1} e^{-z} \mathbb{I}_{z \geq 0}$. Hence there is very little gain when using several independent processes. This is due to the weak variances of the distributions of T and $\bar{T}$.

I.2.3 MUTATIONS WITHOUT SELECTION: THE (1,1)-ES

In any evolutionary algorithm, the strong interactions between selection and evolution operators (crossover, mutation, ...) make it difficult to efficiently compare the effects of different operators without taking into account the characteristics of the selection (i.e. the fitness function).
A standard method for avoiding such a bias, is to start by considering the effects of each operator alone: Selection without evolution operators (see the studies of *takeover time* [GD91, CDC96]), or *innovation time* of crossover or mutation in the absence of selection [TG93, Gol98].

More precisely, the *takeover time* is the time (number of generations) necessary for the best individual to colonize a population, under some selection scheme, and the *innovation time* for crossover is the mean time (number of generations) for recombination to achieve a solution that is better than any achieved to this point. Goldberg proposes an experimental study, on a simple fitness function, to determine the boundaries of the 'success domain' in the space of selection pressure vs. crossover probability, such that take over time is greater than or equal to the innovation time.

In that line, this subsection is restricted to the study of both the stochastic and deterministic bit flipping mutations in the absence of selection (i.e. inside a simple (1,1)-ES algorithm). The purpose is then to evaluate as precisely as possible the statistical distribution of the first hitting time of a given target point.
Two processes X and $\bar{X}$ are considered in the following, respectively using the deterministic and the stochastic bit flipping mutations.

Process X. The process $X(n)$ is a process starting from some point X^0, whose evolution is governed in the following way. At each time n, you build the string $X(n)$ by inverting each bit of $X(n-1)$ independently with probability c/l. The process $X(n)$ is a Markov chain whose transition matrix of size $2^l \times 2^l$ defined by:

$$P_{X,X'} := \mathbb{P}(X(1) = X'/X(0) = X),$$

is given by:

$$P_{X,X'} = (cl^{-1})^{d(X,X')} \left(1 - cl^{-1}\right)^{l-d(X,X')}.$$

The process X corresponds to a (1,1)-ES with a stochastic c/l mutation.

Process $\bar{X}$. The process $\bar{X}(n)$ is a process starting from some point X^0, whose evolution is governed in the following way. At each time n, you choose a random integer uniformly over $\{1,...,l\}$. Then you invert the corresponding bit of the string $\bar{X}(n-1)$. The resulting string is $\bar{X}(n)$. The process $\bar{X}$ is a Markov chain whose transition matrix is:

$$\bar{P}_{X,X'} = \begin{cases} 0 & \text{if } d(X,X') \neq 1 \\ l^{-1} & \text{if } d(X,X') = 1 \end{cases}$$

The process $\bar{X}$ corresponds to a (1,1)-ES with a deterministic 1-bit-flip mutation.

Goal of the study. We fix some point X^f and we look for the statistical distributions of the respective first hitting times T and $\bar{T}$ of the point X^f by the chains $X(n)$ and $\bar{X}(n)$.

Remark I.2.1
In the following, we use Markov chains to model the dynamic of the processes $X(n)$ and $\bar{X}(n)$. In fact, the Markov chain approach allows us to derive the complete distributions of the hitting times. The complete distribution is indeed necessary to estimate the variance of the hitting time, and also to find the distribution and the expectation of the minimum T_N of the first hitting times of N independent processes. An estimate of the expected time without an estimate of the corresponding variance, does not give any hint of the order T_N. For instance, if the variance is large, one can expect that $\mathbb{E}[T_N] \ll \mathbb{E}[T_1]$, since one of the independent process will certainly take an exceptional low value. Such results are indeed amongst the most important corollaries derived in the thesis.
On the other hand, note that one could have obtained partial results comparing these mutations more easily, without a Markov chain approach. Removing the steps where the stochastic mutation does not modify the bitstring at all (i.e, a fraction of e^{-c}), we have the following result.
We can prove without a Markov chain approach that the deterministic 1-bit-flip mutation and the stochastic active *mutation require the same effort to reach their goal, in the absence of selection and in the asymptotical framework.*
The approach is outlined below, and shows that considering only active mutation steps, the laws of the hitting times with either mutation are identical in the asymptotical framework. However, this approach doesn't give the actual laws.

Proof. Recall that occasionally, the stochastic c/l mutation does not flip any bit. A direct consequence is that the stochastic mutation visits fewer points of the search space than the deterministic mutation does. However, this fact is not sufficient to conclude and compare the mutations. In fact, the probability that the (1,1)-ES process visits twice (or more) the same point of the search space is not the same according to the mutation used. Hence, we have to compare the cardinal of the set of visited points rather than their number. In the following, we only outline the proof:
Let $P1$ be the probability that the deterministic mutation (M1) process converges before time t:

$$P1 = \mathbb{P}(\bar{T} \leq t) = \frac{\#\{\text{visited points up to time } t \text{ for M1}\}}{2^l}$$

Symmetricly, note $P2$ the probability that the stochastic active mutation (M2$^+$) process converges before time t:

$$P2 = \mathbb{P}(T \leq t | \text{ non active mutation steps are removed})$$

$$P2 = \frac{\#\{\text{visited points up to time } t \text{ for M2}^+\}}{2^l}$$

To estimate the cardinal of the above sets, consider the probability that a mutation applied to string s, comes back at the next step to s. For the deterministic mutation this probability is exactly $1/l$. Whereas for the stochastic active mutation, this probability is $(\exp -c)/l$, it can be obtained by summing over all possible numbers of mutated bits. Hence, the difference between $P1$ and $P2$ is of the order of $1/l$. This approach however, doesn't yield to more precise results. Hence, we still need the Markov chains approach to find the exponential distribution (and expectation) of first hitting times as discussed above.

Note that in the case of fixed l values, the difference between P1 and P2 is of the order of $1/l$ in favor to the active stochastic mutation. However, this result doesn't hold if we include the steps where nothing happens: the difference between $P1$ and $P2$ is then of order 1 in favor to the deterministic mutation! □

I.2.3.1 Main results

In this subsection, we begin by stating the main results. The proofs of these results follow in the next subsections.

Proposition I.2.2
If $d(X^0, X^f) \neq 0$, then,

$$\mathbb{E}[T] \simeq \frac{2^l}{1-e^{-c}}, \qquad \mathbb{E}[\bar{T}] \simeq 2^l,$$

in the sense that $2^{-l} \times \mathbb{E}[T]$ (resp. $2^{-l} \times \mathbb{E}[\bar{T}]$) goes to $1/(1-e^{-c})$ (resp. 1) when $l \to \infty$.

Note that whatever c, the (1,1)-ES with a stochastic c/l mutation is slower than the (1,1)-ES with a deterministic 1-bit-flip mutation.

Remark I.2.3
We can compare these results with the mean values the first hitting times of X^f by some other simple processes. Let us consider the following processes.
1. Random process
Choose a string uniformly over the state space E, until the optimal string is selected. The first hitting time obeys the distribution:

$$\mathbb{P}(T_1 = n) = 2^{-l}(1-2^{-l})^{n-1}$$

whose mean is:

$$\mathbb{E}[T_1] = 2^l.$$

2. Exhaustive process
Choose some arbitrary order relation over E and the strings of E are tested one after the other according to the order relation. The first hitting time obeys a uniform distribution over $\{1, ..., 2^l\}$ whose mean is:

$$\mathbb{E}[T_2] = 2^{l-1}.$$

The second method presents the advantage that it never tests the same string twice, and consequently it looses less time than the first one. It has to be noticed that the strategy corresponding to the Markov chain $X(n)$ is the less efficient.

We can be more precise and give the complete statistical distributions of T and $\bar{T}$ in the framework $l \gg 1$.

Proposition I.2.4
If $d(X^0, X^f) \neq 0$, then $2^{-l}(1-e^{-c})T$ (resp. $2^{-l}\bar{T}$) converges in distribution to a random variable with exponential distribution and mean 1 as $l \to \infty$.

This Proposition has two straightforward corollaries.

Corollary I.2.5
If $d(X^0, X^f) \neq 0$, then the standard deviations of the variables T and $\bar{T}$ are equal to their respective means as $l \to \infty$.

$$\mathbb{E}\left[(T - \mathbb{E}[T])^2\right] \simeq \frac{2^{2l}}{(1-e^{-c})^2}, \qquad \mathbb{E}\left[(\bar{T} - \mathbb{E}[\bar{T}])^2\right] \simeq 2^{2l}.$$

Corollary I.2.6
Assume that we deal with a set of N processes X_i (resp. $\bar{X}_i$) which evolve independently. Then the global first hitting time T (resp $\bar{T}$), which is the minimum of the N independent hitting times, is such that the statistical distribution of $T2^{-l}$ (resp. $\bar{T}2^{-l}$) converges to that of an exponential random variable with mean $N^{-1}(1-e^{-1})^{-1}$ (resp. N^{-1}). In particular its expectation is:

$$\mathbb{E}[T] \simeq \frac{2^l}{N(1-e^{-c})}, \qquad \mathbb{E}[\bar{T}] \simeq \frac{2^l}{N},$$

The expected first hitting time is N times smaller than that of a single process. Hence, the number of evaluations remains the same, and there is no advantage to increase N.

I.2.3.2 Study of the chain $\bar{X}(n)$

Let us introduce the auxiliary Markov process $Z(n)$ with state space $\{0, ..., l\}$ starting from $Z(0) = k_f := d(X^0, X^f)$ and with transition matrix $\bar{Q}$:

$$\bar{Q}_{k,k'} = \begin{cases} 0 & \text{if } k' \neq k-1, k+1 \\ 1 - \frac{k}{l} & \text{if } k' = k+1 \\ \frac{k}{l} & \text{if } k' = k-1 \end{cases}$$

It is easy to check that:

$$d(\bar{X}(n), X^f) \overset{law}{=} Z(n). \tag{6}$$

Obviously the first hitting time of the point 0 by $Z(n)$ corresponds to the first hitting time $\bar{T}$ of the point X^f by the chain $\bar{X}(n)$.
Let us denote by ν_k the first hitting time of the point k by the chain $Z(n)$ and by $\bar{\nu}_k$ the first hitting time ≥ 1 of the point k. These times coincide if the chain starts from a point distinct from k. But if the chain starts from k, then $\nu_k = 0$ while $\bar{\nu}_k \geq 2$.
Remark: The Markov chain Z is known in the mathematical and physical literature under the term "Ehrenfest model". One can find a nice introduction to this model in [Ios, Section 4.6], and one can also find in [KS, Section 7.3] the computation of the expectation of the first hitting time ν_0 when starting from $l/2$, so that Lemma I.2.7 is not really new. Nevertheless, we shall use Lemma I.2.7 and intermediate results derived in its proof to exhibit the complete distribution of ν_0, and not only its expectation. The complete distribution is indeed necessary to estimate the variance of ν_0, and also to find the distribution and the expectation of the minimum of the first hitting times of N independent process $(\bar{X}_i)_{i=1,...,N}$.

Lemma I.2.7
1. If $k_f = 0, 1, 2, 3$, we have for any l[37]:

$$\mathbb{E}_0[\nu_0] = 0,$$

$$\mathbb{E}_1[\nu_0] = 2^l - 1,$$

$$\mathbb{E}_2[\nu_0] = \frac{l}{l-1}2^l - \frac{2l}{l-1},$$

$$\mathbb{E}_3[\nu_0] = \frac{l^2 - l + 2}{(l-2)(l-1)}2^l - \frac{3l^2 - 3l + 2}{(l-2)(l-1)}.$$

2. If $k_f = 1, ..., l$, then we have for $l \gg 1$:

$$\mathbb{E}_{k_f}[\nu_0] \simeq 2^l,$$

which means that $2^{-l} \times \mathbb{E}_{k_f}[\nu_0]$ goes to 1 as $l \to \infty$.

Proof. Step 1. Statistical expectations of the hitting times.
Obviously $\mathbb{E}_0[\nu_0] = 0$. Let us assume in the following that $k_f \geq 1$. There exists a unique invariant probability measure for the chain $Z(n)$ (i.e. $\mu\bar{Q} = \mu$) which is given by:

$$\mu(k) = 2^{-l}C_l^k.$$

By the general theory of Markov chains, since $Z(n)$ is irreducible and recurrent [ch.5, exercise 3.14][Rev75], the first hitting time $\bar{\nu}_k$ satisfies:

$$\mathbb{E}_k[\bar{\nu}_k] = \frac{1}{\mu(k)}.$$

In particular:

$$\mathbb{E}_0[\bar{\nu}_0] = 2^l. \tag{7}$$

Starting from 0, the first jump is non random: it occurs at time 1 and the process goes to the point 1. Thus:

$$\mathbb{E}_0[\bar{\nu}_0] = \mathbb{E}_0[\bar{\nu}_0/Z(1) = 1] = 1 + \mathbb{E}_0[\bar{\nu}_0 - 1/Z(1) = 1].$$

Applying the weak Markov property (cf Appendix):

$$\mathbb{E}_0[\bar{\nu}_0] = 1 + \mathbb{E}_1[\nu_0],$$

which implies:

$$\mathbb{E}_1[\nu_0] = 2^l - 1. \tag{8}$$

Let $k \in \{0, ..., l-1\}$. Starting from $k+1$, the chain Z jumps either in k or in $k+2$. We can therefore split $\mathbb{E}_{k+1}[\nu_k]$ into two terms:

$$\mathbb{E}_{k+1}[\nu_k] = \mathbb{E}_{k+1}[\nu_k/Z(1) = k]\mathbb{P}_{k+1}(Z(1) = k) + \mathbb{E}_{k+1}[\nu_k/Z(1) = k+2]\mathbb{P}_{k+1}(Z(1) = k+2)$$

which implies by the weak Markov property:

$$\mathbb{E}_{k+1}[\nu_k] = 1 \times \bar{Q}_{k+1,k} + (1 + \mathbb{E}_{k+2}[\nu_k])\,\bar{Q}_{k+1,k+2}. \tag{9}$$

[37] $\mathbb{E}_k$ (resp. $\mathbb{P}_k$) stands for the expectation (resp. the probability) with respect to the statistical distribution of the chain $Z(n)$ starting from $Z(0) = k$.

The Markov chain X only goes to neighbored levels of $\{0,1\}^l$, and therefore the Markov chain Z has to reach $k'-1$,...,k while going from k' to k (with $k<k'$). Using the strong Markov property we then get for any $k'>k$:

$$\mathbb{E}_{k'}[\nu_k] = \sum_{j=k}^{k'-1} \mathbb{E}_{j+1}[\nu_j] \tag{10}$$

Furthermore, substituting Eq. (10) with $k'=2$ into (9), we get the inductive relation:

$$\mathbb{E}_{k+2}[\nu_{k+1}] = (\mathbb{E}_{k+1}[\nu_k]-1)\,\frac{k+1}{l-(k+1)} - 1. \tag{11}$$

Substituting (8) and (11) into the sum (10) with $k=0$, we get the complete expression of $\mathbb{E}_{k_f}[\nu_0]$, valid for any l. We can also bound $\mathbb{E}_{k_f}[\nu_0]$ for any $k_f \geq 1$ by:

$$\mathbb{E}_1[\nu_0] \leq \mathbb{E}_{k_f}[\nu_0] \leq \mathbb{E}_1[\nu_0] + \mathbb{E}_{k_f}[\nu_1]. \tag{12}$$

In the following two steps, we are going to prove that the last term of the right-hand member of this estimate is negligible compared to $\mathbb{E}_1[\nu_0]$ in the asymptotic framework $l\to\infty$. This is the main issue of the proof, and although it sounds surprising, it can be explained heuristically. Indeed, when the chain Z is at level 1, it has a chance $1/l$ to reach 0. It is much likely to go to level 2, and then to go the middle levels which contain almost all states. Thus, the chain needs to visit level 1 about $O(l)$ times to finally succeed in reaching level 0. So we could expect that $\mathbb{E}_1[\nu_0] \sim l\mathbb{E}_1[\bar{\nu_1}] \sim l\mathbb{E}_{k_f}[\nu_1]$. Hence $\mathbb{E}_{k_f}[\nu_1] \sim 2^l/l$. Let us turn to the proof of this statement.
Step 2. If $k \leq [l/2]$, then $\mathbb{E}_k[\nu_1] \leq 2^{l+1}/(l-1)$.
From (11) we compute :

$$\mathbb{E}_2[\nu_1] \;=\; \frac{2^l}{l-1} - \frac{l+1}{l-1}, \tag{13}$$

$$\mathbb{E}_3[\nu_2] \;=\; \frac{2^{l+1}}{(l-2)(l-1)} - \frac{l(l+1)+2}{(l-2)(l-1)} \tag{14}$$

Besides, for any $k \leq l/2$, Eq. (11) also implies that:

$$\mathbb{E}_{k+1}[\nu_k] \leq \mathbb{E}_k[\nu_{k-1}]$$

Summing with respect to $k=3,...$ starting from (14) yields that, for any $k\leq 2$:

$$\mathbb{E}_k[\nu_2] \leq \frac{2^{l+1}}{(l-1)(l-2)} \times \left(\frac{l}{2}-2\right) \leq \frac{2^l}{l-1}$$

Summing with (13) establishes the desired result.
Step 3. If $k \geq [l/2]$, then $\mathbb{E}_k[\nu_{[l/2]}] \leq l^2/4$.
If $Z(0)=l$, then the chain has no other choice but to jump to $l-1$:

$$\mathbb{E}_l[\nu_{l-1}] = 1. \tag{15}$$

Eq. (11) also reads as :

$$\mathbb{E}_k[\nu_{k-1}] = 1 + \frac{l-k}{k}\,(\mathbb{E}_{k+1}[\nu_k]+1)$$

which implies that, for any $k \geq l/2$:

$$\mathbb{E}_k[\nu_{k-1}] \geq 2 + \mathbb{E}_{k+1}[\nu_k]$$

By a recursive argument starting from (15): we get that:

$$\mathbb{E}_{k+1}[\nu_k] \leq 2(l-k)$$

Summing with respect to k yields the desired result.
Step 4. Conclusion
Injecting the statements of Steps 2 and 3 into:

$$\mathbb{E}_{k_f}[\nu_1] \leq \sup_{k \leq [l/2]} \mathbb{E}_k[\nu_1] + \sup_{k \geq [l/2]} \mathbb{E}_k[\nu_{[l/2]}]$$

we get that, for any $k_f \geq 1$:

$$\mathbb{E}_{k_f}[\nu_1] \leq \frac{3}{l} 2^l. \tag{16}$$

which yields the result of the Lemma in view of Eq. (12). □

Remark I.2.8
This proof also shows that 2^l steps are necessary to go from 1 to 0 (in distance to target point), while coming from any $k_f > 1$ to 1 in distance takes $O(2^l/l)$ steps.

Let us now study the statistical distribution of $2^{-l}\nu_0$.

Lemma I.2.9
If $k_f \geq 1$, then $2^{-l}\nu_0$ converges in distribution to a random variable with exponential distribution and mean 1 as $l \to \infty$.

Proof.
Step 1. Tightness of $2^{-l}\nu_0$.
By Lemma I.2.7:

$$\sup_{l \in \mathbb{N}} \mathbb{E}_{k_f}[2^{-l}\nu_0] < \infty$$

which is sufficient to establish the tightness (and hence the relative compactness) of the sequence of variables $2^{-l}\nu_0$ [EK86, Theorem II-2-2]. Tightness insures that there exists at least one weak limit of the sequence $(2^{-l}\nu_0)_{l \in \mathbb{N}}$.
Step 2. Convergence in distribution.
Let a, b be two positive real numbers. Since the chain Z is Markovian, we get from the strong Markov property that:

$$\mathbb{P}_k(\nu_0 \geq [2^l a] + [2^l b]/\nu_0 \geq [2^l a]) = \sum_{j=1}^{l} \mathbb{P}_k\left(Z([2^l a]) = j/\nu_0 \geq [2^l a]\right) \mathbb{P}_j(\nu_0 \geq [2^l b]) \tag{17}$$

On the one hand the following obvious estimate holds true for any $j \geq 1$:

$$\mathbb{P}_j(\nu_0 \geq [2^l b]) \geq \mathbb{P}_1(\nu_0 \geq [2^l b]). \tag{18}$$

On the other hand, conditioning with respect to the first time when the chain Z hits 1:

$$\mathbb{P}_j(\nu_0 \geq [2^l b]) = \sum_{t_1=0}^{\infty} \mathbb{P}_1(\nu_0 \geq [2^l b] - t_1)\mathbb{P}_j(\nu_1 = t_1)$$

If $\delta > 0$, then Markov inequality yields:

$$\mathbb{P}_j(\nu_1 \geq [2^l \delta]) \leq \frac{\mathbb{E}_j[\nu_1]}{[2^l \delta]}.$$

Combining with Eq. (16) implies that:

$$\mathbb{P}_j(\nu_1 \geq [2^l \delta]) \leq \frac{3}{\delta l}.$$

We then get:

$$\mathbb{P}_j(\nu_0 \geq [2^l b]) \leq \mathbb{P}_1(\nu_0 \geq [2^l b] - [2^l \delta]) \left(1 - \frac{3}{\delta l}\right) + \frac{3}{\delta l}. \tag{19}$$

Substituting (18) and (19) into (17) establishes:

$$\begin{aligned}\liminf_{l\to\infty} \mathbb{P}_1(\nu_0 \geq [2^l b]) \leq \liminf_{l\to\infty} \mathbb{P}_k(\nu_0 \geq [2^l a] + [2^l b]/\nu_0 \geq [2^l a])\\ \leq \limsup_{l\to\infty} \mathbb{P}_k(\nu_0 \geq [2^l a] + [2^l b]/\nu_0 \geq [2^l a]) \leq \limsup_{l\to\infty} \mathbb{P}_1(\nu_0 \geq [2^l b] - [2^l \delta]).\end{aligned} \tag{20}$$

Let ν be a weak limit of $2^{-l}\nu_0$. Such a limit exists by Step 1, but needs not to be unique, and may depend on the starting point k. Multiplying by 2^{-l} both members inside the $\mathbb{P}(...)$ terms of (20), we obtain:

$$\mathbb{P}_1(\nu \geq b) \leq \mathbb{P}_k(\nu \geq a + b/\nu \geq a) \leq \mathbb{P}_1(\nu \geq b - \delta)$$

In this expression δ is an arbitrary positive real number. Letting $\delta \to 0$, we find that the distribution of ν does not depend on k (take $a = 0$, $b > 0$) and satisfies for any $a, b > 0$:

$$\mathbb{P}(\nu \geq b) = \mathbb{P}(\nu \geq a + b/\nu \geq a).$$

Lemma B.3 in the Appendix then implies that ν is a random variable with exponential distribution. Since its mean is 1, we then get the uniqueness of the weak limit, and consequently the convergence in distribution of $2^{-l}\nu_0$ to this variable. □

I.2.3.3 Study of the chain $X(n)$

The approach is very similar to that of the previous section. Let us introduce the process Z:

$$Z(n) = d(X(n), X^f).$$

$(Z(n))_{n\in\mathbb{N}}$ is a Markov chain with transition matrix Q whose coefficients are for $j, j' \in \{0, ..., l\}$:

$$Q_{j,j'} = \sum_{k=0}^{j} \sum_{k'=0}^{l-j} \mathbb{I}_{k'-k=j'-j} C_j^k C_{l-j}^{k'} (cl^{-1})^{k+k'} (1 - cl^{-1})^{l-k-k'}.$$

The chain Z is recurrent and its unique invariant probability measure is μ:

$$\mu(j) = 2^{-l} C_l^j.$$

Let us adopt the following notations: ν_k stands for the first hitting time of the point k by the chain $Z(n)$ and $\bar{\nu}_k$ stands for the first hitting time ≥ 1 of the point k. The expectation with respect to the statistical distribution of the chain Z starting from k is denoted by $\mathbb{E}_k$.

Lemma I.2.10
If $k_f \geq 1$, then:

$$\lim_{l\to\infty} 2^{-l}\mathbb{E}_{k_f}[\nu_0] = \frac{1}{1-e^{-c}}$$

Proof. In the following a is the constant defined by $a := \frac{\ln(1+e^c)-\ln 2}{c}$. It belongs to $(1/2,1)$.
Step 1. If $k \leq al$, then $\mathbb{E}_{k+1}[\nu_k] \leq 2^l(C_l^k)^{-1}e^c/((1-a)c)$.
Since the unique invariant probability measure of the chain Z is known, we can claim that the expectation of the hitting time $\bar{\nu}_k$ is:

$$\mathbb{E}_k[\bar{\nu}_k] = \frac{1}{\mu(k)} = 2^l(C_l^k)^{-1}.$$

By conditioning with respect to the first jump, we get that this expectation is bounded below by:

$$\mathbb{E}_k[\bar{\nu}_k] \geq \mathbb{P}_k\left(Z(1) = k+1\right)\left(1 + \mathbb{E}_{k+1}[\nu_k]\right). \tag{21}$$

Since the probability to go from k to $k+1$ is bounded below by the probability that one of the $l-k$ bits which coincide with those of X^f is inverted while the other are not:

$$\mathbb{P}_k\left(Z(1)=k+1\right) \geq (1-cl^{-1})^{l-1}(l-k)cl^{-1} \geq (1-ckl^{-1})e^{-1} \geq (1-a)ce^{-c}.$$

Substituting into (21) yields the result.
Step 2. If $k \geq al$, then $\lim_{l\to\infty} 2^{-l}\mathbb{E}_k[\inf_{j<[al]} \nu_j] = 0$.
Firstly, if $Z(0) = l$, then the first mutation event will bring the chain below $l-1$:

$$\begin{aligned}
\mathbb{E}_l\left[\inf_{j\leq l-1}\nu_j\right] &= \sum_{j=1}^{\infty} j\mathbb{P}_l(Z(1)=l, Z(2)=l, ..., Z(j-1)=l, Z(j)\leq l-1) \\
&= \sum_{j=1}^{\infty} j\mathbb{P}_l(Z(1)=l)^{j-1}\mathbb{P}_l(Z(1)\leq l-1) \\
&= \frac{1}{1-e^{-c}}
\end{aligned} \tag{22}$$

If $Z(0) = k \geq al$, then conditioning with respect to time 1 yields:

$$\mathbb{E}_k\left[\inf_{j\leq k-1}\nu_j\right] = \mathbb{P}_k(Z(1)\leq k-1) + \sum_{k'=k}^{l}\mathbb{P}_k(Z(1)=k')\times\left(1+\mathbb{E}_{k'}\left[\inf_{j\leq k-1}\nu_j\right]\right)$$

Since $\mathbb{E}_{k'}\left[\inf_{j\leq k-1}\nu_j\right] \leq \sum_{k''=k}^{k'}\mathbb{E}_{k''}\left[\inf_{j\leq k''-1}\nu_j\right]$, we have

$$\mathbb{E}_k\left[\inf_{j\leq k-1}\nu_j\right] \leq 1 + \sum_{k'=k}^{l}\mathbb{P}_k(Z(1)=k')\left(\mathbb{E}_k\left[\inf_{j\leq k-1}\nu_j\right] + \sum_{k''=k+1}^{k'}\mathbb{E}_{k''}\left[\inf_{j\leq k''-1}\nu_j\right]\right)$$

which can be rewritten as:

$$\mathbb{E}_k\left[\inf_{j\leq k-1}\nu_j\right] \leq \frac{1}{\mathbb{P}_k\left(Z(1)\leq k-1\right)}\left(1+\mathbb{P}_k\left(Z(1)\geq k+1\right)\times\sup_{k'\geq k+1}\mathbb{E}_{k'}\left[\inf_{j\leq k'-1}\nu_j\right]\right) \tag{23}$$

Starting from $Z(0)=k$, the probability that $Z(1)$ be smaller than k is bounded below by the probability that the $l-k$ bits which coincide with those of X^f are not inverted and at least one of the k bits which do not coincide with those of X^f is inverted. Thus:

$$\begin{aligned}\mathbb{P}_k\left(Z(1)\leq k-1\right) &\geq (1-cl^{-1})^{l-k}\sum_{j=1}^{k}C_k^j(cl^{-1})^j(1-cl^{-1})^{k-j}\\ &\geq e^{-c}\sum_{j=1}^{\infty}\frac{(ca)^j}{j!}=e^{-c}(e^{ac}-1).\end{aligned}$$

and obviously:

$$\mathbb{P}_k\left(Z(1)\geq k+1\right)=1-\mathbb{P}_k\left(Z(1)\leq k-1\right)-\mathbb{P}_k\left(Z(1)=k\right)\leq 1-e^{-c}(e^{ac}-1)-e^{-c}$$

The definition of a then implies that $\mathbb{P}_k\left(Z(1)\geq k+1\right)/\mathbb{P}_k\left(Z(1)\leq k-1\right)\leq 1$ and substituting into (23):

$$\mathbb{E}_k\left[\inf_{j\leq k-1}\nu_j\right]\leq\frac{2}{1-e^{-c}}+\sup_{k'\geq k+1}\mathbb{E}_{k'}\left[\inf_{j\leq k'-1}\nu_j\right]$$

An induction with respect to k starting from l and (22) establishes:

$$\mathbb{E}_k\left[\inf_{j\leq k-1}\nu_j\right]\leq(2(l-k)+1)\,\frac{1}{1-e^{-c}}$$

and then:

$$\mathbb{E}_k\left[\inf_{j\leq[al]}\nu_j\right]\leq\sum_{k'=[al]+1}^{k}\mathbb{E}_{k'}\left[\inf_{j\leq k'-1}\nu_j\right]\leq l^2\frac{(1-a)^2}{1-e^{-c}}$$

which yields the desired result.

Step 3. For any $k\geq 2$, $\lim_{l\to\infty}2^{-l}\mathbb{E}_k[\nu_1]=0$.

If $k\geq al$, then:

$$\mathbb{E}_k[\nu_1]\leq\mathbb{E}_{[al]}[\nu_1]+\mathbb{E}_k[\inf_{j<[al]}\nu_j].$$

From Step 2, it is sufficient to prove the statement for $k\leq[al]$. Obviously:

$$\mathbb{E}_k[\nu_1]\leq\sum_{j=1}^{k-1}\mathbb{E}_{j+1}[\nu_j].$$

By Step 1, and denoting $K=e^c/((1-a)c)$:

$$\mathbb{E}_k[\nu_1]\leq K2^l\sum_{j=1}^{k-1}(C_l^k)^{-1}\leq K2^l\sum_{1}^{l-1}(C_l^k)^{-1}\leq K2^{l+1}\sum_{1}^{[l/2]+1}(C_l^k)^{-1}.$$

For $k\leq[l/2]+1$, we have $k/(l-k+1)\leq 1$, so that $k\mapsto(C_l^k)^{-1}$ is a decreasing sequence. Consequently $(C_l^k)^{-1}\leq(C_l^2)^{-1}$ for $2\leq k\leq[l/2]+1$:

$$\mathbb{E}_k[\nu_1]\leq K2^{l+1}\left(\frac{1}{l}+\frac{2}{l(l-1)}\times([l/2]-1)\right)\leq K2^{l+1}l^{-1}$$

which establishes the desired result.

Step 4. $\lim_{l\to\infty} 2^{-l}\sum_{j=1}^{\infty}\frac{c^j e^{-c}}{j!}\mathbb{E}_j[\nu_0] = 1.$

Let us denote by p the probability distribution:

$$p(j) = C_l^j (cl^{-1})^j(1-cl^{-1})^{l-j} \text{ for } j \in \{0,...,l\}.$$

We still denote by Q the transition matrix of the chain Z. Conditioning with respect to the first jump, we have for any initial probability measure p_0 over $\{0,...,l\}$ [38]:

$$\mathbb{E}_{p_0}[\nu_0] = \sum_{i=1,j=0}^{l} p_0(i)Q_{i,j}(\mathbb{E}_j[\nu_0]+1) = \sum_{j=0}^{l}(p_0Q)(j)(\mathbb{E}_j[\nu_0]+1) - p_0(0)(\mathbb{E}_p[\nu_0]+1),$$

which also reads as:

$$\mathbb{E}_{p_0}[\nu_0] = (\mathbb{E}_{Qp_0}[\nu_0]+1) - p_0(0)(\mathbb{E}_p[\nu_0]+1). \tag{24}$$

Let p_0 be a probability measure over $\{0,...,l\}$. Defining inductively $p_{n+1} = p_nQ$, applying (24) inductively with respect to n and summing, we find that for any n:

$$\mathbb{E}_{p_n}[\nu_0] = \mathbb{E}_{p_0}[\nu_0] + n - \sum_{j=0}^{n-1} p_j(0)(\mathbb{E}_p[\nu_0]+1). \tag{25}$$

By the ergodic theorem for Markov chains [Rev75, ch. 6, theorem 2.8] p_n converges to the unique invariant probability measure μ as n goes to infinity. Dividing by n Eq. (25), and letting $n\to\infty$, we get that, for any l:

$$\mu(0)(\mathbb{E}_p[\nu_0]+1) = 1.$$

Since $\mu(0) = 2^{-l}$ and $p(j) \to \frac{c^j e^{-c}}{j!}$ as $l\to\infty$, we get the desired result.

Step 5. For any $j \geq 1$, $\lim_{l\to\infty} 2^{-l}\mathbb{E}_j[\nu_0] = \frac{1}{1-e^{-c}}.$

From the obvious estimate

$$\mathbb{E}_1[\nu_0] \leq \mathbb{E}_j[\nu_0] \tag{26}$$

which holds true for any $j \geq 1$, we can deduce by using Step 4 that:

$$\limsup_{l\to\infty} 2^{-l}\mathbb{E}_1[\nu_0] \leq \frac{1}{1-e^{-c}} \tag{27}$$

Applying the inequality

$$\mathbb{E}_j[\nu_0] \leq \mathbb{E}_1[\nu_0] + \mathbb{E}_j[\nu_1] \tag{28}$$

we get from Steps 3 and 4 that:

$$\liminf_{l\to\infty} 2^{-l}\mathbb{E}_1[\nu_0] \geq \frac{1}{1-e^{-c}} \tag{29}$$

Combining (27) and (29) establishes that:

$$\lim_{l\to\infty} 2^{-l}\mathbb{E}_1[\nu_0] = \frac{1}{1-e^{-c}}. \tag{30}$$

Substituting into (26) and (28) and using Step 3, the proof of the Lemma is complete. □

[38] The notation $\mathbb{E}_p$ is a shorthand for the expectation with respect to the statistical distribution of the chain Z starting from $Z(0)$ which obeys the distribution $\mathbb{P}_p(Z(0)=k) = p(k)$.

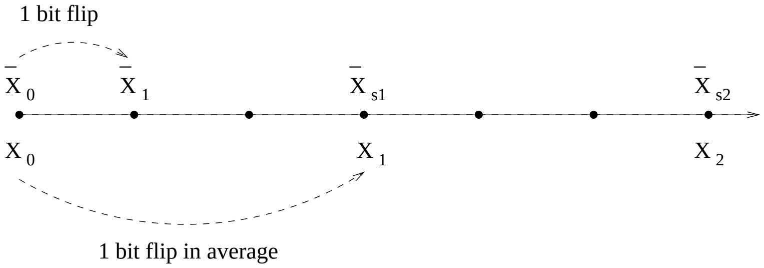


FIG 6 : The chain $X(n)$ jumps over some strings visited by the chain $\bar{X}(n)$.

Lemma I.2.11
If $k_f \geq 1$, then $2^{-l}(1-e^{-c})\nu_0$ converges in distribution to a random variable with exponential distribution and mean 1 as $l \to \infty$.

Proof. The proof of the convergence in distribution of $2^{-l}\nu_0$ is based on the very same arguments as those of the proof of Lemma I.2.9. □

I.2.3.4 Heuristic comparison between the chains X and $\bar{X}$

The statistical distributions of the chain $X(n)$ and of the corresponding stopping time T can actually be derived from the statistical distributions of the chain $\bar{X}(n)$ and of the corresponding stopping time $\bar{T}$. However the reader must be warned that this derivation is not as rigorous as the proofs of the previous Lemmas, in the sense that there is a gap between the statement of Proposition I.2.12 and its application. Nevertheless, the arguments presented in this section are rather intuitive, and we feel that they are worth noticing.
Let us introduce a sequence of independent and identically distributed random variables $(N_j)_{j\in\mathbb{N}}$ with Poisson distribution:

$$\mathbb{P}(N_1 = j) = \frac{c^j e^{-c}}{j!}, \tag{31}$$

and denote by S_n the partial sums:

$$S_n = \sum_{j=1}^{n} N_j.$$

The following proposition is based on the fact that the distribution of N_n is in the asymptotic framework $l \to \infty$ equivalent to the statistical distribution of the number of inverted bits during the n-th transition of the process X.

Proposition I.2.12
In the asymptotic framework $l \to \infty$ we have:

$$(X(n))_{n\in\mathbb{N}} \overset{law}{\simeq} (\bar{X}(S_n))_{n\in\mathbb{N}}.$$

Proof. Let X^0, X^1 be two strings of the state space E and denote $d := d(X^0, X^1)$. On the one hand, denoting $\bar{P}^j$ is the j-th power of the matrix $\bar{P}$:

$$\begin{aligned}
\mathbb{P}\left(\bar{X}(S_{n+1}) = X^1 \middle| \bar{X}(S_n) = X^0\right) &= \sum_{j=0}^{\infty} \mathbb{P}(N_{n+1} = j)\mathbb{P}\left(\bar{X}(j) = X^1 \middle| \bar{X}(0) = X^0\right) \\
&= \sum_{j=0}^{\infty} \frac{c^j e^{-c}}{j!}(\bar{P}^j)_{X^0,X^1} \\
&\overset{l\to\infty}{\simeq} c^d e^{-c} l^{-d},
\end{aligned}$$

because $(\bar{P}^d)_{X^0,X^1} = d!l^{-d}$ and $(\bar{P}^j)_{X^0,X^1} = O(l^{-d-1})$ for any $j \neq d$. On the other hand, :

$$\begin{aligned} \mathbb{P}\left(X(n+1) = X^1 | X(n) = X^0\right) &= (cl^{-1})^d(1-cl^{-1})^{l-d} \\ &\overset{l\to\infty}{\simeq} c^d e^{-c} l^{-d}. \end{aligned}$$

The comparison of these estimates implies the statement of the Lemma. □

In the following discussion, we shall apply the relation $X(.) = \bar{X}(S_.)$ without taking care of the small error terms. Although we do know that this not completely rigorous because the small error terms which are generated at every transition could cumulate and become non-negligible, we shall obtain by this way a nice interpretation of the relations between the expressions of the expected hitting times T and $\bar{T}$ of the chains X and $\bar{X}$ that have been proved in the above sections.

By the renewal theory (cf Appendix and (B.1)), since $\mathbb{E}[N_1] = c$, the speed rate of S_n is $c \times n$. But the chain $X(n)$ jumps over some values taken by the chain $\bar{X}(n)$. More exactly, when $\bar{X}(n)$ hits X^f for the first time at time ν_0, then X has some probability p_{miss} to miss X^f, if ν_0 is strictly between two successive times S_n, S_{n+1}. This probability can be computed by the renewal theory (cf Appendix and (B.2)) in the asymptotic framework $\nu_0 \gg 1$, which holds true when $l \gg 1$:

$$p_{miss} = 1 - \frac{\mathbb{P}(N_1 = 0)}{\mathbb{E}[N_1]} = 1 - \frac{1-e^{-c}}{c}.$$

If ν_0^j stands for the j-th hitting time of X^f by the chain $\bar{X}(n)$, then T is given by the first ν_0^j (divided by the speed rate c) which corresponds exactly to S_n for some n. The expectation of T can be split over the number J which is the first j when ν_0^j coincides with some S_n:

$$\mathbb{E}[T] = \frac{1}{c}\sum_{j=0}^{\infty} \mathbb{E}_{k_f}[\nu_0^j / J = j]\mathbb{P}(J = j).$$

By the strong Markov property, the excursions of the chain $\bar{X}(n)$ between two successive hitting times ν_0^j and ν_0^{j+1} are independent and identically distributed. Thus:

$$\mathbb{E}[T] = \frac{1}{c}\sum_{j=1}^{\infty} \left(\mathbb{E}_{k_f}[\nu_0] + \mathbb{E}_0[\bar{\nu}_0](j-1)\right)\mathbb{P}(J = j).$$

Since $\mathbb{P}(J = j) = p_{miss}^{j-1}(1-p_{miss})$ and $\mathbb{E}_0[\bar{\nu}_0] = 2^l$, we get that:

$$\mathbb{E}[T] = \frac{1}{c}\left(\mathbb{E}_{k_f}[\nu_0] + \frac{c-1+e^{-c}}{1-e^{-c}}\mathbb{E}_0[\bar{\nu}_0]\right) \overset{l\to\infty}{\simeq} 2^l \frac{1}{1-e^{-c}}.$$

<u>INTERPRETATION:</u> This heuristical approach shows that the difference in mutations efficiency, can be interpreted as a consequence of the coupling that exists between their trajectories in the search space. This coupling is illustrated in figure 6 and proved in proposition I.2.12. The respective trajectories coincide at instants that follow a Poisson distribution. Hence, the stochastic mutation jumps over some strings visited by the deterministic mutation. This means that the stochastic mutation is quicker in reaching new regions of the search space, but slower in reaching a particular target point. This suggests that these mutations are more complementary than concurrent, as their exploration properties are useful in different contexts.

I.2.4 MUTATION AND SELECTION : the (1+1)-ES

As recalled in the chapter's introduction, there has been some work about the expected complexity of some simple evolutionary algorithms on the OneMax function:
Bäck [Bäc92] derived the transition probabilities of the (1+1)-ES for the population Markov chain on the OneMax problem for the stochastic $\frac{c}{l}$ mutation. In the same framework, M "uhlenbein [M$\ddot{9}$2] considers a simpler Markov chain that has (provable) worse performance. He gives an upper bound for convergence time, and proves that this bound is minimal for the stochastic $\frac{1}{l}$ mutation, equal to $l\log(l)$. Bäck [Bäc93] numerically approximates the expected absorption time of the transition matrix, and validates the $l\log(l)$ bound as a close approximation of the true behavior. Rudolph [Rud97b] widely comments on these studies, as well as on the validity of the approximations. He also proves a $l\log(l)$ upper bound for the (1+1)-ES using the deterministic 1-bit-flip mutation.
Recently, Droste & al. [DJW98] were able to improve on those results by actually showing that these upper bounds are reached: they prove that the complexity of the (1+1)-ES with stochastic $\frac{c}{l}$ mutation is indeed in $O(l\log(l))$. Moreover, they generalize this result to any linear binary function.

However, the above mentioned works cannot be used to actually compare the stochastic and the deterministic bit flipping mutation operators, as they provide order of magnitude results without expliciting the multiplicative constants. The purpose of this section is to achieve such comparison by actually giving more accurate expansions of the convergence times as l goes to infinity. In the same way as in section I.2.3, two processes are compared:

Process definition. Let us consider first the process $\bar{X}$ starting from X^0 whose evolution is governed by the following way. At each time n, you choose a random integer uniformly over $\{1, ..., l\}$. Then you invert the corresponding bit of the string $\bar{X}(n-1)$. The resulting string is denoted by $\tilde{X}(n-1)$. If $d(\tilde{X}(n-1), X^f) < d(\bar{X}(n-1), X^f)$, then $\bar{X}(n) = \tilde{X}(n-1)$, otherwise $\bar{X}(n) = \bar{X}(n-1)$. The transition matrix of the chain $\bar{X}(n)$ is consequently:

$$\bar{P}_{X,X'} = \begin{cases} 1 - d(X, X^f)l^{-1} & \text{if } d(X, X') = 0 \\ l^{-1} & \text{if } d(X, X') = 1 \text{ and } d(X', X^f) = d(X, X^f) - 1 \\ 0 & \text{otherwise} \end{cases}$$

$\bar{X}$ corresponds to a (1+1)-ES on a OneMax problem with a deterministic 1-bit-flip mutation per individual. The process X is defined analogously, but with the stochastic c/l mutation.

Goal of the study. We look for the statistical distributions of the respective first hitting times T and $\bar{T}$ of the point X^f by the chains $X(n)$ and $\bar{X}(n)$.

I.2.4.1 Main results

Proposition I.2.13
1. Assuming that the starting point X^0 is such that $d(X^0, X^f) = k$, then the expectation of $\bar{T}$ is:

$$\mathbb{E}_k[\bar{T}] = l \sum_{j=1}^{k} j^{-1} \overset{l \geq k \gg 1}{\simeq} l(\ln k + C)$$

C is the Euler's constant whose value is $C \simeq 0.58$.

2. Assuming that the starting point X^0 is chosen uniformly over the state space $E = \{0,1\}^l$, the normalized first hitting time $\bar{T}/(l \ln l)$ converges in probability to 1 as $l \to \infty$.
For $l \gg 1$, the expectation of $\bar{T}$ can be expanded as:

$$\mathbb{E}[\bar{T}] \overset{l \gg 1}{\simeq} l(\ln l + C - \ln 2)$$

up to terms of order $o(l)$, while the normalized variance obeys:

$$\bar{V} := \frac{\sqrt{\mathbb{E}[\bar{T}^2] - \mathbb{E}[\bar{T}]^2}}{\mathbb{E}[\bar{T}]} \overset{l \gg 1}{\simeq} \frac{\pi}{\sqrt{6}(\ln l + C - \ln 2)}$$

The convergence rate of the normalized variance towards 0 is about $(\ln l)^{-1}$, which proves rigorously the convergence in probability of $\bar{T}/(l \ln l)$ towards 1, but this convergence rate is rather slow for practical purposes. The normalized variance is actually of order 1 for a large band of values of l. In order to get $\bar{V}_\mu < 20\%$ (resp. $\bar{V}_\mu < 10\%$), it is necessary that $l > 7.10^2$ (resp. $l > 4.10^5$). We can actually be much more precise and give a very accurate description for the statistical distribution of $\bar{T}$ in the asymptotic framework $l \to \infty$.

Proposition I.2.14
Assuming that the starting point X^0 is chosen uniformly over the state space $E = \{0,1\}^l$, the normalized first hitting time $(\bar{T} - l \ln l)/l$ converges in distribution as $l \to \infty$ to $-\ln 2 - \ln Z$, where Z is a random variable with an exponential density $p(z) = \mathbb{I}_{z \geq 0} e^{-z}$.

Note that we get back the results of Proposition I.2.13 since $\mathbb{E}[-\ln Z] = C$ and $\mathbb{E}[(\ln Z)^2] = \pi^2/6 + C^2$, where C is the Euler's constant.

Corollary I.2.15
Assume that we deal with a set of N processes $\bar{X}_i$ which evolve independently. The global first hitting time $\bar{T}$ is the minimum of the N independent hitting times.
1. If N is fixed (independent of l), then there exists a random variable Z_N with probability density $p_N(z) = N(1-e^{-z})^{N-1} e^{-z} \mathbb{I}_{z \geq 0}$ and mean $\sum_{j=1}^N j^{-1}$ such that the global first hitting time $\bar{T}$ satisfies:

$$\frac{\bar{T} - l \ln l}{l} \overset{l \to \infty}{\longrightarrow} -\ln 2 - \ln Z_N. \tag{32}$$

2. If $N = N(l)$ grows sub-exponentially to infinity as $l \to \infty$ (more exactly if $N(l) \to \infty$ and $\frac{\ln \ln N(l)}{\ln l / \ln \ln l} \to 0$ as $l \to \infty$), then

$$\frac{\bar{T} - l \ln l + l \ln \ln N(l)}{l} \overset{l \to \infty}{\longrightarrow} -\ln 2. \tag{33}$$

with probability 1.

This corollary means in particular that the gain when using N independent processes is very weak, of the order of $\ln \ln N$. To get a significant gain, it is necessary to deal with an exponentially large (as a function of l) number of processes.

Let us now consider the process X starting from X^0 whose evolution is governed by the following way. At each time n, you build a string $\tilde{X}(n-1)$ by inverting each bit of $X(n-1)$ independently with probability c/l. If $d(\tilde{X}(n-1), X^f) < d(X(n-1), X^f)$, then

$X(n) = \tilde{X}(n-1)$, otherwise $X(n) = X(n-1)$. The transition matrix of the chain $X(n)$ is consequently:

$$P_{X,X'} = \begin{cases} P_{X,X} & \text{if } d(X,X') = 0 \\ (cl^{-1})^{d(X,X')} \left(1 - cl^{-1}\right)^{l-d(X,X')} & \text{if } d(X',X^f) < d(X,X^f) \\ 0 & \text{otherwise} \end{cases}$$

with $P_{X,X} = 1 - \sum_{k=0}^{d(X,X^f)} \sum_{k'=0}^{l-d(X,X^f)} \mathbb{I}_{k<k'} C^k_{d(X,X^f)} C^{k'}_{l-d(X,X^f)} (cl^{-1})^{k+k'} \left(1 - cl^{-1}\right)^{l-k-k'}$.

Proposition I.2.16
1. Assuming that the starting point X^0 is such that $d(X^0, X^f) = k$, the normalized first hitting time $T/(l \ln k)$ converges in probability to e^c/c as $l \to \infty$.

2. Assuming that the starting point X^0 is chosen uniformly over the state space $\{0,1\}^l$, the normalized first hitting time $T/(l \ln l)$ converges in probability to e^c/c as $l \to \infty$.
For $l \gg 1$, the expectation of T can be expanded as:

$$\mathbb{E}[T] \stackrel{l \gg 1}{\simeq} l \left(\frac{e^c}{c} \ln l + R(c) \right)$$

up to terms of order $o(l)$. The function $c \mapsto R(c)$ is plotted in Figure 7. In particular $R(1) \simeq -1.90$.

Note that the function $c \mapsto c^{-1}e^c$ is minimal at $c = 1$. The mutation rate l^{-1} is therefore optimal.

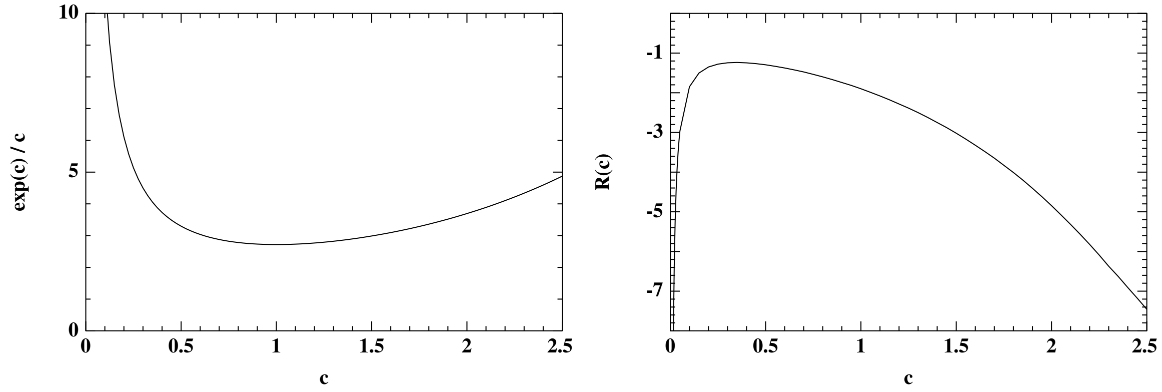


FIG 7 : *Functions $c \mapsto c^{-1}e^c$ and $c \mapsto R(c)$*

We can actually be much more precise and give a very accurate description for the statistical distribution of T in the asymptotic framework $l \to \infty$.

Proposition I.2.17
Assuming that the starting point X^0 is chosen uniformly over the state space $E = \{0,1\}^l$, the normalized first hitting time $(T - c^{-1}e^c l \ln l)/l$ converges in distribution as $l \to \infty$:

$$\frac{T - c^{-1}e^c l \ln l}{l} \stackrel{l \to \infty}{\longrightarrow} -\frac{e^c}{c}(\ln Z + C) + R(c),$$

where Z is a random variable with an exponential density $p(z) = \mathbb{I}_{z \geq 0} e^{-z}$, $C \simeq 0.58$ is the Euler's constant and $R(c)$ is plotted in Figure 7.

Corollary I.2.18
Assume that we deal with a set of N processes X_i which evolve independently. The global first hitting time T is the minimum of the N independent hitting times.
1. If N is fixed (independent of l), then there exists a random variable Z_N with probability density $p_N(z) = N(1-e^{-z})^{N-1}e^{-z}\mathbb{I}_{z\geq 0}$ and mean $\sum_{j=1}^{N} j^{-1}$ such that the global first hitting time $\bar{T}$ satisfies:

$$\frac{T - c^{-1}e^c l \ln l}{l} \overset{l\to\infty}{\longrightarrow} R(c) - \frac{e^c}{c}(\ln Z_N + C). \tag{34}$$

2. If $N = N(l)$ grows sub-exponentially to infinity as $l \to \infty$ (more exactly if $N(l) \to \infty$ and $\frac{\ln\ln N(l)}{\ln l/\ln\ln l} \to 0$ as $l \to \infty$), then

$$\frac{T - c^{-1}e^c l \ln l + c^{-1}e^c l \ln\ln N(l)}{l} \overset{l\to\infty}{\longrightarrow} R(c) - \frac{e^c}{c}C. \tag{35}$$

with probability 1.

Remark I.2.19 *As recalled in Remark I.2.1, the* 1-bit-flip *and the stochastic $\frac{c}{l}$ mutations require the same effort to reach their goal in the absence of selection. This is not the case here in the presence of selection: Convergence time of the stochastic* **active mutation** *is still a factor $\frac{e^c-1}{c}$ greater than the deterministic mutation in the case of the (1+1)-ES on the OneMax problem.*

I.2.4.2 Study of the chain $\bar{X}$

<u>**Proof of Proposition I.2.13**</u> Let us introduce the auxiliary process $Z(n)$ with state space $\{0, ..., l\}$ defined by:

$$Z(n) = d(\bar{X}(n), X^f). \tag{36}$$

Z is a Markov chain starting from $Z(0) = d(X^0, X^f)$ and whose jumps are governed by the transition matrix $\bar{Q}$:

$$\bar{Q} = \begin{pmatrix} 1 & 0 & \dots & \dots & \dots & 0 \\ \frac{1}{l} & 1-\frac{1}{l} & 0 & \dots & \dots & 0 \\ 0 & \frac{2}{l} & 1-\frac{2}{l} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1-\frac{1}{l} & \frac{1}{l} & 0 \\ 0 & \dots & \dots & 0 & 1 & 0 \end{pmatrix}$$

Obviously the first hitting time of the point 0 by $Z(n)$ corresponds to the first hitting time $\bar{T}$ of the point X^f by the chain $\bar{X}(n)$. Starting from k, the first hitting time of the point 0 by the chain Z is:

$$\bar{T} = \sum_{j=1}^{k} \tau_j, \tag{37}$$

where the τ_j's are the times necessary for the chain to jump from j to $j-1$. By the strong Markov property, the τ_j's are independent random variables with respective distributions ($Q^n_{j,j}$ is the n-th power of the coefficient $Q_{j,j}$):

$$\mathbb{P}(\tau_j = t) = (1 - Q_{j,j}){Q_{j,j}}^{t-1} = \frac{j}{l}\left(1 - \frac{j}{l}\right)^{t-1} \quad \text{for } t \in \mathbb{N}^*.$$

In particular their first and second order moments are:

$$\mathbb{E}[\tau_j] = \frac{l}{j}, \qquad \mathbb{E}[\tau_j^2] = \frac{2l^2 - jl}{j^2}$$

We can then deduce the expectation of $\bar{T}$:

$$\mathbb{E}_k[\bar{T}] = \sum_{j=1}^{k} \mathbb{E}[\tau_j] = l \sum_{j=1}^{k} j^{-1} \overset{k \gg 1}{\simeq} l(\ln k + C),$$

up to terms of order $o(l)$. If the chain $\bar{X}$ starts from a point X^0 which is chosen uniformly over the state space $\{0,1\}$, then the chain Z starts from the point k_0 which obeys the distribution μ:

$$\mu(k_0) = 2^{-l} C_l^{k_0}. \tag{38}$$

and then:

$$\mathbb{E}_\mu[\bar{T}] = \sum_{k=0}^{l} \mu(k) \mathbb{E}_k[\bar{T}].$$

Using Stirling's formula [AS65](see Appendix):

$$\mathbb{E}_\mu[\bar{T}] \overset{l \gg 1}{\simeq} l \left(\ln l + C + \sqrt{l} \int_0^1 e^{-lf(a)} \frac{\ln(a)}{\sqrt{2\pi a(1-a)}} da \right),$$

where $f(a) = \ln 2 + a \ln(a) + (1-a) \ln(1-a)$. Since f is minimal at $a = 1/2$, $f(1/2) = 0$, $f'(1/2) = 0$ and $f''(1/2) = 4$, we get by using Laplace technique that:

$$\mathbb{E}_\mu[\bar{T}] \overset{l \gg 1}{\simeq} l \left(\ln l + C - \ln 2 \right)$$

up to terms of order $o(l)$. The calculation of the second order moment of $\bar{T}$ can be carried out by the same way:

$$\begin{aligned}
\mathbb{E}_k[\bar{T}^2] &= \sum_{j=1}^{k} \mathbb{E}[\tau_j^2] + \sum_{i \neq j = 1}^{k} \mathbb{E}[\tau_i]\mathbb{E}[\tau_j] = l^2 \left(\sum_{j=1}^{k} j^{-2} \right) - l \left(\sum_{j=1}^{k} j^{-1} \right) + l^2 \left(\sum_{j=1}^{k} j^{-1} \right)^2 \\
&\overset{l \gg 1}{\simeq} l^2 \left(\frac{\pi^2}{6} + C^2 + (\ln k)^2 + 2C \ln k \right) - l(\ln k + C).
\end{aligned}$$

So the normalized variance defined by:

$$\bar{V}_k := \frac{\sqrt{\mathbb{E}_k[\bar{T}^2] - \mathbb{E}_k[\bar{T}]^2}}{\mathbb{E}_k[\bar{T}]}$$

satisfies:

$$\bar{V}_k \overset{l \gg 1}{\simeq} \frac{\pi}{\sqrt{6}(\ln k + C)}.$$

Starting from a random point point X^0 chosen uniformly over the state space E, we have using (38):

$$\mathbb{E}_\mu[\bar{T}^2] \overset{l \gg 1}{\simeq} l^2 \left\{ (\ln l)^2 + (2C - 2\ln 2) \ln l + \left(\frac{\pi^2}{6} + C^2 + (\ln 2)^2 - 2C \ln 2 \right) \right\}$$

up to terms of order $o(l^2)$. As a consequence the normalized variance defined by:

$$\bar{V}_\mu := \frac{\sqrt{\mathbb{E}_\mu[\bar{T}^2] - \mathbb{E}_\mu[\bar{T}]^2}}{\mathbb{E}_\mu[\bar{T}]}$$

satisfies:

$$\bar{V}_\mu \overset{l \gg 1}{\simeq} \frac{\pi}{\sqrt{6}(\ln l + C - \ln 2)}$$

which goes to 0 as $l \to \infty$ at rate $(\ln l)^{-1}$, which implies that $\bar{T}/(l \ln l)$ converges in probability to 1.

Remark I.2.20
From Eq.(37) we can deduce a closed form expression for the histogram of $\bar{T}$, which holds true for any l. Unfortunately these expressions are all the more complicated as l is large.

Proposition I.2.21
Let us denote $p_j(t) = \frac{j}{l}\left(1 - \frac{j}{l}\right)^{t-1}$ *for* $t \in \mathbb{N}^*$ *and* $j = 1, ..., l$.
1. Assuming that the starting point X^0 *is such that* $d(X^0, X^f) = k$*, then:*

$$\mathbb{P}_k(\bar{T} = t) = p_1 \star ... \star p_k(t),$$

where $p \star q(t) = \sum_{t'=1}^{t-1} p(t')q(t-t')$.
2. Assuming that the starting point X^0 *is chosen uniformly over the state space* $E = \{0,1\}^l$*, then:*

$$\mathbb{P}(\bar{T} = t) = \sum_{k=1}^{l} 2^{-l} C_l^k \mathbb{P}_k(\bar{T} = t)$$

In Figure 8 are plotted the theoretical histograms of $\bar{T}$ under the assumption that the starting point X^0 is chosen uniformly over the state space E for different values of l.

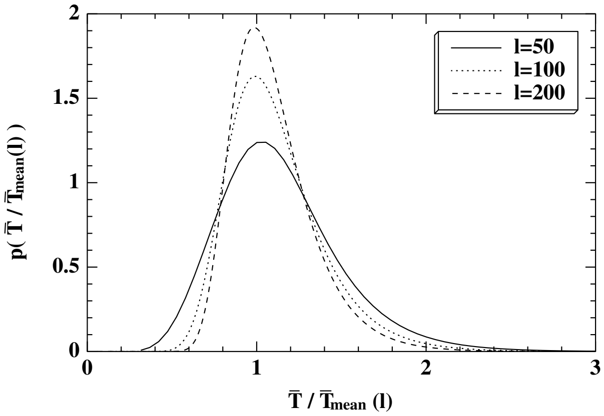


FIG 8 : *Theoretical histograms of* $\bar{T}$*. The scales are normalized by the quantity* $\bar{T}_{mean}(l) := l(\ln l + C - \ln 2)$.

Proof of Proposition I.2.14 Let us denote by T_r^l the random variable which depends on l:

$$T_r^l := \frac{\bar{T} - l \ln l + l \ln 2}{l}$$

Let α be a real number larger than -1. From the representation (37), the Laplace transform of T_r^l is:

$$\mathbb{E}_k[e^{-\alpha T_r^l}] = e^{\alpha(\ln 2 - \ln l)} \mathbb{E}\left[e^{-\alpha(\tau_1 + \ldots + \tau_k)/l}\right]$$

Since the τ_j's are independent:

$$\mathbb{E}_k[e^{-\alpha T_r^l}] = e^{-\alpha(\ln 2 - \ln l)} \prod_{j=1}^{k} \mathbb{E}\left[e^{-\alpha \tau_j / l}\right]$$

Computing the Laplace transforms of the variables τ_j's whose distributions are geometric is easy:

$$\mathbb{E}\left[e^{-\alpha' \tau_j}\right] = e^{-\alpha'} \left(\frac{l}{j}(1 - e^{-\alpha'}) + e^{-\alpha'}\right)^{-1}$$

so that (with $\alpha' = \alpha/l$, we have $1 - e^{-\alpha'} \simeq \alpha/l$):

$$\mathbb{E}_k[e^{-\alpha T_r^l}] = e^{-\alpha(\ln 2 - \ln l)} e^{-k\alpha/l} \left[\prod_{j=1}^{k} \left(1 + \frac{\alpha}{j} - \frac{\alpha}{l}\right)\right]^{-1} \tag{39}$$

up to terms of order $O(l^{-1})$. We first deal with the product (multiplied by $k!$):

$$\begin{aligned}
\prod_{j=1}^{k} \left(j + \alpha(1 - \frac{j}{l})\right) &= \prod_{j=1}^{k} (j + \alpha) \prod_{j=1}^{k} \left(1 - \frac{1}{l} \frac{\alpha j}{j + \alpha}\right) \\
&= \frac{\Gamma(k + \alpha + 1)}{\Gamma(\alpha + 1)} \exp\left(\sum_{j=1}^{k} \ln\left(1 - \frac{\alpha}{l} \frac{j}{j + \alpha}\right)\right)
\end{aligned}$$

where Γ is the Euler's Gamma function $\Gamma(x) := \int_0^\infty t^{x-1} e^{-t} dt$. In the asymptotic $l \to \infty$ the sum inside the exponential is equivalent to:

$$\sum_{j=1}^{k} \ln\left(1 - \frac{\alpha}{l} \frac{j}{j + \alpha}\right) \overset{l \gg 1}{\simeq} -\frac{\alpha}{l} \sum_{j=1}^{k} \frac{j}{j + \alpha} \overset{l \gg 1}{\simeq} -\frac{k}{l}\alpha,$$

up to terms of order $O(kl^{-2})$. Substitution into (39) then yields that:

$$\mathbb{E}_k[e^{-\alpha T_r^l}] \overset{l \to \infty}{\longrightarrow} 2^{-\alpha} l^{-\alpha} \frac{\Gamma(k + 1)\Gamma(\alpha + 1)}{\Gamma(k + \alpha + 1)}$$

Since $\mathbb{E}_\mu[.] = \sum_{k=0}^{l} 2^{-l} C_l^k \mathbb{E}_k[.]$ and $\Gamma(k + 1)/\Gamma(k + \alpha + 1) \simeq k^{-\alpha}$ for $k \gg 1$, we obtain:

$$\mathbb{E}_\mu[e^{-\alpha T_r^l}] \overset{l \to \infty}{\longrightarrow} \Gamma(\alpha + 1) \tag{40}$$

for any $\alpha > 0$. This proves the convergence in distribution of T_r^l as $l \to \infty$ to the unique variable T_r such that $\mathbb{E}[e^{-\alpha T_r}] = \Gamma(\alpha + 1)$. It is easy to identify this variable in terms of a random variable Z with exponential distribution and mean 1: $T_r = -\ln Z$.

Proof of Corollary I.2.15 Since $\bar{T} = \inf_{i=1,\dots,N} \bar{T}_i$ and the $\bar{T}_i$ are independent and obey the same distribution:

$$\mathbb{P}\left(\bar{T} \geq \bar{t}\right) = \mathbb{P}\left(\bar{T}_1 \geq \bar{t}\right)^N \tag{41}$$

Choosing $\bar{t} = l\ln l - l\ln 2 + lt$ and using Proposition I.2.14:

$$\mathbb{P}\left(\bar{T}_1 \geq \bar{t}\right) \overset{l\to\infty}{\longrightarrow} \mathbb{P}(-\ln Z_1 \geq t) = \mathbb{P}(Z_1 \leq e^{-t})$$

where Z_1 is an exponential variable with mean 1, so that:

$$\mathbb{P}\left(\bar{T}_1 \geq \bar{t}\right) \overset{l\to\infty}{\longrightarrow} 1 - \exp(-e^{-t})$$

If N is independent of l, substituting into (41) establishes:

$$\mathbb{P}\left(\frac{\bar{T} - l\ln l + l\ln 2}{l} \geq t\right) \overset{l\to\infty}{\longrightarrow} \left(1 - \exp(-e^{-t})\right)^N .$$

Therefore the convergence (32) holds true if Z_N is the random variable whose repartition function is:

$$\mathbb{P}(Z_N \leq z) = (1 - \exp(-z))^N$$

It means that Z_N obeys the statistical distribution of the first hitting time of N independent random variables with exponential density and mean 1. This proves the first statement of the corollary.

Let us turn to the case $N = N(l)$. Denoting

$$T^l_{r,1} := \frac{\bar{T}_1 - l\ln l + l\ln 2}{l}$$

and following carefully the proof of Proposition I.2.14 and the error terms, it can be shown that (40) still holds true for large α, but smaller than l:

$$\mathbb{E}[e^{-\alpha T^l_{r,1}}] = \Gamma(\alpha + 1) \text{ if } 1 \ll \alpha \ll l.$$

Using a Tauberian theorem (Theorem XII-5-4-[Fel71] which establishes connections between the Laplace transform and the repartition function of a random variable), we then get that, if $t \ll \Gamma^{-1}(l)$:

$$\mathbb{P}\left(T^l_{r,1} \leq t\right) \overset{l\gg 1}{\simeq} \exp\left(-e^{-t}\right)$$

Since $\Gamma^{-1}(l) \simeq \ln l/\ln\ln l$, substituting into (41) establishes that, for any $t \ll \ln l/\ln\ln l$:

$$\mathbb{P}\left(\frac{\bar{T} - l\ln l + l\ln 2}{l} \geq t\right) \overset{l\gg 1}{\simeq} \left(1 - \exp\left(-e^{-t}\right)\right)^{N(l)}$$

Let $\delta > 0$. Choosing $t = -\ln[(1+\delta)\ln N(l)]$ (which is smaller than $\ln l/\ln\ln l$ by assumption), we have $e^{-t} = N(l)^{1+\delta}$ so that:

$$\mathbb{P}\left(\frac{\bar{T} - l\ln l + l\ln 2}{l} \geq t\right) \overset{l\gg 1}{\simeq} \left(1 - \frac{1}{N(l)^{1+\delta}}\right)^{N(l)} \overset{l\gg 1}{\simeq} 1$$

since $N(l) \to \infty$. Choosing $t = -\ln[(1-\delta)\ln N(l)]$, we have $e^{-t} = N(l)^{1-\delta}$ so that:

$$\mathbb{P}\left(\frac{\bar{T} - l\ln l + l\ln 2}{l} \geq t\right) \overset{l\gg 1}{\simeq} \left(1 - \frac{1}{N(l)^{1-\delta}}\right)^{N(l)} \overset{l\gg 1}{\simeq} 0$$

which completes the proof of the Corollary.

I.2.4.3 Study of the chain X

Proof of Proposition I.2.16 The strategy of the proof is similar to that of Proposition I.2.13. However the details are a little more complicated because the transition matrix of the chain $X(n)$ is more complicated than the one of the chain $\bar{X}(n)$. Let us introduce the auxiliary process $Z(n)$ with state space $\{0,...,l\}$ defined by:

$$Z(n) = d(X(n), X^f). \tag{42}$$

Z is a Markov chain starting from $Z(0) = d(X^0, X^f)$. The coefficients of the corresponding transition matrix Q are given by:

$$Q_{j,j-j'} = \begin{cases} \displaystyle\sum_{k=0}^{(l-j)\wedge(j-j')} C_j^{k+j'} C_{l-j}^k (cl^{-1})^{2k+j'}(1-cl^{-1})^{l-2k-j'} & \text{if } j' \in \{1,...,j\} \\ Q_{j,j} & \text{if } j' = 0 \\ 0 & \text{if } j' \notin \{0,...,j\} \end{cases} \tag{43}$$

with $Q_{j,j} = 1 - \sum_{j'=1}^{j} Q_{j,j-j'}$.
Let us assume that the chain Z arrives at some point j at some time t. We then denote by τ_j the waiting time for the next jump, and by N_j the length of this jump. By the strong Markov property, the random variables τ_j and N_j are independent and their respective statistical distributions are given by:

$$\mathbb{P}(\tau_j = t) = (1 - Q_{j,j})Q_{j,j}^{t-1} \text{ for } t \in \mathbb{N}^*, \tag{44}$$

$$\mathbb{P}(N_j = n) = \frac{Q_{j,j-n}}{1 - Q_{j,j}} \text{ for } n = 1,...,j. \tag{45}$$

The statistical distribution of the first hitting time T can be expressed in terms of the variables N_j and τ_j. In particular, conditioning with respect to the first jump yields:

$$\mathbb{P}_j(T = t) = \sum_{t'=1}^{t}\sum_{n=1}^{j} \mathbb{P}_j\left(T = t/\tau_j = t', N_j = n\right)\mathbb{P}(\tau_j = t')\mathbb{P}(N_j = n)$$

Since the weak Markov property insures that:

$$\mathbb{P}_j\left(T = t/\tau_j = t', N_j = n\right) = \mathbb{P}_{j-n}\left(T = t - t'\right),$$

we get the inductive relation:

$$\mathbb{P}_j(T = t) = \sum_{t'=1}^{t}\sum_{n=1}^{j} \mathbb{P}_{j-n}\left(T = t - t'\right)\mathbb{P}(\tau_j = t')\mathbb{P}(N_j = n) \tag{46}$$

Indeed, iterating this formula establishes the following closed-form expression which holds true for any k and l:

$$\mathbb{P}_k(T = t) = \sum_{J=1}^{k}\sum_{n_1+...+n_J=k}\prod_{i=1}^{J}\mathbb{P}(N_{k-n_1-...-n_{i-1}} = n_i)p_k \star p_{k-n_1} \star ... \star p_{k-n_k-...n_J}(t). \tag{47}$$

where $p \star q(t) = \sum_{t'=1}^{t-1} p(t')q(t-t')$. If we assume that the starting point X^0 is chosen uniformly over the state space $\{0,1\}^l$, then we can express the histogram of T as follows:

$$\mathbb{P}(T = t) = \sum_{k=0}^{l} 2^{-l} C_l^k \mathbb{P}_k(T = t) \tag{48}$$

We now turn to the study of the expectation of T in order to exhibit its behavior for large values of l. Multiplying (46) by t and summing over t yields: the inductive relation:

$$\mathbb{E}_j[T] = \mathbb{E}[\tau_j] + \sum_{n=1}^{j} \mathbb{E}_{j-n}[T]\mathbb{P}(N_j = n) \tag{49}$$

Iterating with respect to t establishes the following closed-form expression which holds true for any k and l:

$$\mathbb{E}_k[T] = \sum_{J=1}^{k} \sum_{n_1+\ldots+n_J=k} \left(\sum_{i=0}^{J-1} \mathbb{E}[\tau_{k-n_1-\ldots-n_i}] \right) \prod_{i=1}^{J} \mathbb{P}(N_{k-n_1-\ldots-n_{i-1}} = n_i). \tag{50}$$

In the asymptotic framework $l \gg 1$, this complicated expression gets much simpler.
Step 1. There exists functions f_τ and f_N which do not depend on l such that, for $l \gg 1$: $\mathbb{E}[\tau_j] = f_{\tau,c}(j/l)$ and $\mathbb{E}[N_j] = f_{N,c}(j/l)$.

$$\begin{aligned} \frac{1}{f_{\tau,c}(a)} &= 1 - e^{-c} \sum_{k=0}^{\infty} \frac{(c(1-a))^k}{k!} \left(\sum_{j=0}^{k} \frac{(ca)^j}{j!} \right), & (51)\\ f_{N,c}(a) &= c(2a-1) + \frac{c^2(1-a)a \left(\sum_{k=0}^{\infty} \frac{(c^2(1-a)a)^k}{k!(k+1)!} \right) + ca \left(\sum_{k=0}^{\infty} \frac{(c^2(1-a)a)^k}{k!^2} \right)}{e^c f_{c,\tau}(a)} & (52) \end{aligned}$$

This step results from the long but straightforward calculations of the expectations of N_j and τ_j whose statistical distributions are given by (43)-(45).

Step 2. If $a \in (0,1]$, then we have for $l \gg 1$: $\mathbb{E}_{[al]}[T] = \sum_{j=1}^{[al]} \frac{f_{\tau,c}(j/l)}{f_{N,c}(j/l)}$.

Let $0 < \delta \ll 1$, such that $1 \ll \delta l \ll l$. By Step 1, we get that over the interval $j \in \{[al], ..., [al] - [\delta l]\}$, the statistical distributions of N_j and τ_j are almost identical to the ones of $N_{[al]}$ and $\tau_{[al]}$. Using the inductive relation (49):

$$\mathbb{E}_{[al]}[T] \simeq \sum_{k=1}^{\infty} \sum_{n_1+\ldots+n_{k-1} \le [\delta l] < n_1+\ldots+n_k} \prod_{j=1}^{k} \mathbb{P}(N_{[al]} = n_j) \left(k\mathbb{E}[\tau_{[al]}] + \mathbb{E}_{[al]-n_1+\ldots+n_k}[T] \right)$$

By the renewal theory (see Appendix and (B.1)):

$$\frac{1}{[\delta l]} \sum_{k=1}^{\infty} k\mathbb{P}\left(N_1 + \ldots + N_{k-1} \le [\delta l] < N_1 + \ldots + N_k\right) \overset{l\to\infty}{\simeq} \frac{1}{\mathbb{E}[N_{[al]}]} = \frac{1}{f_{N,c}(a)},$$

which implies that:

$$\mathbb{E}_{[al]}[T] \simeq \frac{f_{\tau,c}(a)}{f_{N,c}(a)}[\delta l] + \mathbb{E}_{[al]-[\delta l]}[T].$$

Iterating this process completes the proof of Step 2.
Step 3. Computation of $\mathbb{E}_\mu[T]$ for $l \gg 1$.
The function $a \mapsto \frac{f_{c,\tau}(a)}{f_{c,N}(a)}$ has a singularity near the point $a = 0$ of the type a^{-1}. Nevertheless $g_c := a \mapsto \frac{f_{c,\tau}(a)}{f_{c,N}(a)} - \frac{e^c}{ca}$ is a smooth function. As a consequence:

$$\mathbb{E}_{[al]}[T] = \sum_{j=1}^{[al]} \frac{e^c l}{cj} + \sum_{j=1}^{[al]} g_c(j/l)$$

which yields:

$$\mathbb{E}_{[al]}[T] = l\left(c^{-1}e^c \ln l + c^{-1}e^c \ln a + c^{-1}e^c C + G_c(a)\right)$$

where G_c is the integral of g_c:

$$G_c(a) = \int_0^a g_c(s)ds.$$

As stated in the proof of Proposition I.2.13, if the chain X starts from a point X^0 which is chosen uniformly over the state space $\{0,1\}$, then the chain Z starts from the point k_0 which obeys the distribution (38) so that:

$$\mathbb{E}_\mu[T] = \sum_{k=0}^{l} \mu(k)\mathbb{E}_k[T].$$

Using Stirling's formula [AS65] to expand $\mu(k)$:

$$\mathbb{E}_\mu[T] = l\left(c^{-1}e^c \ln l + c^{-1}e^c C + eC + \sqrt{l}\int_0^1 e^{-lf(a)} \frac{c^{-1}e^c \ln a + G_c(a)}{\sqrt{2\pi a(1-a)}} da\right).$$

where $f(a) = \ln 2 + a\ln(a) + (1-a)\ln(1-a)$. Using Laplace technique to compute the integral then establish the result by choosing $R(c) = -c^{-1}e^c \ln 2 + c^{-1}e^c C + G_c(1/2)$.

Step 4. Conclusion.

Adopting the same strategy as in the proof of Proposition I.2.13, long but straightforward calculations establish that:

$$\frac{\sqrt{\mathbb{E}[T^2] - \mathbb{E}[T]^2}}{\mathbb{E}[T]} \stackrel{l\to\infty}{\longrightarrow} 0$$

at rate $(\ln l)^{-1}$, which implies that $T/(l \ln l)$ converges in probability to e.

Remark I.2.22

In the above proof we have established a closed form expression (47-48) for the histogram of T which holds true for any l. This exact expression is unfortunately not very useful, because very complicated. Nevertheless a computer can deal with it whenever l does not exceed a few hundreds. In Figure 9 are plotted the theoretical histograms of $\bar{T}$ for different values of l.

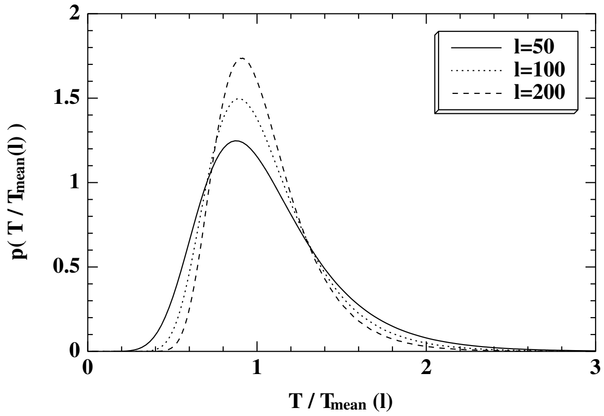


FIG 9 : *Theoretical histograms of T in the case $c = 1$, which corresponds to the (1+1)-ES with mutation rate $1/l$. The scales are normalized by the quantity $T_{mean}(l) := l(e\ln l + R(1))$, $R(1) \simeq -1.90$.*

Proofs of Proposition I.2.17 and Corollary I.2.18 Although the calculations are a little more complicated, the proofs are exactly the same as the ones of Proposition I.2.14 and Corollary I.2.15 respectively.

I.2.5 DISCUSSION AND FURTHER DIRECTIONS

A first outcome of the results of this paper is a clear advantage of the deterministic 1-bit-flip mutation over the stochastic $\frac{c}{l}$ mutation, both in the absence of selection, and when using the (1+1)-ES selection on the OneMax problem.
However, this paper does not claim that the deterministic bit flipping mutation should be preferred to the stochastic bit flipping mutation, as it is clear that the deterministic 1-bit-flip mutation with non linear functions cannot avoid any local optimum. Rather, we point out the fact that a balance should be found between the ability of a mutation to escape from local optima, and its efficiency in terms of convergence speed. The results of this paper suggest that the positive[39] property of mutations can result in a slower convergence: The stochastic c/l mutations are able to escape from any local optima as they are positive operators; but, on both flat and linear landscapes, they are slower than the simple deterministic 1-bit-flip mutation.

Another point to consider, when trying to analyze these theoretical results in the light of practical work, is that the hitting times derived for both mutations were computed in terms of number of mutations performed before reaching the target point of the search space. In the framework of Evolutionary Optimization, a common practice is to consider that the main source of CPU consumption of an Evolutionary Algorithm comes from the computations of the fitnesses. In that perspective (also considered in [DJW98]), a mutation that does not flip any bits does not cost any supplementary fitness evaluation. As was recalled in section I.2.1, this happens e^{-c} of the times with the stochastic $\frac{c}{l}$ mutation. So considering the number of fitness computations instead of the number of steps, the deterministic and the stochastic bit flipping mutations require the same effort to reach their goal in the absence of selection. On the other hand, there is still a factor $\frac{e^c-1}{c}$ greater than 1 in favor of the stochastic mutation in the case of the (1+1)-ES on the OneMax problem. Note that this factor goes to 1 when c goes to 0: stochastic mutation then becomes as slow as the deterministic 1-bit-flip mutation. Indeed, the process spends e^c of the time doing nothing, ce^c mutating one bit, and the remaining time $(1 - e^c - ce^c)$ mutating more than one bit. Neglecting the steps where nothing happens, the process spends $\frac{ce^{-c}}{1-e^{-c}} \simeq 1 - \frac{c}{2}$ $[c \to 0]$ mutating exactly one bit, and therefore looks like the deterministic 1-bit-flip mutation. But this is true only in terms of fitness evaluations, as most efforts are then devoted to generate random numbers and not doing anything else.

As already mentioned in section I.2.1, some practitioners of Evolutionary Algorithms actually use inhomogeneous decreasing schemes for the mutation rate (see also the theoretical results of [Cer96]). Indeed, there might be some benefit in starting with large mutation and decreasing the mutation strength to the deterministic 1-bit-flip mutation in the final generations. On the other hand, if the landscape can be locally seen as a OneMax-like problem to the nearest local optima, then, the deterministic bit flipping mutation will be quicker than the stochastic mutation to reach that local optima.
Another way to pursue this work in the same line would be to study a mixed mutation, identical to the stochastic mutation, but ensuring that at least one bit of the bitstring is flipped. And of course, the ultimate step along that line would be to study some self-adaptive

[39] a mutation that links any two points of the space with non-zero probability

mutation strength, as the one proposed in [Oba94, BS96].

On the other hand, the stochastic mutation certainly is the most widely used mutation operator on the binary search space. But it should be noted that, in the binary framework, one generally uses a GA-like algorithm, involving another type of selection, and considering the crossover operator to be the main operator: the mutation operator is considered a background operator in the GA literature [Hol75, Gol89]. Thus, any generalization of the tendencies proved in the (1+1)-ES framework is hazardous.

A natural direction for future work is to try to extend the results of the (1+1)-ES algorithm to other types of functions or algorithms. Recall that in the OneMax case there is no deception and no interaction between the bits. The results obtained here are strongly related to this lack of deception and interaction. Analogous issues in the case of real valued representations are discussed in [Sal96]. One then can wonder whether such results may hold for more challenging problems or even for population based algorithms on the OneMax problem?
First, it can be noted that the results can be easily applied to any site-wise optimizable binary function (as flipping one bit from 0 to 1 always improves the fitness). These functions have tunable amounts of epistatic interactions.
Note also that for *long k-path* problems with fixed k value (deceptive problem with no local optima in *Hamming* space), Rudolph [Rud96] proves that the (1+1)-ES with stochastic $\frac{1}{l}$ mutation is a lot quicker (see next section) than the deterministic *1-bit-flip* mutation. Hence, for the *long k-path*, the mutations efficiency results are the opposite of those of the OneMax case.
Note finally that this section addressed a particular case of the class of unimodal[40] fitness functions. Still within unimodal functions, next section considers the *long k-path* problem. Compared to the OneMax problem, the *long k-path* probably situate at the other end of the class of unimodal functions. Together with the results in the OneMax case, the study of *long k-path* problems allow us to put some theoretical bounds on the efficiency of both deterministic and stochastic mutations within the class of unimodal fitness functions.

[40]No local optima but the global one.

I.3 Convergence Time for Longpath Problems

Overview Last section compared two binary mutations in a particular case of unimodal functions, the OneMax problem. Still within unimodal functions, this section addresses the same issue for the *long k-path* problem. Since the behavior of the deterministic 1-bit-flip mutation is already known, we only consider the stochastic bit flipping mutation: The asymptotical behavior of a (1+1)-ES process with a stochastic $\frac{c}{l}$ mutation on Rudolph's long k-paths is extensively investigated. First, in the case of $k = l^\alpha$, we prove that the long k-path is a longpath for the (1+1)-ES process, in the sense that the entire path has to be followed before convergence occurs. For $\alpha < 1/2$, expected convergence time is still exponential but some shortcuts will occur meanwhile which speed up the process.
Second, in the case of constant k, the statistical distribution of the convergence time is calculated, and the influence of population size is investigated for different $(\mu + \lambda) - ES$. Besides, the histogram of the solution's first hitting time shows an anomalous peak close to zero, which corresponds to an exceptional set of events that speed up the expected convergence time with a factor of l^2. A direct consequence of this exceptional set is that performing independent (1+1)-ES processes proves to be more advantageous than any population based $(\mu + \lambda) - ES$.

I.3.1 INTRODUCTION

Long k-path problems are unimodal functions where the stochastic $1/l$ mutation largely outperforms the deterministic 1-bit-flip mutation. The relative efficiency of these mutations is at the opposite of the results obtained in the preceding section, for yet another unimodal function: for instance the OneMax function.

Since the behavior of the deterministic 1-bit flipping is, by construction, well understood on long k-path problems, we propose to focus on analyzing the exploration properties of the stochastic bit flipping mutation, in order to better expliciting the role of *shortcuts* (simultaneous k bit flips) in speeding up the convergence.

Long paths are unimodal problems with only one path (in Hamming space) to the optimum. The length of this path is *exponential*[41]. A 1-bit-flip mutation Hill Climber *can* solve it. But it takes an exponential time in string length to reach the solution.
Long path problems [HGD94] have been introduced in GA literature in order to enlighten the notion of problem difficulty for optimization algorithms. Unimodality does not necessarily imply problem easiness. Hence some unimodal problems have been added to the list of well known difficulties: Isolation (*needle in a haystack*), deception, and multimodality -studied in [Gol93] [Gol89] [FM93]- are no longer the only causes for search difficulty.
Horn's longpath [HGD94] for the single-bit-flip Hill Climber is an example of such a difficult unimodal fitness function. It is based on a sequence of strings with the property that two successive strings are at Hamming distance 1 from each other. The k-th string in the sequence has a fitness value of k. All strings s which are not in the sequence get a fitness value of $(l-d)/l$, where d is the Hamming distance between the string s and the first string of the sequence.

Hence, a single-bit-flip Hill Climber is the first lead toward the first string of the path, and then up the whole path. Because of its length, Horn's longpath is an intractable search problem for the single-bit-flip Hill Climber: It visits every string of this path before it

[41] Unless the growth rate is explicitly stated, any asymptotic behavior beyond a polynomial growth in string length will be called *exponential*

ends at the global optimum. Hence, convergence time is equal to the path length, which is exponential in problem size l.

Rudolph [Rud94b] presents a generalization of Horn's longpath to a class of fitness functions called *long k-paths*.
The long k-path $P(l,k)$ is defined by a recursion on the length l of the strings, with $P(1,1) = \{0,1\}$ as the base path. Given the long k-path $P(l,k)$, $P(l+k,k)$ is constructed as follows: A sub-path S_0 of $P(l+k,k)$ is created by taking the path $P(l,k)$ and by prepending k zeros to each point on $P(l,k)$. A sub-path S_1 of $P(l+k,k)$ is created by taking $P(l,k)$ *in reverse order* and prepending k ones to each point on the reversed path. A third path B consists of $k-1$ points created by prepending the following substrings of length k to the last point in the path $P(l,k)$: $0\ldots01$, $0\ldots011$, $\ldots$, $001\ldots11$ and $01\ldots11$. The resulting long k-path $P(l+k,k)$ on strings of length $l+k$ is then defined by the concatenation of S_0, the "bridge path" B and S_1. Horn's longpath is the long k-path for $k=2$.
As an example, consider the construction of the path $P(7,3)$. Starting from $P(1,3) = \{0,1\}$, we obtain the sequence $P(4,3)$:

$$\underbrace{0000_1 \quad 0001_2}_{S_0} \quad \underbrace{0011_3 \quad 0111_4}_{Bridge} \quad \underbrace{1111_5 \quad 1110_6}_{S_1} \tag{53}$$

from which the path $P(7,3)$ is constructed:

$$\begin{array}{ccc} 0000000_1 & 0011110_7 & 1111110_9 \\ 0000001_2 & 0111110_8 & 1111111_{10} \\ 0000011_3 & & 1110111_{11} \\ 0000111_4 & & 1110011_{12} \\ 0001111_5 & & 1110001_{13} \\ 0001110_6 & & 1110000_{14} \\ & & \\ S_0 & \text{Bridge} & S_1 \end{array} \tag{54}$$

- The Hamming distance between two consecutive points of the path is 1.
- A simultaneous flipping of $i < k$ bits can only lead to a point which is i positions away from the original point. To jump over more than k consecutive positions in the path, all the bits of at least one of the blocks $2j+1, \ldots, 2(j+1)$, for $j \in \{0, \ldots, (l-1)/k\}$, have to be flipped. Such a jump will be called a *shortcut*.
- The length of the long k-path is given by $|P(l,k)| = (k+1)2^{(l-1)/k} - k + 1$. Note that for a fixed string length, the path becomes shorter with increasing k.

The behavior of the GA with only one-point crossover and no mutation was experimented in [HR96a]: a GA with complementary crossover (a one-point crossover between a string and its complement) exploits a kind of Royal Road structure ([MFH93] [MH93]) present in the long k-path.
This results in the occurrence of many shortcuts and a growth rate of l^2 for the time to reach the optimum of the long k-path. The authors expect that the complementary crossover approximates the one-point crossover if the latter is used in combination with a big enough population size. However, no conclusion can be drawn about the asymptotic behavior of the GA.

Rudolph [Rud94b] proves $O(l^{k+1}/k)$ as an asymptotic upper bound for the expected time spent by a (1+1)-ES with stochastic $1/l$ mutation rate on a long k-path. For a fixed value of k, this bound is a polynomial of the string length. It shows that for large values of l the (1+1)-ES using the stochastic $1/l$ mutation is a lot quicker than using the deterministic 1-bit-flip mutation in optimizing the path. The difference between the two algorithms is that the stochastic mutation can do more than one bit-flip at the same time. Note that in the case of variable k, as in $k = \sqrt{l}$, the upper bound changes to an uninteresting exponential. So the problem of convergence time with ($k = l^\alpha$, $\alpha > 0$) remained unsolved.

In this section, we restrict to the case of the stochastic bit flipping mutation, and calculate the exact expected time for convergence is calculated for both fixed and l-dependent k. In the case of fixed k, the statistical distribution of convergence time and its variance are given. This allows us to study the influence of population size (as a function of l) on the expected convergence time, with $(\mu + \lambda) - ES$ strategies.

The first section illustrates the construction of the long k-path, and defines some notations. Sections I.3.3 is devoted to the study of expected convergence time for $k = l^\alpha$. Section I.3.4 considers the case of constant k: expected convergence time and its fluctuations are calculated. In the light of these results, different $(\mu+\lambda)-ES$ are compared. The influence of the mutation rate choice is also discussed.

I.3.2 CONSTRUCTION OF THE (k,l)-path.

Construction of the (1,l)-path This is done recursively with respect to l. Let $P_{1,l}$ be the (1,l)-path. A sub-path S_0 is created by prepending 0 to each string in path $P_{1,l}$ and sub-path S_1 by prepending 1 to each string in the reverse of path $P_{1,l}$. The (1,l+1)-path is obtained by concatenating sub-paths S_0 and S_1 . The (k,l)-path can now be built for any k as follows.

Construction of the (k,l)-path 1) Consider the path $P_{1,(l-1)/k}$, which contains $2^{(l-1)/k}$ strings with length $(l-1)/k$.

$$\text{For } \frac{l-1}{k} = 3 : \qquad \begin{array}{ccc} & & \begin{array}{c} 0 \\ 1 \end{array} \\ & 0 < & \\ 0 < & & \\ & 1 < & \\ & & \begin{array}{c} 1 \\ 0 \end{array} \\ & & \begin{array}{c} 0 \\ 1 \end{array} \\ & 1 < & \\ 1 < & & \\ & 0 < & \\ & & \begin{array}{c} 1 \\ 0 \end{array} \end{array} \quad \Longleftrightarrow \quad \begin{array}{c} 000 \\ 001 \\ 011 \\ 010 \\ 110 \\ 111 \\ 101 \\ 100 \end{array}$$

2) Substituting for each 0 (resp. each 1) a substring with length k consisting of 0 (resp. 1), results in a sequence of $2^{(l-1)/k}$ strings with length $l-1$.

$$\text{For } \frac{l-1}{k} = 3: \qquad \begin{array}{l} 0\ldots0 < \begin{array}{l} 0\ldots0 < \begin{array}{l} 0\ldots0 \\ 1\ldots1 \end{array} \\ 1\ldots1 < \begin{array}{l} 1\ldots1 \\ 0\ldots0 \end{array} \end{array} \\ 1\ldots1 < \begin{array}{l} 1\ldots1 < \begin{array}{l} 0\ldots0 \\ 1\ldots1 \end{array} \\ 0\ldots0 < \begin{array}{l} 1\ldots1 \\ 0\ldots0 \end{array} \end{array} \end{array}$$

3) Adding an ultimate branching to each leaf that consists alternatively of $<\begin{array}{c}0\\1\end{array}$ and $<\begin{array}{c}1\\0\end{array}$, builds the so-called skeleton of the (k,l)-path, which consists of $2^{(l+k-1)/k}$ strings with length l. This is actually a sub-sequence extracted from the (k,l)-path as built in [Rud94b].

$$\text{For } \frac{l-1}{k} = 3: \qquad \begin{array}{l} 0\ldots0 < \begin{array}{l} 0\ldots0 < \begin{array}{l} 0\ldots0 < \begin{array}{l} 0 \\ 1 \end{array} \\ 1\ldots1 < \begin{array}{l} 1 \\ 0 \end{array} \end{array} \\ 1\ldots1 < \begin{array}{l} 1\ldots1 < \begin{array}{l} 0 \\ 1 \end{array} \\ 0\ldots0 < \begin{array}{l} 1 \\ 0 \end{array} \end{array} \end{array} \\ 1\ldots1 < \begin{array}{l} 1\ldots1 < \begin{array}{l} 0\ldots0 < \begin{array}{l} 0 \\ 1 \end{array} \\ 1\ldots1 < \begin{array}{l} 1 \\ 0 \end{array} \end{array} \\ 0\ldots0 < \begin{array}{l} 1\ldots1 < \begin{array}{l} 0 \\ 1 \end{array} \\ 0\ldots0 < \begin{array}{l} 1 \\ 0 \end{array} \end{array} \end{array} \end{array}$$

4) The whole path is then constructed from the skeleton, by simply adding bridges between any two strings of the skeleton. A bridge consists of $k-1$ strings which contain either the sequence of substrings ∗0....01∗, ∗0...011∗, ∗001..11∗, and ∗01...11∗ or the inverse one ∗01...11∗, ∗001..11∗, ∗0...011∗, and ∗0....01∗. The whole (k,l)-path is so made of $(k+1)2^{(l-1)/k} - k + 1$ strings.

Remark I.3.1 *In the following, we only consider the case of elitist* $(\mu+\lambda)$*-ES, and the case of the* $(\mu+\lambda)$*-ES is not addressed. Whether the lack of elitism actually improve the convergence speed or not, is left unanswered here.*
Nevertheless, experiments with low k *values, suggest that for low* λ/μ *values (ex. (1,5)-ES,* $k=4$*,* $l=45$*) using the stochastic* $1/l$ *mutation does not follow the path, but takes a* shortcut *from the beginning of the path towards the global optimum: Once it reaches the beginning of the path, the (1,5)-ES process fails in finding the next points of the path, and falls down more or less immediately. This situation can be repeated a number of times, which maximizes the chances of hitting the global optimum (at successive visits of the first point of the path).*
On the other hand, when λ/μ *increases (ex. (1,30)-ES), the selection pressure increases and the process follows the path.*

Note finally that the behavior of (μ, λ)-ES with the deterministic mutation can be predicted: as it cannot take shortcuts, convergence time is still proportional to path's length. Hence unless λ takes exponentially big values, convergence time is exponential in string size l.

The process $X(t)$ In the following, $\mathbf{e}_i := 0.....0$ denotes the first string of the path, $\mathbf{e}_f := 1..10....0$ the last one; N_l denotes the length of the path $N_l = (k+1)2^{(l-1)/k} - k + 1$, N_l^s the length of the skeleton $N_l^s = 2^{(l+k-1)/k}$, and T stands for the first hitting time of string $\mathbf{e}_f$ by the process $X(t)$.

The search follows a process $X(t)$ starting from $X(0) = \mathbf{e}_i$, and evolves as a (1+1)-ES: At each generation $t \in \mathbb{N}$, some mutated string $X'(t)$ is built from $X(t)$ by independently inverting each bit of $X(t)$ with probability l^{-1}. If $X'(t)$ lies farther in the path than $X(t)$, then $X(t+1) = X'(t)$ otherwise $X(t+1) = X(t)$. The path's last string $\mathbf{e}_f$ is therefore the unique absorbing state of the Markov chain X.

In order to characterize the distribution of the mutations used in the process $X(t)$, we first introduce some new notations:
- $\mathcal{MU}$ denotes the set of all possible mutations ($\{$ invert, no-invert$\}^l$)
- Ω_j denotes the set of mutations that invert exactly j bits (we have $\mathcal{MU} = \cup_{j=0}^{l} \Omega_j$).
- J is the random variable representing the number of bits flipped per bitstring with the stochastic $1/l$ mutation:

$$\mathbb{P}(J = j) = p_j = \sum_{i=0}^{l} C_i^l l^{-i} (1 - l^{-1})^{l-i}$$

To select a mutation from $\mathcal{MU}$, first choose j with probability p_j, then choose μ uniformly in Ω_j. The sequence of mutations of the process $X(t)$ are independent and identically distributed with distribution:

$$p_\mu = \sum_{j=0}^{l} p_j \frac{\mathbb{1}_{\mu \in \Omega_j}}{|\Omega_j|}$$

I.3.3 ASYMPTOTIC STUDY FOR LARGE l AND $k = l^\alpha$

This subsection considers the case of $k = l^\alpha$ and large values of l. Recall that the case of $k = O(l)$ is not interesting because it yields a path length of $O(l)$ and an expected number of trials of no more than $O(l^2)$.

I.3.3.1 Main results

The main result of the section is given in proposition I.3.2. It states that expected convergence time is exponential for $k = l^\alpha$.
In subsection 3.1, some simple estimates show that shortcuts speed up the convergence only in the case $\alpha < 1/2$. If $\alpha \geq 1/2$, the path is entirely visited before any shortcut is likely to happen.

Proposition I.3.2
Assume that $k = [l^\alpha]$ for $\alpha \in (0, 1)$.[42]

[42] $[x]$ is the integer part of a real number x.

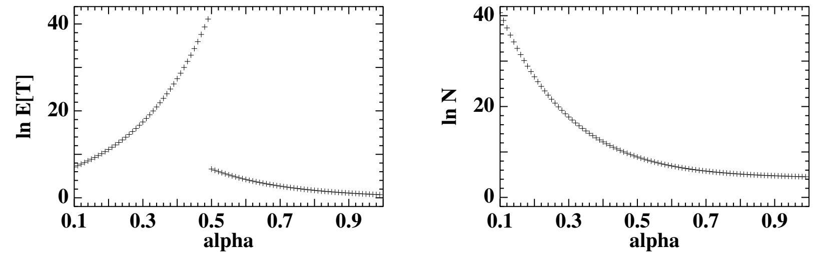


FIG 10 : *Expected convergence time (left) and path length $N_l = (k+1)2^{(l-1)/k} - k + 1$ (right) both plotted for $l = 91$ and $k = l^\alpha$, as a function of α.*

1. If $\alpha < 1/2$, then

$$\lim_{l\to\infty} \frac{1}{l^\alpha \ln(l)} \ln \mathbb{E}[T] = 1. \tag{55}$$

2. If $\alpha \geq 1/2$, then

$$\lim_{l\to\infty} \frac{1}{l^{1-\alpha} \ln(2)} \ln \mathbb{E}[T] = 1. \tag{56}$$

Before turning to the proof we give some results about the waiting times of some shortcuts.

I.3.3.2 Waiting times of some particular events

Let $\mathcal{S}$ be the family that consists of $\frac{l-1}{k}$ shortcuts denoted by $\mathcal{S}_j$ for $j = 1, ..., \frac{l-1}{k}$. The shortcut $\mathcal{S}_j$ is the operator which inverts the k bits whose labels lie within the segment $[(j-1)k+1, jk+1]$.
Similarly let $\mathcal{S}'$ be the family which consists of all the shortcuts which invert at least the k successive bits whose labels lie within one of the intervals $[(j-1)k+1, jk+1]$, $j = 1, ..., (l-1)/k$. Note that some of the shortcuts of the family $\mathcal{S}'$ do not give rise to a jump for the process $X(t)$. Yet, it is clear that $\mathcal{S} \subset \mathcal{S}'$, and $\mathcal{S}'$ actually contains many more shortcuts than $\mathcal{S}$. Nevertheless the waiting time for one particular event of $\mathcal{S}$ or for any event of $\mathcal{S}'$ are almost the same when $k \sim l^\alpha$, of the order of l^{l^α}.

Lemma I.3.3
1. If $l \gg 1$, then for any fixed j, the waiting time $T_{\mathcal{S}_j}$ for the shortcut $\mathcal{S}_j$ obeys an exponential distribution with mean el^k.
2. The waiting time T_{seq} for the sequence $\mathcal{S}_1$,..., $\mathcal{S}_{\frac{l-1}{k}}$ (i.e. waiting for $\mathcal{S}_1$, and then waiting for $\mathcal{S}_2$, and so on until $\mathcal{S}_{\frac{l-1}{k}}$) obeys a $\chi^2(2\frac{l-1}{k})$ distribution with mean $el^k(l-1)/k$ and density:

$$p_{T_{seq}}(t) = \frac{1}{\Gamma(\frac{l-1}{k}) el^k} \left(\frac{t}{el^k}\right)^{\frac{l-1}{k}-1} \exp -\frac{t}{el^k}.$$

3. The waiting time $T_{\mathcal{S}'}$ for some shortcut of the family $\mathcal{S}'$ obeys an exponential distribution with mean $l^k k/(l-1)$.

Although $T_{\mathcal{S}_j}$, T_{seq} and $T_{\mathcal{S}'}$ are integer-valued, we describe their respective distributions by using the notion of "densities", which is usually reserved for real-valued random variables. As an example the statement "$T_{\mathcal{S}_j}$ has an exponential density with mean el^k" should be understood as follows:

$$\mathbb{P}(T_{\mathcal{S}_j} = t) = \frac{1}{el^k} \exp - \frac{t}{el^k} \text{ for any integer } t$$

Proof. At each time $t \in \mathbb{N}$, the probability that the shortcut $\mathcal{S}_j$ occurs is:

$$P_{\mathcal{S}_j} = l^{-k}(1 - l^{-1})^{l-k}.$$

The event "shortcut $\mathcal{S}_i$ occurs" is independent from the event "shortcut $\mathcal{S}_j$ occurs", for all values of j an i, this follows directly from the fact that mutation considers each bit of the string independently from the other bits. Hence, the statistical distribution of the first time $T_{\mathcal{S}_j}$ when $\mathcal{S}_j$ occurs is given by:

$$\mathbb{P}(T_{\mathcal{S}_j} > t) = (1 - P_{\mathcal{S}_j})^t$$

For $1 \ll k \ll l$, we have $P_{\mathcal{S}_j} = l^{-k}e^{-1}$ and consequently:

$$\mathbb{P}(T_{\mathcal{S}_j} \geq l^k t) = \exp(-e^{-1}t)$$

which proves the first statement of the Lemma. Since the exponential variables have no memory (see Appendix A), we have:

$$\mathbb{P}(T_{seq} > t) = \mathbb{P}(T_{\mathcal{S}_1} + ... + T_{\mathcal{S}_{\frac{l-1}{k}}} > t)$$

Furthermore the sum of $\frac{l-1}{k}$ independent exponential variables with mean m obeys the so-called $\chi^2(2\frac{l-1}{k})$ distribution with mean $\frac{l-1}{k}m$ [Fel71, Vol 2, ch. I-3], which proves the second statement.
At each time $t \in \mathbb{N}$, the probability that some shortcut which inverts at least the k successive bits whose labels lie within one of the intervals $[(j-1)k+1, jk+1]$ occurs is:

$$P_{\mathcal{S}'} = \sum\nolimits_{\mu \in \mathcal{S}'} p_\mu$$

As above, this proves that the statistical distribution of the first time $T_{\mathcal{S}'}$ when one of these shortcuts occurs, is given by:

$$\mathbb{P}(T_{\mathcal{S}'} \geq t) = (1 - P_{\mathcal{S}'})^t$$

If we note $\mathcal{S}'_j$ the mutations which invert at least the k successive bits whose labels lie within one of the intervals $[(j-1)k+1, jk+1]$, then we clearly have:

$$P_{S'} \leq \sum_{j=1}^{(l-1)/k} P_{S'_j} = \frac{l-1}{k} l^{-k} \tag{57}$$

Further, let $P^l_{S'}$ be the probability for a string length of l, and $\hat{S}'_j$ the mutations that invert at least the bits at positions $[(j-1)k+1, jk+1]$, and that invert no successive k bits at positions $[(n-1)k+1, nk+1]$ for $n \neq j$. We have the recurrence:

$$P^l_{\hat{S}'_1} = l^{-k}(1 - P^{(l-k)}_{S'})$$

Since $P_{S'}^{(l-k)} \to 0$ for $l >> 1$, we have that

$$P^l_{\hat{S}'_1} \simeq l^{-k}$$

Back to the the family $\mathcal{S}'$, we have

$$\begin{aligned} P^l_{S'} &\geq P^l_{\cup \hat{S}'_j} \\ &\geq \frac{l-1}{k} P^l_{\hat{S}'_1} \\ &\geq \frac{l-1}{k} l^{-k} \end{aligned}$$

The latter inequality, with equation 57 yield that

$$l^k \frac{k}{l-1} P_{\mathcal{S}'} \to 1 \text{ for } l >> 1,$$

we conclude that

$$\mathbb{P}(T_{\mathcal{S}'} > l^k \frac{k}{l-1} t) \simeq exp(-t)$$

□

Lemma I.3.4
The waiting times between two successive jumps of the process $X(t)$ are independent from each other and have means bounded by l below and by el above.

Proof. The independence is insured by the mutation process, which is completely blind. Furthermore, whatever $X(t)$ is (except the final point $\mathbf{e}_f$ of course), there is a string just after it in the longpath which differs only from $X(t)$ in one bit, so that the probability p that a jump occurs is always bounded below by:

$$p \geq l^{-1}(1-l^{-1})^{l-1} \simeq e^{-1} l^{-1}$$

Furthermore, for $j = 1, ..., k-1$, there is at most one string in the longpath which is after $X(t)$ and which differs from $X(t)$ in exactly j bits. As a consequence, the next jump of $X(t)$ must be either a jump to one of these strings, or a shortcut of the family $\mathcal{S}'$. As a consequence, the probability p that a jump occurs is always bounded above, by:

$$p \leq \sum_{j=1}^{k-1} l^{-j}(1-l^{-1})^{l-j} + P_{\mathcal{S}'} \simeq e^{-1} l^{-1} + l^{-k+1}/k \underset{\text{for } k \geq 2}{\leq} (e^{-1} + 2^{-1}) l^{-1} \leq l^{-1}$$

The bounds on p then imply the bounds on the mean waiting time T, since $\mathbb{P}(T > t) = (1-p)^t$ and $\mathbb{E}[T] = \sum_{t=1}^{\infty} \mathbb{P}(T \geq t) = 1/p$. □
The above proof actually shows that, if $k \geq 3$, the mean waiting time between two jumps of $X(t)$ is $el + O(1)$.

Proof of Proposition I.3.2 In the following, to improve readability of the text, we often consider the asymptotical exponential laws of the waiting times studied above, rather than the exact geometrical laws. Note however that the proof (and conclusions) follow in the same manner using the geometrical law.

We now give an equivalent construction of the process $X(t)$. Recall that for the process $X(t)$ (defined in section I.3.2), the mutations are chosen in $\mathcal{MU}$ according to the distribution:

$$p_\mu = \sum_{j=0}^{l} p_j \frac{\mathbb{I}_{\mu \in \Omega_j}}{|\Omega_j|}$$

We introduce:

- **A sequence of independent mutations** $(\mu_t)_{t \geq 1}$ chosen in $\mathcal{MU} \setminus \mathcal{S}'$, each with probability
$$\frac{p_\mu}{\sum_{m \notin \mathcal{S}'} p_m} \mathbb{I}_{\mu \notin \mathcal{S}'}$$
- **A process** $X_{ws}(t)$ starting from $X_{ws}(0) = e_i$, the first point of the path, defined as follows: mutation μ_{t+1} is applied to string $X_{ws}(t)$, and the mutated string is accepted if it lies farther than $X_{ws}(t)$ in the (k,l)-path, otherwise $X_{ws}(t+1) = X_{ws}(t)$. In the following T_{ws} stands for the first hitting time of the last string $\mathbf{e}_f$ by the process X_{ws}.
- **A sequence of random variables** $(T_j)_{j \geq 0}$ such that
$T_0 = 0$ and $T_j = T_{j-1} + \sigma_j$, where (σ_j) are independent geometric variables with parameter $p = l^{-k}(l-1)/k$:
$$\forall n \in \mathbb{N} \setminus \{0\}, \; \mathbb{P}(\sigma_j = n) = p(1-p)^{n-1}$$
- **A sequence of independent mutations** $(\mu'_j)_{j \geq 1}$ chosen in $\mathcal{S}'$, each with probability
$$\frac{p_\mu}{\sum_{m \in \mathcal{S}'} p_m} \mathbb{I}_{\mu \in \mathcal{S}'}$$

We then **construct a process** $\bar{X}(t)$ by first setting $j = 1$, then iterating:

1. While $T_{j-1} \leq t < T_j - 1$, apply mutation μ_{t+1} (from the above sequence) to string $\bar{X}(t)$ and accept the mutated if it lies farther than $\bar{X}(t)$ in the path, otherwise $\bar{X}(t+1) = \bar{X}(t)$.
2. If $t = T_j - 1$, apply mutation μ'_{j+1} to string $\bar{X}(t)$ and accept the mutated if it lies farther than $\bar{X}(t)$ in the path, otherwise $\bar{X}(t+1) = \bar{X}(t)$.
3. Indent $j \to j+1$ and go to step 1.

Let us prove that the law of the process $\bar{X}(t)$ is the same as that of $X(t)$. Since the selection rule is the same for both processes (following a (1+1)-ES scheme), we only need to prove that the mutations of the process $\bar{X}(t)$ follow the same distribution as the mutations of $X(t)$.
First, we prove the following Lemma

Lemma I.3.5
If $(T_i)_{i \geq 0}$ *is a sequence of random variable such that* $T_0 = 0$ *and* $T_i = T_{i-1} + \tau_i$ *where* (σ_i) *are*

independent geometric variables with parameter p ($\forall n \in \mathbb{N} \setminus \{0\}$, $\mathbb{P}(\sigma_j = n) = p(1-p)^{n-1}$), and

$$G(n) = \sum_{i \geq 0} \mathbb{P}(T_i = n),$$

then, for all $n \in \mathbb{N} \setminus \{0\}$

$$G(n) = p$$

Proof. It is proven in [GS86] that if the (σ_i) are independent and identically distributed random variables, such that the renewal process defined by (T_i) is aperiodic –which is the case here since the law of the (σ_i) is geometric– then

$$\Phi(z) = 1 + F(z)\Phi(z) \tag{58}$$

where F is the generating function of σ_i, and Φ is the generating function of the sequence $\{G(n), n = 0, 1, \dots\}$:

$$\Phi(z) = \sum_{n \geq 0} z^n G(n)$$

It is then easy to verify that

$$G(n) = p \text{ for } n \geq 1, \text{ and } G(0) = 1$$

fulfills Equation (58), which characterizes the function $G(n)$. □

Let $(\bar{\mu}_n)$ be the sequence of mutations of $\bar{X}$. We now prove that the distribution of $(\bar{\mu}_n)$ is the same as that of (μ_n).

$$\begin{aligned}
\mathbb{P}(\bar{\mu}_n = \mu) &= \mathbb{P}(n \in \{T_j\}_{j\geq 1}, \bar{\mu}_n = \mu) + \mathbb{P}(n \notin \{T_j\}_{j\geq 1}, \bar{\mu}_n = \mu) \\
&= \mathbb{P}(n \in \{T_j\}_{j\geq 1})\mathbb{P}(\mu_1' = \mu) + \mathbb{P}(n \notin \{T_j\}_{j\geq 1})\mathbb{P}(\mu_1 = \mu) \\
&= \frac{1}{\mathbb{E}[T_1]} \frac{p_\mu}{\sum_{m\in\mathcal{S}'} p_m} \mathbb{I}_{\mu\in\mathcal{S}'} + (1 - \frac{1}{\mathbb{E}[T_1]}) \frac{p_\mu}{\sum_{m\notin\mathcal{S}'} p_m} \mathbb{I}_{\mu\notin\mathcal{S}'}
\end{aligned}$$

Using the fact that $T_{\mathcal{S}'}$ satisfies $\sum\limits_{m\in\mathcal{S}'} p_m = \mathbb{P}(T_{\mathcal{S}'} = 1)$ (Lemma I.3.3) and that $T_{\mathcal{S}'}$ is geometric, we have

$$\sum_{m\in\mathcal{S}'} p_m = \frac{1}{\mathbb{E}[T_{\mathcal{S}'}]}$$

We obtain the desired result:

$$\mathbb{P}(\bar{\mu}_n = \mu) = p_\mu$$

The mutation distributions of $X(t)$ and $\bar{X}(t)$ are identical, hence the processes $X(t)$ and $\bar{X}(t)$ follow the same law. In the following we study the convergence time T of the process $\bar{X}(t)$.

We can split all the possible ways to go from $\mathbf{e}_i$ to $\mathbf{e}_f$ into two complementary groups:

1. G1: $T_{ws} < T_{\mathcal{S}'}$
2. G2: $T_{ws} \geq T_{\mathcal{S}'}$

Study of the process X_{ws} The process X_{ws} visits at most every string of the whole path (that is to say at most $N_l := (k+1)2^{(l-1)/k} - k + 1$ strings), and the waiting times between two jumps have means bounded above by el. Thus:

$$\mathbb{E}[T_{ws}] \leq elN_l \simeq elk2^{(l-1)/k} \tag{59}$$

Furthermore, since no shortcut of $\mathcal{S}'$ happens for X_{ws}, the process must visit at least one over two strings of the skeleton, or a string of the neighboring bridge, that is to say at least $N_s^l := 2^{(l-1)/k}$ strings. Furthermore the waiting times between two jumps are statistically independent and have means bounded below by l. Hence

$$\mathbb{E}\left[T_{ws}\right] \geq l2^{(l-1)/k} \tag{60}$$

If we note τ_i the waiting times for the successive jumps of the process $X(t)$, and since the variables τ_i are exponential with mean bounded from above by el

$$\mathbb{E}[T_{ws}^2] = \mathbb{E}[(\sum_1^{N_l} \tau_i)^2] = N_l\mathbb{E}[\tau_1^2] + N_l(N_l-1)\mathbb{E}[\tau_1]^2$$

Hence,

$$\mathbb{E}[T_{ws}^2] \quad \leq \quad 2(el)^2 N_l^2 \tag{61}$$

By the usual large deviations principle for the sum of independent random variables [Str84, theorem 3-8], we get that, for any $\delta > 0$, there exists some $c_\delta > 0$ such that:

Proposition I.3.6

$$\mathbb{P}\left(T_{ws} < (1-\delta)lN_l^s\right) \quad \leq \quad \exp(-c_\delta lN_l^s), \tag{62}$$

$$\mathbb{P}\left(T_{ws} \geq (1+\delta)elN_l\right) \quad \leq \quad \exp(-c_\delta elN_l). \tag{63}$$

If we consider the convergence time T of the process $\bar{X}(t)$, the expectation of T writes:

$$\mathbb{E}[T] = \mathbb{E}_{ws}\mathbb{E}_{S'}[T/T_{ws} < T_{S'}]\mathbb{P}(G_1) + \mathbb{E}_{ws}\mathbb{E}_{S'}[T/T_{S'} \leq T_{ws}]\mathbb{P}(G_2) \tag{64}$$

where $\mathbb{E}_{ws}$ is the expectation with respect the distribution of the sequence $(\mu_n)_{n\geq 1}$, and $\mathbb{E}_{S'}$ is the expectation with respect the distribution of the sequence $(\mu'_j, T_j)_{j\geq 1}$. In the following, the notation $\mathbb{E}$ stands for $\mathbb{E}_{ws}\mathbb{E}_{S'}$.

We shall now estimate the four terms of the right-hand member of this equality:

Estimate of $\mathbb{P}(G_1)$ $\mathbb{P}(G_1) = \mathbb{P}(T_{ws} < T_{\mathcal{S}'})$. The third statement of Lemma I.3.3 establishes that $\mathbb{P}(T_{\mathcal{S}'} > t) = \exp -\frac{t}{l^k k/(l-1)}$ for any t. Since T_{ws} and $T_{\mathcal{S}'}$ are independent, we then have:

$$\mathbb{P}(G_1) = \mathbb{P}(T_{\mathcal{S}'} > T_{ws}) = \mathbb{E}\left[\exp -\frac{T_{ws}}{l^k k/(l-1)}\right] \tag{65}$$

• **First case:** $\alpha < 1/2$ We decompose the expectation with respect to the event: $T_{ws} \geq \frac{1}{2}lN_l^s$:

$$\begin{aligned}\mathbb{E}\left[\exp -\frac{T_{ws}}{l^k k/(l-1)}\right] &= \mathbb{E}\left[\exp -\frac{T_{ws}}{l^k k/(l-1)}/T_{ws} < \frac{1}{2}lN_l^s\right]\mathbb{P}(T_{ws} < \frac{1}{2}lN_l^s) \\ &\quad + \mathbb{E}\left[\exp -\frac{T_{ws}}{l^k k/(l-1)}/T_{ws} \geq \frac{1}{2}lN_l^s\right]\mathbb{P}(T_{ws} \geq \frac{1}{2}lN_l^s) \\ &\leq \mathbb{P}(T_{ws} < \frac{1}{2}lN_l^s) + \exp -\frac{lN_l^s}{2l^k k/(l-1)}\end{aligned} \tag{66}$$

If $k = [l^\alpha]$, $\alpha < 1/2$, then (62) and the fact that $lN_l^s \sim l2^{l^{1-\alpha}} \gg 2l^{l^\alpha} l^{\alpha-1}$ implies:

$$\mathbb{P}(G_1) \simeq 0, \text{ when } k = [l^\alpha],\ \alpha < 1/2 \text{ and } l \gg 1 \tag{67}$$

- **Second case** $\alpha \geq 1/2$ Applying Jensen's inequality to Eq. (65):

$$\mathbb{P}(G_1) \geq \exp -\frac{\mathbb{E}[T_{ws}]}{l^k k/(l-1)}$$

From Eqs. (59) and (60), If $k = [l^\alpha]$, $\alpha \geq 1/2$, then the fact that $elN_l \sim el2^{l^{1-\alpha}} \ll l^{l^\alpha} l^{\alpha-1}$ implies:

$$\mathbb{P}(G_1) \simeq 1, \text{ when } k = [l^\alpha],\ \alpha \geq 1/2 \text{ and } l \gg 1 \tag{68}$$

Expectation of T under $G1$ Under G1, $T_{S'} > T_{ws}$, then $T = T_{ws}$ and

$$\mathbb{E}[T/T_{S'} > T_{ws}] = \mathbb{E}[T_{ws}/T_{S'} > T_{ws}] \leq \mathbb{E}[T_{ws}] \leq elN_l \tag{69}$$

- Case of $\alpha \geq \frac{1}{2}$

$$\begin{aligned}
\mathbb{E}[T/T_{S'} > T_{ws}] &= \mathbb{E}[T_{ws}/T_{S'} > T_{ws}] \\
&= \frac{\mathbb{E}[T_{ws}\mathbb{1}_{T_{S'} > T_{ws}}]}{\mathbb{P}(T_{S'} > T_{ws})} \\
&= \frac{\mathbb{E}_{ws}[T_{ws}\mathbb{P}_{S'}(T_{S'} > T_{ws})]}{\mathbb{P}(T_{S'} > T_{ws})} \\
&\geq \mathbb{E}[T_{ws}\exp(-\frac{T_{ws}}{\mathbb{E}[T_{S'}]})] \\
&\geq \mathbb{E}[T_{ws}] - \mathbb{E}[T_{ws}(1-\exp(-\tfrac{T_{ws}}{\mathbb{E}[T_{S'}]}))]
\end{aligned} \tag{70}$$

Further, using the fact that $1 - e^{-\rho} \leq \rho \wedge 1$, and the bounds on $\mathbb{E}[T_{ws}]$ and $\mathbb{E}[T_{ws}^2]$ of equations (59) and (61), we have

$$\begin{aligned}
\mathbb{E}[T_{ws}(1-\exp(-\tfrac{T_{ws}}{\mathbb{E}[T_{S'}]}))] &\leq \rho\mathbb{E}[T_{ws}\mathbb{1}_{T_{ws} < \rho\mathbb{E}[T_{S'}]}] + \mathbb{E}[T_{ws}\mathbb{1}_{T_{ws} \geq \rho\mathbb{E}[T_{S'}]}] \\
&\leq \rho\mathbb{E}[T_{ws}] + \mathbb{E}[T_{ws}^2]^{1/2}\mathbb{P}(T_{ws} > \rho\mathbb{E}[T_{S'}])^{1/2} \\
&\leq \mathbb{E}[T_{ws}](\rho + 2\mathbb{P}(T_{ws} > \rho\mathbb{E}[T_{S'}])^{1/2})
\end{aligned} \tag{71}$$

By (63) and the fact that $lN_l << E[T_{ws}] \sim l^{\alpha-1}l^{l^\alpha}$ for $\alpha \geq 1/2$, we have $\mathbb{P}(T_{ws} > \rho\mathbb{E}[T_{S'}]) \to 0$ for $l >> 1$. Hence, if we choose $\rho = 1/l$ in Eq. (71), then we get that $\mathbb{E}[T_{ws}(1-\exp(-\frac{T_{ws}}{\mathbb{E}[T_{S'}]}))]$ is at a lower order than $\mathbb{E}[T_{ws}]$.

We then get from equation (70) that

$$\mathbb{E}[T/T_{S'} > T_{ws}] \geq \mathbb{E}[T_{ws}](1 - \frac{1}{l}) \tag{72}$$

Combining (69) and (72), if $k = [l^\alpha]$ and $\alpha \geq \frac{1}{2}$ we obtain:

$$\lim_{l\to\infty} \frac{1}{l^{1-\alpha}\ln(2)} \ln \mathbb{E}\left[T/G1\right] = 1. \tag{73}$$

Expectation of T under G2 In the worst case, the following sequence of events (happening after $T_{S'}$) insures that the process reaches $\mathbf{e}_f$, whatever the intermediate event: shortcut $\mathcal{S}_1$, then $\mathcal{S}_2$, etc until $\mathcal{S}_{(l-1)/k}$. Hence

$$\mathbb{E}[T/T_{S'} < T_{ws}] \leq \mathbb{E}[T_{S'}/T_{S'} < T_{ws}] + \sum_{j=1}^{(l-1)/k} \mathbb{E}[T_{S_j}] \tag{74}$$

Note that the expectation in the second term of the right hand term of the above inequality is no longer conditional. Indeed, after the first shortcut happens (first term), the subsequent events can be described as independent exponential variables, and in particular independent from $T_{S'}$.

We now estimate the expression $\mathbb{E}[\tau/\tau < t]$, where τ is exponentially distributed with mean t_0.

$$\begin{aligned} \mathbb{E}[\tau/\tau < t] &= \frac{\int_0^t \frac{s}{t_0} e^{-s/t_0} ds}{\int_0^t \frac{1}{t_0} e^{-s/t_0} ds} \\ &= t_0 \frac{1 - e^{-\frac{t}{t_0}}(\frac{t}{t_0} + 1)}{1 - e^{-\frac{t}{t_0}}} \\ &= t_0 f(\frac{t}{t_0}) \end{aligned} \tag{75}$$

where

$$f(t) = \frac{1 - e^{-t}(t+1)}{1 - e^{-t}} \tag{76}$$

Combining the fact that f is uniformly bounded by 1, and Lemma (I.3.3), in equation (74) we get the estimate

$$\mathbb{E}\mathbb{E}[T/T_{S'} < T_{ws}] \leq \mathbb{E}[T_{S'}] + e\frac{l-1}{k} l^k \leq e(\frac{l-1}{k} + 1) l^k \tag{77}$$

In the best case we need only one shortcut of the family $\mathcal{S}'$ to reach $\mathbf{e}_f$.

$$\mathbb{E}\mathbb{E}[T/T_{S'} < T_{ws}] \geq \mathbb{E}\mathbb{E}[T_{S'}/T_{S'} < T_{ws}] \tag{78}$$

First, applying the result of Eq. (75) to the right term of the latter inequality, and then using the fact that $f(\frac{t}{t_0}) \geq \frac{t}{4t_0} \wedge \frac{1}{2}$ (with $t_0 = \mathbb{E}[T_{S'}]$) we get

$$\begin{aligned} \mathbb{E}\mathbb{E}[T_{S'}/T_{S'} < T_{ws}] &\geq \mathbb{E}[T_{S'}]\, \mathbb{E}[f(\frac{T_{ws}}{\mathbb{E}[T_{S'}]})] \\ &\geq \mathbb{E}[T_{S'}]\mathbb{E}[\frac{T_{ws}}{4\mathbb{E}[T_{S'}]} \wedge \frac{1}{2}] \\ &\geq \mathbb{E}[\frac{T_{ws}}{4} \wedge \frac{\mathbb{E}[T_{S'}]}{2}] \end{aligned}$$

- We now consider the case of $\alpha < \dfrac{1}{2}$

$$
\begin{aligned}
\mathbb{E}[\frac{T_{ws}}{4} \wedge \frac{\mathbb{E}[T_{S'}]}{2}] &\geq \mathbb{E}[\frac{T_{ws}}{4} \mathbb{I}_{T_{ws} < 2\mathbb{E}[T_{S'}]}] + \frac{\mathbb{E}[T_{S'}]}{2} \mathbb{P}(T_{ws} \geq 2\mathbb{E}[T_{S'}]) \\
&\geq \frac{\mathbb{E}[T_{S'}]}{2}(1 - \mathbb{P}(T_{ws} < 2\mathbb{E}[T_{S'}]))
\end{aligned}
$$

Since $\alpha < \frac{1}{2}$ we have $\mathbb{E}[T_{S'}] << lN_l^s$, then by (62):

$$
\begin{aligned}
\mathbb{P}(T_{ws} < 2\mathbb{E}[T_{S'}]) &\leq \mathbb{P}(T_{ws} < \frac{l}{2} N_l^s) \\
&\leq e^{-c_{1/2} l N_l^s}
\end{aligned}
$$

Hence, in the asymptotical framework, we have

$$
\mathbb{E}\mathbb{E}[T/T_{S'} < T_{ws}] \geq \mathbb{E}[T_{S'}](1 + \frac{1}{l}) \tag{79}
$$

Combining (77) and (79), if $k = [l^\alpha]$ then for $\alpha < \frac{1}{2}$:

$$
\lim_{l \to \infty} \frac{1}{l^\alpha \ln(l)} \ln \mathbb{E}\left[T/G2\right] = 1. \tag{80}
$$

Substituting (67), (69) and (80) into equation (64) (shown below), proves the first point of the Proposition ($\alpha < 1/2$). For instance, $\ln(a_l + b_l) \simeq \ln(a_l)$ if $b_l/a_l \to 0$ for $l >> 1$.

$$
\mathbb{E}[T] = \underbrace{\mathbb{E}_{ws}\mathbb{E}_{S'}[T/T_{ws} < T_{S'}]}_{\leq elN_l} \quad \underbrace{\mathbb{P}(G_1)}_{\leq 2\exp - \frac{lN_l^s}{2l^{l^\alpha} l^{\alpha-1}}} \quad + \underbrace{\mathbb{E}_{ws}\mathbb{E}_{S'}[T/T_{S'} < T_{ws}]}_{\sim l l^{l^\alpha}} \underbrace{\mathbb{P}(G_2)}_{\sim 1}
$$

Substituting (68), (73) and (77) into (64) proves the second point of the Proposition, for $\alpha \geq 1/2$. For instance, $\ln(a_l + b_l) \simeq \ln(2b_l)$ if $a_l/b_l \to 1$ for $l >> 1$.

$$
\mathbb{E}[T] = \underbrace{\mathbb{E}_{ws}\mathbb{E}_{S'}[T/T_{ws} < T_{S'}]}_{\sim lN_l} \underbrace{\mathbb{P}(G_1)}_{\sim 1} + \underbrace{\mathbb{E}_{ws}\mathbb{E}_{S'}[T/T_{S'} < T_{ws}]}_{\leq ell^{l^\alpha}} \quad \underbrace{\mathbb{P}(G_2)}_{\leq 1 - \exp(\frac{-elN_l}{l^{\alpha-1} l^{l^a lpha}})}
$$

Remarks

1. Note that for $\alpha \geq 1/2$, the waiting time for one shortcut occurrence (equation 77 under G2) is much longer than –the upper bound of– the expected convergence time when following the entire path. However, this is not enough to describe the effective behavior of the process:

2. A direct consequence of equations (67) and (68) is that shortcuts speed up the convergence in the case of $\alpha < 1/2$ only. Otherwise, for $\alpha \geq 1/2$, the process will reach $\mathbf{e}_f$ by following the whole path before any shortcut occurs.

3. Note also that the critical value $\alpha_c = 1/2$ holds true in the limit $l \to \infty$, but is slightly different for large but finite l. Indeed comparing the different expressions, we get that the transition between the two regimes occurs when $l^{\alpha_c} \ln(l) = l^{1-\alpha_c} \ln(2)$, i.e. for:

$$
\alpha_c = \frac{1}{2} - \frac{\ln \ln l - \ln \ln 2}{2 \ln l}
$$

which converges to the value $1/2$ at rate $\ln \ln l / \ln l$.

I.3.4 ASYMPTOTIC BEHAVIOR FOR LARGE l AND FIXED k

This section is devoted to the case of a fixed value of k in the asymptotic framework $l \gg 1$. The main result is stated in Subsection I.3.4.1 and claims that the expected first hitting time is $el^{k+1}/k\ln(l/k)$. This result is not an estimate but the dominant term of the expansion of $\mathbb{E}[T]$ with respect to l. We then give, with the same order of precision, the complete probability density of T. It shows that, although the normalized variance of T goes to 0 as $l \to \infty$, the decay rate is so slow that the variance actually remains of order 1 for a large band of values of l. We are also able to exhibit a class of exceptional realizations of probability kl^{-1} where the process hits the final string much quicker than the expected value $\mathbb{E}[T]$. After studying this set of realizations, we describe how we can take advantage of it in Subsection I.3.4.4. The influence of mutation probability is investigated in section I.3.4.5. Finally subsection I.3.4.6 compares different population based evolution strategies behavior.

I.3.4.1 Main results

Let us consider the first hitting time T of the process $X(t)$ (defined in section I.3.2).

Proposition I.3.7
The expectation of the first hitting time T for large l is

$$\mathbb{E}[T] \underset{l\to\infty}{\simeq} \frac{el^{k+1}}{k\ln(l/k)}. \tag{81}$$

which means that $\mathbb{E}[T] \times \left(\frac{el^{k+1}}{k\ln(l/k)}\right)^{-1}$ converges to 1 as l goes to infinity.

Proof. See Subsection I.3.4.2. □

The following Proposition gives more detail about the statistical distribution of the first hitting time T.

Proposition I.3.8
1. The normalized variance of T goes to 0 as $l \to \infty$:

$$V(T) := \frac{\sqrt{\mathbb{E}[T^2] - \mathbb{E}[T]^2}}{\mathbb{E}[T]} \underset{l\to\infty}{\longrightarrow} 0$$

2. As $l \to \infty$, the statistical distribution of T can be represented at the first order as $el^k\bar{T}$, where $\bar{T}$ is a random variable with density :

$$p_{\bar{T}}(t) = \sum_{m=1}^{\infty} \sum_{\substack{n_1,\ldots,n_m=1 \\ n_1+\ldots+n_{m-1}<L \\ n_1+\ldots+n_m\geq L}}^{\infty} \int_0^{\infty}\ldots\int_0^{\infty} d\tau_1\ldots d\tau_m \prod_{j=1}^{m} e^{-n_j\tau_j}(1-e^{-\tau_j})\delta_{\tau_1+\ldots+\tau_m=t},$$

and $L = (l-1)/k$.

Proof. See Subsection I.3.4.2. □

In the above expression of density $p_{\bar{T}}(t)$, the integer m stands for the number of shortcuts which are necessary to reach the final string, τ_j, $j = 1,\ldots,m$ are the times between two shortcuts, and n_j, $j = 1,\ldots,m$ are the ranges of the corresponding shortcuts, as explained in Subsection I.3.4.2.

By integrating $tp_{\bar{T}}(t)$ and $t^2 p_{\bar{T}}(t)$ with respect to t, we can compute the normalized variance of T. It then appears that the relative fluctuations with respect to the mean stay of order 1 for a large band of values of l, although it decays to 0, as shown by Figure 11, where the normalized variance is plotted.

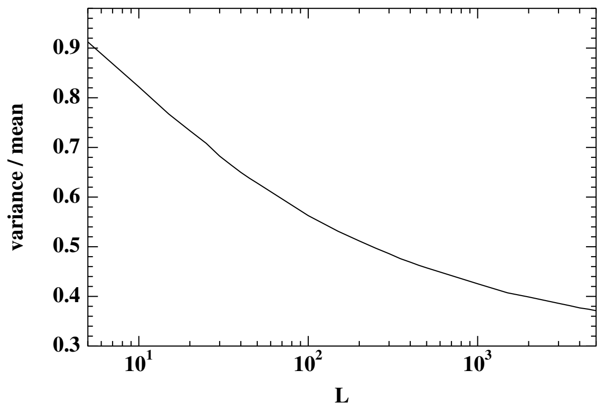


FIG 11 : *Normalized variance of the variable T, which, at the first order only, depends on the ratio $L := (l-1)/k$.*

The probability densities of T for typical and relevant values of the parameters k and l, are plotted in Figure 12. The theoretical histograms of T present an anomalous peak close to zero. This corresponds to outliers which are not at all artifacts of the theory. This leftmost peak in the histogram of T, corresponds to the event Ω_1: shortcut $\mathcal{S}_1$ is the first to occur (i.e. before $\mathcal{S}_2$, ..., $\mathcal{S}_{(l-1)/k}$). The peak disappears when the first term ($m = 1$) in the expression of $p(\bar{t})$ is omitted. The event Ω_1 results in a smaller convergence time than average. This exceptional set of realizations is studied carefully in Subsection I.3.4.4.

I.3.4.2 Proofs of Propositions I.3.7 and I.3.8

The (k,l)-path can be decomposed in the following manner:

$$
\begin{array}{rcllrcl}
\mathcal{M}_1 & = & \{0..0 * * & \dots & *\} & \bigcup & \{bridges\} \\
\mathcal{M}_2 & = & \{1..11..1 * * & \dots & *\} & \bigcup & \{bridges\} \\
\mathcal{M}_3 & = & \{1..10..01..1 * * & \dots & *\} & \bigcup & \{bridges\} \\
\mathcal{M}_4 & = & \{1..10..00..01..1 * * & \dots & *\} & \bigcup & \{bridges\} \\
 & \vdots & & & & \vdots & \\
\mathcal{M}_{(l-k-1)/k} & = & \{1..10..0 & \dots & 0..01..1*\} & \bigcup & \{bridges\} \\
\mathcal{M}_{(l-1)/k} & = & \{1..10..0 & \dots & 0..00..0*\} & &
\end{array}
$$

Note that $\mathcal{M}_1$ contains about one half of all the strings of the path, and more generally $\mathcal{M}_j$ contains about a proportion 2^{-j} of all the strings of the path. The longpath also presents some remarkable symmetries:
Applying $\mathcal{S}_1$ to string n^o j, with $j < N_l/2$, gives the string n^o $N_l - j + 1$.
Applying$\mathcal{S}_2$ to string n^o j, with $j < N_l/4$, gives the string n^o $N_l/2 - j + 1$.
Furthermore since the waiting times $T_{\mathcal{S}_j}$ are independent and identically distributed, we have for any fixed j_0:

$$\mathbb{P}(T_{\mathcal{S}_{j_0}} < T_{\mathcal{S}_1}, .., T_{\mathcal{S}_{j_0-1}}, T_{\mathcal{S}_{j_0+1}}, ..., T_{\mathcal{S}_{(l-1)/k}}) = \frac{k}{l-1}.$$

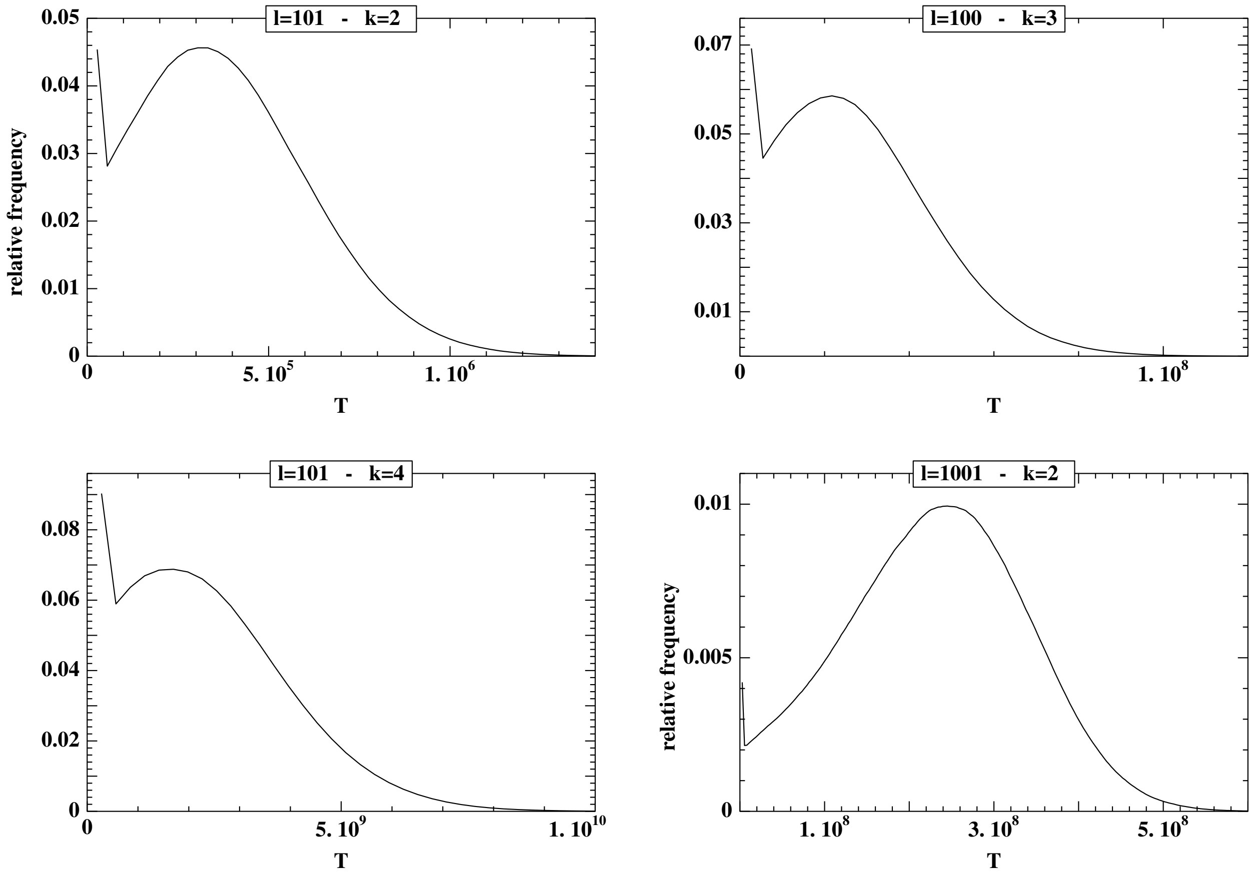


FIG 12 : *Theoretical histograms of the random variable T for different values of the parameters k and l.*

Let us discuss *when* the process enters $\mathcal{M}_1^c$ (the complementary of $\mathcal{M}_1$).
• If no shortcut of $\mathcal{S}'$ occurs, then it will take an exponentially long time (path length).
• If neither $\mathcal{S}_1$ nor $\mathcal{S}_2$ occurs, then the process must still visit an exponentially large number of strings, which takes an exponentially long time.

• If $\mathcal{S}_2$ occurs, but not $\mathcal{S}_1$, then we may distinguish two situations:
1- If one of the $\mathcal{S}_j$, $j \geq 3$, occurs before $\mathcal{S}_2$, then the process must still visit an exponentially large number of strings, which takes an exponentially long time.
2- The only way that could produce a not-exponentially large time is to have $\mathcal{S}_2$ be the first shortcut of the family $\mathcal{S}'$ to occur. Then the process jumps at the end part of $\mathcal{M}_1$ and will soon enter $\mathcal{M}_2$.
This event, that we denote by $\mathcal{E}$, therefore provides a probability $k/(l-1)$ to enter $\mathcal{M}_2$ (this is the probability that $\mathcal{S}_2$ be smaller than the other $\mathcal{S}_j$). We shall see that the contribution of $\mathcal{E}$ is negligible because its probability is too small to be taken into account.
The only probable way to enter $\mathcal{M}_1^c$ is indeed to have $\mathcal{S}_1$ occur. The access time to $\mathcal{M}_1^c$, hence, obeys an exponential distribution with mean el^k (lemma I.3.3). It will be denoted in the following by $el^k\tau_1$:

$$\mathbb{P}(\tau_1 > t) = \exp(-t)$$

Let us now study the point *where* the process enters $\mathcal{M}_1^c$.
From the above discussion, this point depends on the shortcuts $\mathcal{S}_j$, $j \geq 2$, which may have occurred before $\mathcal{S}_1$.
If $\mathcal{S}_2$ occurs before $\mathcal{S}_1$, then the process enters $\mathcal{M}_2$: thus the process has probability 1/2 to enter $\mathcal{M}_2$ in this way. That is why the contribution of the event $\mathcal{E}$ can be neglected.
If $\mathcal{S}_2$ does not occur before $\mathcal{S}_1$, but $\mathcal{S}_3$ does, then the process enters $\mathcal{M}_3$.
More generally, if $\mathcal{S}_i$, $i \leq j-1$, do not occur before $\mathcal{S}_1$, but $\mathcal{S}_j$ does, then the process enters $\mathcal{M}_j$. Let us denote this particular j by N_1. Since the waiting times for the shortcuts $\mathcal{S}_i$ obey independent exponential distributions with mean el^k, we have:

$$\mathbb{P}\left(N_1 = j/\tau_1 = t\right) = \mathbb{P}(\tau_2', ..., \tau_{j-1}' > t, \tau_j' < t/\tau_1 = t\big) = \left(1 - e^{-t}\right) e^{-(j-1)t}$$

We can then iterate the above arguments. We then find that we need to do it M times, where M is the first time that $\sum_{i=1}^m N_j$ reaches $(l-1)/k$, which means that the process enters $\mathcal{M}_{(l-1)/k}$.
As a consequence, by denoting

$$L = \frac{l-1}{k}, \tag{82}$$

we can describe the first hitting time T as follows.
For large l, the first hitting time T obeys the distribution of:

$$T = el^k \bar{T}, \tag{83}$$

where $\bar{T}$ is defined in the following way. Consider a sequence of independent and identically distributed real-valued random variables τ_j, $j = 1, 2, ...$ with exponential density: $p(\tau) = \mathbb{I}_{\tau \geq 0} e^{-\tau}$. Consider also a sequence of independent integer-valued random variables N_j, $j = 1, 2, ...$, whose distributions depend on τ_j. More exactly, $\mathbb{P}(N_j = n/\tau_j = \tau) = p_\tau(n)$, where

$$p_\tau(n) = e^{-(n-1)\tau}(1 - e^{-\tau}), \qquad n = 1, 2, ... \tag{84}$$

If we denote by M_L the first exit time:

$$M_L = \inf\left\{ m \geq 1, \sum_{j=1}^m N_j \geq L \right\}, \tag{85}$$

then the random variable $\bar{T}$ is defined by:

$$\bar{T} = \sum_{j=1}^{M_L} \tau_j. \tag{86}$$

The closed-form expression of the probability density of $\bar{T}$ given in Proposition I.3.8 can then be deduced directly from the definition of $\bar{T}$.
Roughly speaking, from (83), we expect that $\mathbb{E}[T] = el^k \mathbb{E}[\bar{T}]$, from (86) that $\mathbb{E}[\bar{T}] \simeq \mathbb{E}[M_L]\mathbb{E}[\tau_1] = \mathbb{E}[M_L]$ and from (85) that $\mathbb{E}[M_L] \simeq L/\mathbb{E}[N_1]$. However the distributions of M_L and τ_j are joint, since M_L depends on $(N_j)_j$ whose distribution depends on $(\tau_j)_j$.

Therefore the assertion "$\mathbb{E}[\bar{T}] \simeq \mathbb{E}[M_L]\mathbb{E}[\tau_1]$" is not so obvious, and actually it is false in some sense. Furthermore, the statistical distribution of N_j is:

$$\mathbb{P}(N_j = n) = \int_0^\infty p(\tau)p_\tau(n)d\tau = \frac{1}{n(n+1)}. \tag{87}$$

A noticeable feature which will appear crucial in the following is that $\mathbb{E}[N_j] = +\infty$. It will prevent us from applying the standard theorems of the probability theory (strong law of large numbers and central limit theorem) and it will give rise to anomalous behaviors for M_L and the other relevant quantities. The asymptotic behaviors of the variables M_L and $\bar{T}$ are given by the following Propositions.

Proposition I.3.9
The sequence of normalized random variables $\frac{\ln L}{L}M$ converges in mean, in L^2 and in probability as $L \to \infty$ to 1, which means that, for any $\delta > 0$:

$$\frac{\ln L}{L}\mathbb{E}[M_L] \overset{L\to\infty}{\longrightarrow} 1 \ , \ \left(\frac{\ln L}{L}\right)^2 \mathbb{E}[M_L^2] \overset{L\to\infty}{\longrightarrow} 1 \text{ and } \mathbb{P}\left(\left|\frac{\ln L}{L}M_L - 1\right| \geq \delta\right) \overset{L\to\infty}{\longrightarrow} 0. \tag{88}$$

Proposition I.3.10
For any $q = 1, 2$, we have

$$\frac{\mathbb{E}[\bar{T}^q]}{\mathbb{E}[M_L^q]} \underset{L\to\infty}{\longrightarrow} 1 \tag{89}$$

Combining these Propositions yields the statement of Proposition I.3.7 ($q = 1$) and the first point of Proposition I.3.8 ($q = 2$). In the following, we shall first study the statistical distribution of the variable M_L and prove Proposition I.3.9. The study of the distribution of $\bar{T}$ then proves Proposition I.3.10.

I.3.4.3 Proofs of propositions I.3.9 and I.3.10

We aim at studying the statistical distribution of the stopping time M_L as defined by Eq. 85 in Subsection I.3.4.2. We shall use the well known Tauberian theorem (Theorem 4 [Fel71, Vol. 2, ch. XIII-5]):

Lemma I.3.11
If $u(x) \geq 0$ is monotone and

$$\int_0^\infty e^{-\lambda x}u(x)dx \underset{\lambda\to 0^+}{\sim} \lambda^{-\rho}L(\lambda^{-1})$$

where L is slowly varying (which means that $L(sx)/L(s) \to 1$ as $s \to \infty$ for any $x > 0$), then:

$$u(x) \underset{x\to+\infty}{\sim} \frac{1}{\Gamma(\rho)}x^{\rho-1}L(x).$$

The main result is the Proposition I.3.9.

Proof of Proposition I.3.9. Let us denote by S_m the partial sums:

$$S_m = \sum_{j=1}^m N_j.$$

Then we can express the statistical distribution of the variable M_L in terms of S_m:

$$\mathbb{P}(M_L = m) = \mathbb{P}(S_{m-1} < L) - \mathbb{P}(S_m < L).$$

Furthermore we introduce F the repartition function of the random variable N_1:

$$F(n) = \mathbb{P}(N_1 \le n) = 1 - \frac{1}{n+1},$$

whose Laplace transform is:

$$\begin{aligned} \hat{F}(\lambda) &:= \mathbb{E}[e^{-\lambda N_1}] \\ &= 1 + (e^{\lambda} - 1)\ln(1 - e^{-\lambda}). \end{aligned}$$

Since $\mathbb{P}(S_m \le L) = F^{*m}(L)$, where the star stands for the convolution, we can express the mean of the variable M_L as follows:

$$\begin{aligned} \mathbb{E}[M_L] &= \sum_{m=1}^{\infty} m\left(\mathbb{P}(S_{m-1} < L) - \mathbb{P}(S_m < L)\right) = \sum_{m=0}^{\infty} \mathbb{P}(S_m < L) \\ &= \sum_{m=0}^{\infty} F^{*m}(L-1) \\ &= I(L-1) + F * \sum_{m=0}^{\infty} F^{*m}(L-1), \end{aligned}$$

where $F^{*0}(L) = I(L) = \mathbb{1}_{L \ge 0}$. Denoting by $\hat{H}_1(\lambda)$ the Laplace transform of the series $\mathbb{E}[M_L]$:

$$\hat{H}_1(\lambda) := \sum_{L=0}^{\infty} \mathbb{E}[M_L] e^{-\lambda L},$$

we then get:

$$\hat{H}_1(\lambda) = \frac{e^{-\lambda}}{\lambda} + e^{-\lambda}\hat{F}(\lambda)\hat{H}_1(\lambda)$$

from which we deduce that:

$$\hat{H}_1(\lambda) = \frac{e^{-\lambda}}{\lambda(1 - e^{-\lambda}\hat{F}(\lambda))} \underset{\lambda \to 0^+}{\simeq} \frac{1}{\lambda^2 |\ln \lambda|}.$$

Applying Lemma I.3.11 we then get:

$$\mathbb{E}[M_L] \underset{L \to +\infty}{\simeq} \frac{L}{\ln L}. \tag{90}$$

Let us now deal with the fluctuations.

$$\begin{aligned} \mathbb{E}[M_L^2] &= \sum_{m=1}^{\infty} m^2 \left(\mathbb{P}(S_{m-1} < L) - \mathbb{P}(S_m < L)\right) \\ &= \sum_{m=0}^{\infty} (2m+1) F^{*m}(L-1). \end{aligned}$$

Denoting by $H_2(L) := \sum_{m=0}^{\infty} m F^{*m}(L)$ and by $\hat{H}_2(\lambda)$ its Laplace transform, we have:

$$H_2(L-1) = F * H_2(L-1) + \mathbb{E}[M_L]$$

which implies:

$$\hat{H}_2(\lambda) = e^{-\lambda}\hat{F}(\lambda)\hat{H}_2(\lambda) + \frac{e^{-\lambda}}{\lambda(1-e^{-\lambda}\hat{F}(\lambda))}$$

and consequently:

$$\hat{H}_2(\lambda) = \frac{e^{-\lambda}}{\lambda(1-e^{-\lambda}\hat{F}(\lambda))^2} \underset{\lambda\to 0^+}{\simeq} \frac{1}{\lambda^3(\ln\lambda)^2}.$$

Still applying Lemma I.3.11 we get:

$$\mathbb{E}[M_L^2] \underset{L\to+\infty}{\simeq} \frac{L^2}{(\ln L)^2}. \tag{91}$$

Combining Eq. 90 with Eq. 91, we obtain the convergence in probability of the variable $\frac{\ln L}{L}M_L$ to 1:

$$\begin{aligned}\mathbb{P}\left(\left|M_L\frac{\ln L}{L}-1\right|\leq\delta\right) &\leq \frac{\mathbb{E}\left[\left(M_L\frac{\ln L}{L}-1\right)^2\right]}{\delta^2}\\ &= \frac{\mathbb{E}\left[M_L^2\right]\left(\frac{\ln L}{L}\right)^2-2\mathbb{E}[M_L]\frac{\ln L}{L}+1}{\delta^2} \underset{L\to\infty}{\longrightarrow} 0\end{aligned}$$

□

This result is asymptotic, that is to say, it holds true for large L. However, precisely analyzing the convergence of M_L, it appears that the convergence rate is very slow (logarithmic). If we deal with values of L which are of order 100 or 1000, it is then necessary to be very careful when using Proposition I.3.9. The mean $\mathbb{E}[M_L]$ actually behaves as $L/\ln L$ even for -relatively- small values of L, as shown by Figure 13. However the relative fluctuations with respect to the mean stay of order 1 for a large band of values of L, and vanish only for $L \gg 10^9$ (Figure 13). In Figures 14 we have plotted the computed histograms of the variable M_L for different values of L.

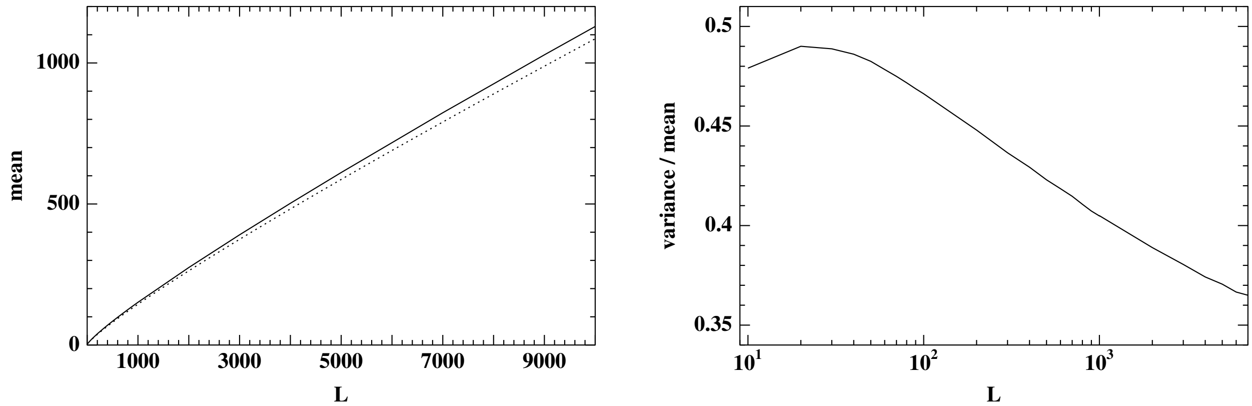


FIG 13 : *Left picture: Mean of the random variable M_L as a function of L (solid line). In dashed line is plotted the curve $L \mapsto L/\ln L$. Right picture: Normalized variance $\mathbb{E}\left[M_L^2-\mathbb{E}[M_L]^2\right]^{1/2}/\mathbb{E}\left[M_L\right]$ of the random variable M_L as a function of L.*

Lemma I.3.12

Let $p \in \{1,2\}$, $q \in \mathbb{N}^*$ and $\psi_1,\ldots,\psi_q$ be continuous functions from $\mathbb{N}^*$ into $\mathbb{R}^+$ satisfying

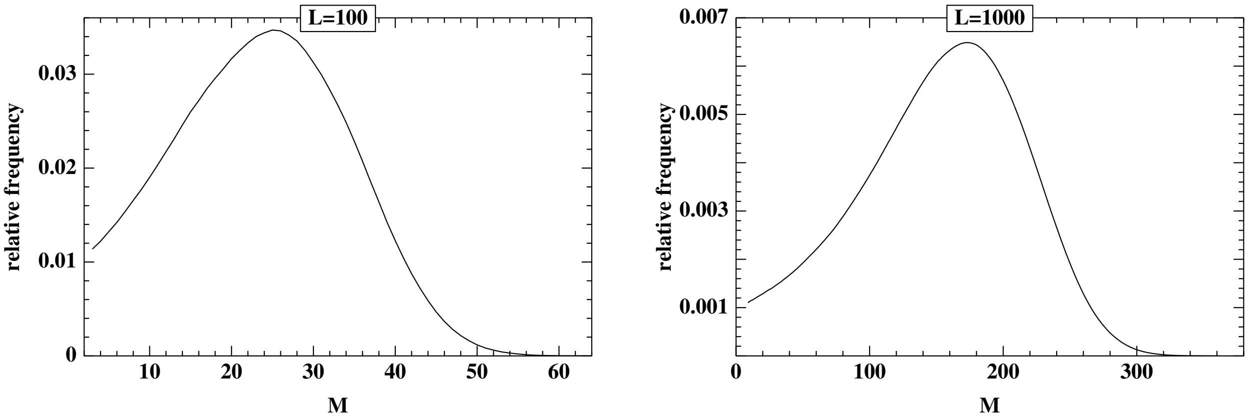


FIG 14 : *Theoretical histograms of the random variables* M_L *for* $L = 100$ *(left picture) and* $L = 1000$ *(right picture).*

$0 < a \leq \psi_j(.) \leq b < \infty$. *Then:*

$$\left| \frac{\mathbb{E}[M_L^p \psi_1(N_1)...\psi_q(N_q)] - \mathbb{E}[M_L^p]\mathbb{E}[\psi_1(N_1)]...\mathbb{E}[\psi_q(N_q)]]}{\mathbb{E}[M_L^p]\mathbb{E}[\psi_1(N_1)]...\mathbb{E}[\psi_q(N_q)]} \right| \underset{L\to\infty}{\longrightarrow} 0.$$

This result is not surprising. Indeed for -relatively- small L, M_L and N_1,...,N_q are correlated. But if L is large, then M_L gets large too, so that the statistical distribution of M_L involves many N_j and becomes independent of the realization of the first particular N_1, ..., N_q. However the convergence rate is very slow (logarithmic) and all the slower as q becomes large.
Proof. For simplicity we give the result for $q = 1$.
The extension to an arbitrary q is straightforward. By the definition of M_L:

$$\mathbb{P}(M_L = m/N_1 = n) = \mathbb{P}\left(\sum_{j=2}^{m-1} N_j < l - n \leq \sum_{j=2}^{m} N_j\right) = \mathbb{P}(M_{L-n} = m - 1/N_1 = n),$$

which establishes that:

$$\mathbb{E}[M_L^p \psi_1(N_1)] = \mathbb{E}[(1 + M_{L-\tilde{N}_1})^p \psi_1(\tilde{N}_1)]$$

where $\tilde{N}_1$ obeys the same distribution of N_1, but is independent from $M_.$. Thus:

$$\mathbb{E}[M_L^p \psi_1(N_1)] = \sum_{n=1}^{\infty} \frac{1}{n(n+1)} \psi_1(n) \mathbb{E}[(1 + M_{L-n})^p]$$

We finally apply Lebesgue's theorem:

$$\lim_{L\to\infty} \frac{\ln L}{L} \mathbb{E}[M_L^p \psi_1(N_1)] = \sum_{n=1}^{\infty} \frac{1}{n(n+1)} \psi_1(n) = \mathbb{E}[\psi_1(N_1)]$$

□

Proof of Proposition I.3.10. By Proposition I.3.8, we can compute $\mathbb{E}[\bar{T}]$ by integrating the density $p_{\bar{T}}(t)$ multiplied by t. Since

$$\int_0^\infty e^{-n\tau}(1-e^{-\tau})d\tau = \frac{1}{n(n+1)} \quad \text{and} \quad \int_0^\infty \tau e^{-n\tau}(1-e^{-\tau})d\tau = \frac{2n+1}{n^2(n+1)^2}$$

we get:

$$\mathbb{E}[\bar{T}] = \sum_{m=1}^{\infty} \sum_{n_1,\dots,n_m=1}^{\infty} \mathbb{I}_{n_1+\dots+n_{m-1}<L\le n_1+\dots+n_m} \sum_{i=1}^{m} \frac{2n_i+1}{n_i^2(n_i+1)^2} \prod_{j\neq i}^{m} \frac{1}{n_j(n_j+1)}.$$

The sum over i contains m terms. The first $m-1$ terms are equal by simple permutation of the index. Only the last one ($i=m$) differs. We can therefore rewrite the long expression as:

$$\mathbb{E}[\bar{T}] = \mathbb{E}\left[(M_L-1)\frac{2N_1+1}{N_1(N_1+1)}\right] + \mathbb{E}\left[\frac{2N_{M_L}+1}{N_{M_L}(N_{M_L}+1)}\right].$$

The second term of the right-hand side is uniformly bounded, while the first term converges to $\mathbb{E}[M_L-1]\mathbb{E}\left[\frac{2N_1+1}{N_1(N_1+1)}\right]$ by Lemma I.3.12. Since the second expectation is equal to 1, we get the result in the case $q=1$. The case $q=2$ can be dealt with the same way, since the expectation $\mathbb{E}[\bar{T}^2]$ can be expressed in terms of the processes N_j and M_L:

$$\begin{aligned}\mathbb{E}[\bar{T}^2] = \;& \mathbb{E}\left[(M_L-1)(M_L-2)\frac{2N_1+1}{N_1(N_1+1)}\frac{2N_2+1}{N_2(N_2+1)}\right] + \mathbb{E}\left[(M_L-1)\frac{3N_1^2+3N_1+1}{N_1^2(N_1+1)^2}\right] \\ & +2\mathbb{E}\left[(M_L-1)\frac{2N_1+1}{N_1(N_1+1)}\frac{2N_{M_L}+1}{N_{M_L}(N_{M_L}+1)}\right] + \mathbb{E}\left[\frac{3N_{M_L}^2+3N_{M_L}+1}{N_{M_L}^2(N_{M_L}+1)^2}\right].\end{aligned}$$

The result can be interpreted as follows. The statistical distribution of the first hitting time $\bar{T}$ depends on the fluctuations of M_L and τ_j. Both are correlated, but for large L, we deal with a large M_L, and consequently with the sum of a large number of τ_j. By the strong law of large numbers, the fluctuations of this sum goes to 0, so that the fluctuations of T become only dependent on those of M_L.

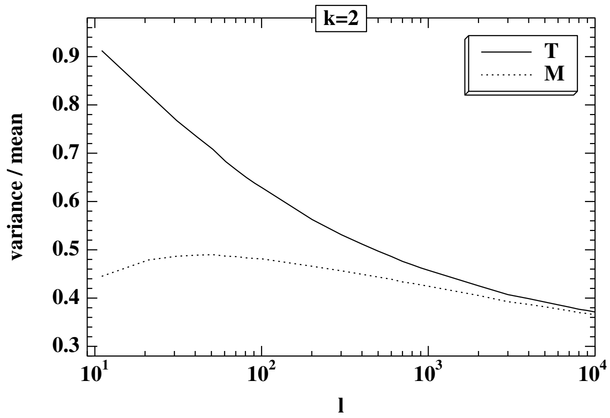


FIG 15 : *Normalized variances of the variables T and M_L ($L=(l-1)/k$).*

I.3.4.4 A class of exceptional events

We now go back to a very interesting point which consists in the anomalous peak close to 0 in the histogram of the first hitting time T. We can describe the corresponding set of realizations (*shortcut $\mathcal{S}_1$ happens first*) which gives rise to a very quick convergence of the process.

Proposition I.3.13
There exists a set of realizations Ω_1 with probability

$$\mathbb{P}(\Omega_1) \underset{l\to\infty}{\simeq} kl^{-1} \tag{92}$$

such that the conditional distribution of the first hitting time given Ω_1 is an exponential distribution with mean $2el^{k-1}k$:

$$\mathbb{P}\left(T \geq a/\Omega_1\right) \underset{l\to\infty}{\simeq} \exp\left(-\frac{a}{2el^{k-1}k}\right). \tag{93}$$

Proposition I.3.13 shows that the expected convergence time under Ω_1 ($2el^{k-1}k$) is roughly l^2 times shorter than the global expected time T ($\frac{el^{k+1}}{k\ln(l/k)}$ in Proposition I.3.7). A noticeable fact is then that the convergence time with $N = O(l)$ processes will be l^2 times shorter than the expected time for a single process with very high probability, because it is then highly probable that at least one of the N processes will realize the favorable event Ω_1: shortcut $\mathcal{S}_1$ occurs first. We quantify this statement in the following corollary.

Corollary I.3.14
Assume that we deal with a set of N processes which evolve independently. Then if N is large, denoting $c = \dfrac{N}{l/k}$, there exists a set of realizations Ω_N with probability

$$\mathbb{P}(\Omega_N) \underset{l\to\infty}{\simeq} 1 - e^{-c} \tag{94}$$

such that the conditional expectation of the first hitting time is:

$$\mathbb{E}\left[T_N/\Omega_N\right] \underset{l\to\infty}{\simeq} 2ef(c)l^{k-1}k \tag{95}$$

where $f(c) = \frac{\mathrm{Ei}(c)-\gamma-\ln c}{e^c-1}$ and γ is the Euler constant $\gamma \simeq 0.57...$ (see Figure 16).
More exactly, the conditional distribution of T given Ω_N is:

$$T \underset{l\to\infty}{\simeq} -2el^{k-1}k\ln\left(\frac{1-Z}{c}\right), \tag{96}$$

where Z is a $[0,1]$-valued random process with density $p(z) = \mathbb{1}_{z\in[0,1]}e^{-z}(1-e^{-1})^{-1}$.

Proof of Proposition I.3.13 We still represent the statistical distribution of the first hitting time T as (83)-(86). In this framework, we define the set Ω_1 as the set of the realizations which fulfill the condition $M_L = 1$:

$$\Omega_1 = \{M_L = 1\} = \{N_1 \geq L\}.$$

Since the distribution of N_1 is given by (87), the probability of this set is simply $\mathbb{P}(\Omega_1) = L^{-1}$. Given Ω_1, the process $X(t)$ obeys the following evolution: before $el^k\tau_1$, it evolves without shortcut and arrives at the string n$^o j$; at time $el^k\tau_1$, by shortcut $\mathcal{S}_1$, the process jumps at string n$^o (k+1)2^{(l-1)k} - k + 1 - j$; finally the process goes to the final string. Furthermore the statistical distribution of τ_1 given Ω_1 is the statistical distribution of τ_1 given $\tau_2, ..., \tau_L > \tau_1$, where the τ_j are independent random variables with exponential distribution and mean 1:

$$\begin{aligned}
\mathbb{P}(\tau_1 > t/\Omega_1) &= \frac{\mathbb{P}(\tau_2, ..., \tau_L > \tau_1 > t)}{\mathbb{P}(\tau_2, ..., \tau_L > \tau_1)} \\
&= \frac{\int_t^\infty d\tau_1 e^{-\tau_1}\left(\int_{\tau_1}^\infty d\tau_2 e^{-\tau_2}\right)^{L-1}}{\int_0^\infty d\tau_1 e^{-\tau_1}\left(\int_{\tau_1}^\infty d\tau_2 e^{-\tau_2}\right)^{L-1}} \\
&= e^{-Lt}
\end{aligned}$$

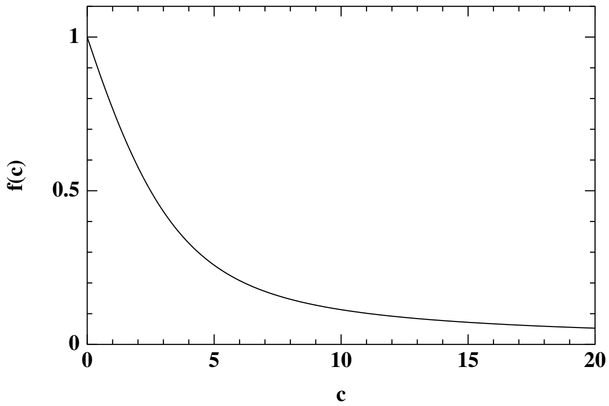


FIG 16 : *Function* $c \mapsto f(c)$.

The result of the Proposition then follows readily.

Proof of Corollary I.3.14 We adopt the same notations as (83)-(86). We add a subscript $i = 1, ..., N$ to each quantity which stands for the labels of the N independent processes. The set Ω_N is then defined as:

$$\Omega_N = \left\{ \inf_{i=1,...,N} M_{L,i} = 1 \right\}.$$

The probability of the complementary set is:

$$\mathbb{P}(\Omega_N^c) = (1 - \mathbb{P}(M_{L,1} = 1))^N = (1 - L^{-1})^N \overset{l \to \infty}{\longrightarrow} e^{-c}.$$

where $c = \dfrac{N}{l/k}$. The global first hitting time is the minimum of the first hitting times corresponding to each process:

$$T = \inf_{i=1,...,N} T_i,$$

where the T_i are independent and identically distributed random variables defined as in (83)-(86). Furthermore, given Ω_N, the statistical distribution of T is:

$$T = \inf_{i \in I} T_i,$$

where $I = \{i = 1, ..., N \text{ s.t. } M_{L,i} = 1\}$ and T_i are independent and identically distributed random variables defined as (93). We have

$$\mathbb{P}(|I| = j/\Omega_N) = \mathbb{P}(|I| = j/|I| \geq 1) = \frac{\mathbb{P}(|I| = j)}{\sum_{j'=1}^{N} \mathbb{P}(|I| = j')}.$$

For any j', the probability that $|I|$ be equal to j' is the probability that j' of the N processes satisfy $M_{L,i} = 1$, and that the other ones satisfy $M_{L,i} > 1$. Since $\mathbb{P}(M_{L,i} = 1) = L^{-1}$, this implies:

$$\mathbb{P}(|I| = j') = C_N^{j'} L^{-j'} (1 - L^{-1})^{N-j'}$$

For $N = cL$ and $L \gg 1$, the binomial distribution becomes equivalent to a Poisson distribution:

$$\mathbb{P}(|I| = j') = e^{-c} \frac{c^{j'}}{j'!}$$

and consequently:

$$\mathbb{P}(|I| = j/\Omega_N) = \frac{e^{-c}}{1-e^{-c}}\frac{c^j}{j!}, \text{ for } j \geq 1.$$

Thus

$$\begin{aligned}
\mathbb{P}(T \geq t) &= \sum_{j=1}^{\infty} \mathbb{P}\left(|I| = j, T_1 \geq t, ..., T_j \geq t\right) \\
&= \sum_{j=1}^{\infty} \frac{e^{-c}}{1-e^{-c}}\frac{c^j}{j!} e^{-jt/(2el^{k-1}k)} \\
&= \frac{\exp\left(ce^{-t/(2el^{k-1}k)}\right) - 1}{e-1}.
\end{aligned}$$

The interested reader can then check that this is exactly the statistical distribution of the random variable defined by (96).

I.3.4.5 Optimization of the mutation probability

In the following we investigate the influence of c (for stochastic $\frac{c}{l}$ mutation) on the time to convergence of the (1+1)-ES process on long k-path problems. This shows that the best mutation rate is k/l.

Proposition I.3.15
Let us assume that the mutation probability is cl^{-1}.
1. For large l, the first hitting time T_d obeys the statistical distribution of $e^{d-1}d^{-k}T_1$, where T_1 is as described in Section I.3.4. In particular its mean is:

$$\mathbb{E}[T_d] \underset{l\to\infty}{\simeq} \frac{e^d}{d^k}\frac{l^{k+1}}{k\ln(l/k)}. \tag{97}$$

2. Knowing k, the best choice for d is $d = k$.

Proof. The same arguments as for the case $d = 1$ yield the result. The only difference comes from the fact that the waiting times $T_{\mathcal{S}_j}$ for the shortcuts $\mathcal{S}_j$ obey different distributions. Indeed, for each instant, the probability that the shortcut $\mathcal{S}_j$ occurs is:

$$P_{\mathcal{S}_j} = \left(\frac{d}{l}\right)^k \left(1 - \frac{d}{l}\right)^{l-k} \simeq d^k e^{-d} l^{-k}.$$

The event "shortcut $\mathcal{S}_i$ occurs" is independent from the event "shortcut $\mathcal{S}_j$ occurs", for all values of j an i, this follows directly from the fact that mutation considers each bit of the string independently from the other bits. Hence, the statistical distribution of $T_{\mathcal{S}_j}$ is given by:

$$\mathbb{P}(T_{\mathcal{S}_j} > t) = (1 - P_{\mathcal{S}_j})^t \simeq \exp -\frac{t}{e^d d^{-k} l^k},$$

which proves that $T_{\mathcal{S}_j}$ obeys an exponential distribution with mean $e^d d^{-k} l^k$. The remainder of the proof is then exactly the same as for the case $d = 1$. □

I.3.4.6 Discussion and comparison of different algorithms

In this subsection we discuss the asymptotic behavior for large l and fixed k of different evolution strategies. The main result is that all population based evolution strategies are less efficient (in number of evaluations) than a (1+1)-ES. Moreover, the best strategy is to simultaneously run several independent (1+1)-ES processes.
This latter result has already been stated in subsection I.3.4.4 : Corollary (I.3.14) shows that with cl/k independent processes ($c > 1$) evolving simultaneously, the first hitting time (of the fastest process) is about l^{k-1} with very high probability $(1 - e^{-c})$. Recall that with one process $X(t)$, the first hitting time is about $l^{k+1}/\ln(l)$ (proposition I.3.7). The first strategy (independent processes) is therefore very advantageous and much more rapid in terms of number of generations, as well as number of evaluations.

Consider now a number of processes that communicate together by exchanging the best so-far string, as in the case of a $(1 + \lambda) - ES$. Expected first hitting time is necessarily longer than that of N processes evolving independently. This is due to the fact that the order of shortcut occurrence determines the convergence time (as explained in I.3.4.2). More precisely, the only way to get a different order of convergence time is that the event Ω_1 happens for some parent: the shortcut $\mathcal{S}_1$ is then the first shortcut to occur. Exchanging the best so-far string forces the N processes to follow the same order of shortcut occurrence. In fact, the string resulting from the first shortcut contaminates –more or less quickly– the other processes, and so on.

In the case of a $(1+\lambda)-ES$, the string resulting from the first shortcut immediately colonizes the population. But expected time for a shortcut and for any improvement gets N times smaller, hence, the expected first hitting time is N times smaller than that of a single process. Hence, the number of evaluations remains the same, and there is no advantage to increasing λ.

Consider now a $(\mu+\lambda)-ES$ with $\lambda = \alpha\mu$, starting at string $\mathbf{e}_i$. Due to elitism, the μ parents are always on the path. If a shortcut occurs, it takes a time of order $\sum_{k=1}^{\mu-1} l/(k\alpha)$ (so less than $l\alpha^{-1}\ln(\mu)$) for the string resulting from the shortcut to dominate the population. It is then very unlikely that another shortcut (exponential distribution) happens within this time, and happens first. An upper bound of this probability is given in the following:
Suppose a shortcut different from $\mathcal{S}_1$ happens for one of the μ parents. The probability that the shortcut $\mathcal{S}_1$ happens for one of the remaining $(\mu - 1)$ parents, within a time $l\alpha^{-1}\ln(\mu)$ (even if it happens after another contaminating shortcut) is:

$$\mathbb{P}\left(\inf_{i=1..\mu-1}(T_i) \leq l\alpha^{-1}\ln(\mu)\right) = 1 - \mathbb{P}\left(T_i > l\alpha^{-1}\ln(\mu)\right)^{\mu-1} \simeq 1 - \exp -\frac{l\mu\alpha^{-1}\ln(\mu))}{l^k} \quad (98)$$

If $k = 2$ and $\mu = O(l)$, this probability is 1 as l goes to infinity, hence the event Ω_1 happens before the first shortcut colonizes the population. however, this is not the probability of the event Ω_1, but an upper bound ($\mathcal{S}_1$ does not necessarily happens first). And, as explained above, μ independent processes are obviously more advantageous, even in this case.
If $k \geq 3$, then the probability tends to zero as l goes to infinity (even if $\mu = O(l)$). Hence all μ individuals follow the first unlucky shortcut, and lose any chance of realizing the event Ω_1. $(\mu+\lambda)-ES$ only speeds the convergence with a factor of λ in terms of number of generations.

I.3.5 SUMMARY AND CONCLUSION

This section investigates the behavior of a (1+1)-ES process using the stochastic c/l bit flipping mutation on Rudolph's long k-paths, in the asymptotic framework $l \gg 1$.

Together with the results of last section on the unimodal OneMax function, this section allows to put some theoretical bounds on the efficiency of the stochastic mutation within the modality spectrum of fitness functions, for instance it ranges from quasi linear to exponential in string length.

More precisely, this section investigates both cases of variable and fixed k values:
First, for $k = l^{\alpha}$, we prove that expected convergence time is exponential. Besides, shortcuts speed up the convergence only if $\alpha < 1/2$. Otherwise (if $\alpha \geq 1/2$), the process reaches the solution by following the whole path before any shortcut occurs.
Second, in the case of a fixed value of k, the expected first hitting time T is equal to $el^{k+1}/k \ln(l/k)$ (at the first order with respect to l). The normalized variance of T goes to 0 as $l \to \infty$ at a very slow rate. Further, the study of the statistical distribution of convergence time shows an anomalous peak close to zero, corresponding to the event *shortcut* $\mathcal{S}_1$ *happens first*. This event happens with probability kl^{-1}, and yields a convergence time, a factor of l^2 smaller than the expected value $\mathbb{E}[T]$. Therefore, the best strategy for taking advantage of this distribution, is to perform $O(l)$ independent (1+1)-ES processes, so that one of these processes is very likely to realize the exceptional event. On the other hand, population based processes limit diversity in shortcut occurrence, drifting all individuals to blindly follow the first occurring shortcut. This implies the same number of evaluations as with a (1+1)-ES.
Finally, expected convergence time distribution is given for any mutation rate d/l, $d > 0$. As expected, the smallest convergence time is obtained with a k/l mutation probability.

The results obtained in this section implicitly illustrate a key phenomenon related to EAs: EA dynamics can be very sensitive to rare and deviant events related to the exploration properties of the stochastic bit flipping mutation. This phenomena is confirmed by the experiments of next chapter which demonstrate that atypical properties of landscapes and operators better reflect EA behavior than average typical quantities does.
Note finally that another important result follows from this study, as it answers the question: Is there a longpath for a (1+1)-ES ? Indeed, in the case $k = l^{\alpha}$ and $\alpha \geq 1/2$, the process follows the path and convergence time is exponential. As for other algorithms, one can still wonder whether such a longpath exists. Yet the notion of "following a path" has to be defined for population based algorithms. Partial answers to these questions are given in the following section, where a fitness function is constructed in the binary framework so that the GA empirically follows a *path* that has been arbitrarily chosen in the search space.

I.4 Candidate Longpaths for Genetic Algorithms

Overview The last section proves that an instance of Rudolph's long k-paths is a longpath for the stochastic bit flipping mutation Hill Climber. In this section, we propose a generic way of constructing longpaths for population based genetic algorithms using crossover and stochastic bit flipping mutation. The construction is based on a sequence of landmarks determining the expected path of the GA. The algorithm is lead from one landmark to another by means of modified OneMax subproblems. The system of landmarks allows one to control the path of the GA. When an exponential number of landmarks is positioned in the search space (exponential in the string length l), we demonstrate that the expected number of generations needed by the GA to reach the optimum is lower-bounded by an exponential of $\sqrt{l}$. We prove that landmarks taken from Rudolph's long k-paths allow to construct a second unimodal longpath for the (1+1)-ES using the stochastic bit flipping mutation, with a path that contains much more individuals than that of the *long k-path*).

I.4.1 INTRODUCTION

A *path* of an iterative search procedure is the recording of a succession of states of this procedure over one particular course of its search for the optimum. A path of a genetic algorithm (GA), for example, is a sequence of populations, each population describing the state of this GA at a given generation of one particular run. By restricting the state information of the GA to the best individual contained in the population, a path becomes a sequence of strings. In the case of an elitist GA, the fitness values of the strings in this sequence are non-decreasing.
As another example of a path, consider the single-bit-flip Hill Climber on the space $E = \{0,1\}^l$ of binary strings of length l. A path of this Hill Climber is a sequence of strings $\{p^n\}_{n=1}^N$ satisfying two properties: the Hamming distance between two successive strings is always 1, and the fitness values of the strings in the sequence are increasing.
When a fitness function is designed in such a way that a given search procedure has no choice but to follow one particular path, the fitness function and the path can be identified. *Horn's longpath* [HGD94] for the single-bit-flip Hill Climber is an example of such a fitness function. Seen as a path, it is a sequence of strings with the property that two successive strings are at Hamming distance 1 from each other. Seen as a fitness function, Horn's longpath yields a fitness value of k for the k-th string in the sequence, and a value of $(l-d)/l$ for all strings s which are not in the sequence, where d is the Hamming distance between the string s and the first string of the sequence.
In this way, a single-bit-flip Hill Climber is first lead towards the first string of the path, and then through the whole path. The reason that this fitness function/path is called a *long*path is that the length of the sequence is *exponential*[43] in l, the length of the strings. Because of its length, Horn's longpath is an intractable search problem for the single-bit-flip Hill Climber.

More generally, we define a fitness function to be a *longpath* for a given iterative search procedure when the expected run of the procedure produces a path whose length is asymptotically lower-bounded by an exponential of the problem size. Moreover, we require that enough information is present in the fitness function to guide the algorithm at each stage of the run. This requirement excludes the search for a "needle-in-a-haystack" on an exponentially large part of the search space. It also excludes the procedure from getting stuck

[43] We will call all asymptotic behavior beyond a polynomial growth *exponential*, though we are aware that e.g. $2^{\sqrt{l}}$ and 2^l have different asymptotic growth rates.

in a local optimum and spending a long time trying to escape from it.
Rudolph [Rud96] presents a generalization of Horn's longpath to a class of fitness functions called *long k-paths*. Let us briefly recall the definition and some of the properties of the long k-path. The long k-path $P(l,k)$ is defined by a recursion on the length l of the strings, as explained in I.3.1. Long k-paths share the following properties:

- The Hamming distance between two consecutive points of the path is 1.
- A mutation of $i < k$ bits can only lead to a point which is i positions away from the original point. To jump over more than k consecutive positions in the path, all the bits of at least one of the blocks $2j+1, \ldots, 2(j+1)$, for $j \in \{0, \ldots, (l-1)/k\}$, have to be flipped. Such a jump we call a *shortcut*.
- The length of the long k-path is given by $|P(l,k)| = (k+1)2^{(l-1)/k} - k + 1$.

Rudolph [Rud96] gives $\mathcal{O}(l^{k+1}/k)$ as an asymptotic upper bound on the expected time spent by a (strictly elitist) (1+1)-ES with $1/l$ mutation rate on a long k-path. For a fixed value of k, this bound is polynomial in terms of the string length. It shows that for large values of l, the (1+1)-ES is a lot quicker than the single-bit-flip Hill Climber in optimizing the path, the only difference between the two algorithms is that the (1+1)-ES can do more than one bit-flip at the same time. Note that in the case of k being a function of l, as in $k = \sqrt{l}$, the upper bound changes to an uninteresting exponential.
This result has been improved in section I.3, where proposition I.3.2 gives the asymptotic statistical distribution of the time T spent by the (1+1)-ES on the path $P(l,k)$. Let us only summarize the expected values of T:

$$E[T] \underset{l\to\infty}{\simeq} \frac{el^{k+1}}{k\ln(l/k)}, \qquad \text{for fixed values of } k, \tag{99}$$

$$\ln E[T] \underset{l\to\infty}{\simeq} l^{\alpha}\ln(l), \qquad \text{for } k = l^{\alpha}, \text{ with } \alpha < 1/2, \tag{100}$$

$$\ln E[T] \underset{l\to\infty}{\simeq} l^{1-\alpha}\ln(2), \qquad \text{for } k = l^{\alpha}, \text{ with } \alpha \geq 1/2. \tag{101}$$

Equation 101 shows that for $k = \sqrt{l}$ the (1+1)-ES indeed spends an exponential amount of time to reach the optimum. Note that experiments cannot reflect the asymptotic behavior, since they are necessarily restricted to small values of l.
Knowing the expected time needed by the (1+1)-ES to solve the long k-path, we can expect that for fixed k and large l, a GA with the stochastic $1/l$ mutation should not need more than polynomial time.

As for the behavior of a GA with one-point crossover and no mutation, experiments carried out in [HR96a] demonstrate that a GA with complementary crossover (a one-point crossover between a string and its complement) exploits a kind of Royal Road structure present in the long k-path. This results in the occurrence of many shortcuts, and a growth rate of l^2 for the time to reach the optimum of the long k-path. Höhn and Reeves expect that the complementary crossover approximates the one-point crossover if the latter is used in combination with a big enough population size. However, no conclusion can be drawn about the asymptotic behavior of this GA.

Knowing that there exist longpaths for the (1+1)-ES, we wonder if we can construct longpaths for population based GAs using (one-point or uniform) crossover and stochastic bit flipping mutation. From the discussion above we know that they must be highly dependent on the choice of genetic operators: a GA is very unlikely to "follow" a path designed for a Hill

Climber, where successive points are at Hamming distance 1 from each other. Considering these facts, we propose separating the problem of constructing a longpath for the GA in two parts. Firstly, we design a path which the GA simply "follows". This problem is studied in Section I.4.2. As a second step (section I.4.3), we turn this *controlled path* into a longpath for the GA.

I.4.2 CONTROLLED PATHS

I.4.2.1 Motivation

A natural direction for the last part of this chapter is the extension of the result of the previous section (the existence of longpaths for stochastic bit flipping mutation) for other types of algorithms, e.g. population based algorithms involving also some crossover operators. However, the problem of the existence of longpaths is ill-posed for population based EAs. Hence, before investigating population based algorithms, the notion of "following a path" has to be extended for populations:

By definition, GAs are stochastic population-based algorithms. It makes little sense to exactly control the composition of the successive populations. Neither is it clearly defined what it means for a GA to arrive at some point in the search space. Is it that the point is contained in the population? Or do we require that the population is centered around this point? This is why up to now we have quoted the use of the verb *follow* in the context of GAs. The following paragraphs will gradually make the terminology more precise.

A GA is said to follow a *controlled path* when the best individual of the population visits a number of predefined points in the search space in a predefined order during the course of its run. Moreover, the expected number of iterations between two successive visits has to be predictable, and the variance on this number should be small. These predefined points are called *landmarks*, the exact meaning of the verb *visit* will be explained shortly.

For such a controlled path to exist, it has to lead a GA through a number of different areas of the search space, each couple of successive areas having exactly one landmark in common. We note that

- these areas have to be different enough not to allow a GA to make shortcuts from one area to another;
- the fitness distribution of each area has to force the search into the direction of the next landmark which is at the beginning of the next area to be traversed.

The fact that a GA has to traverse a number of different areas of the search space helps to define what it means for a GA to *visit* a landmark. When a GA is lead to landmark m^k, its population becomes more homogeneous and focused towards the point m^k which is the only common point of the two different areas A^k and A^{k+1}. It is likely that even before the landmark m^k is detected, the population discovers points of the new area A^{k+1}. This will immediately lead the whole population into this new area, towards landmark m^{k+1}. In this way we can say that the transition from one area to another is the same as visiting the landmark which separates the areas, regardless of the fact that the landmark was detected, or that the population converged to this point.

The notion of different areas is related to the inherent neighborhood metric of a GA, defined through its crossover, mutation and selection operators. Given the fact that most of the members of the current population are in area A^k, and the others in areas with a lower index, is it likely that the next generation of the algorithm contains strings from an area with an index higher than $k+1$? If this is the case, a GA might skip a number of landmarks to end up in an area with a much higher fitness level. The interplay between the definition of the

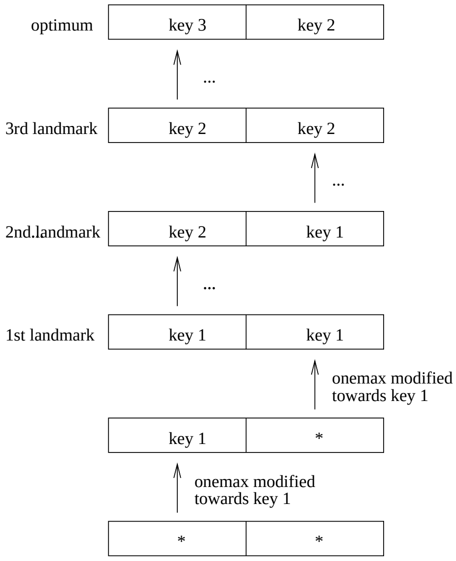


FIG 17 : Example of a controlled path fitness function, defined with $r = 3$ keys. We distinguish 4 different areas separated by 3 landmarks. The areas are the schemas key^1*, $*key^1$, key^2* and $*key^2$. The path starts at the modified OneMax problem leading to the first landmark.

path and the types of operators must ensure that it is unlikely that the operators produce strings which are at higher fitness levels (i.e., that it is unlikely that shortcuts to areas with higher indices occur).

I.4.2.2 A system of keys

Before proposing a way of creating areas at different fitness levels and corresponding landmarks, we introduce some notations. If $s \in E$, then we use s_{L} resp. s_{R} to denote the $l/2$ first resp. last bits of the string. The concatenation of substrings is realized by writing the substrings one after the other. The normalized Hamming distance between two strings $s^1, s^2 \in E$ is denoted by $\mathrm{d}_n(s^1, s^2)$. We have that $\mathrm{d}_n(s^1, s^2) = \mathrm{d}(s^1, s^2)/l$.
The fitness function depicted in Fig. 17 satisfies the necessary requirements for a controlled path. It depends on a system of *keys* to generate the different areas and the landmarks separating the areas. A key is a string of length $l/2$; a set of r keys $\{key^i\}_{i=1}^r$ determines $2r-2$ different areas of the search space, separated by $2r-3$ landmarks, in the following way. When none of the keys appear as the left or right half substring of a string s, the fitness $f(s)$ is defined as a OneMax function on the left half of s, modified towards the first key:

$$f(s_{\mathrm{L}}s_{\mathrm{R}}) = \mathrm{d}_n(s_{\mathrm{L}}, key^1). \tag{102}$$

When key i is spotted on the left half of the string s, the GA is lead towards the $(2i-1)$-th landmark, which is the string $key^i key^i$:

$$f(key^i s_{\mathrm{R}}) = 2i - 1 + \mathrm{d}_n(s_{\mathrm{R}}, key^i), \qquad i \in \{1, \ldots, r-1\}. \tag{103}$$

Analogously, when a key j is spotted on the right half of the string s, the GA is lead towards the $2j$-th landmark, which is the string $key^{j+1} key^j$:

$$f(s_{\mathrm{L}} key^j) = 2j + \mathrm{d}_n(s_{\mathrm{L}}, key^{j+1}), \qquad j \in \{1, \ldots, r-1\}. \tag{104}$$

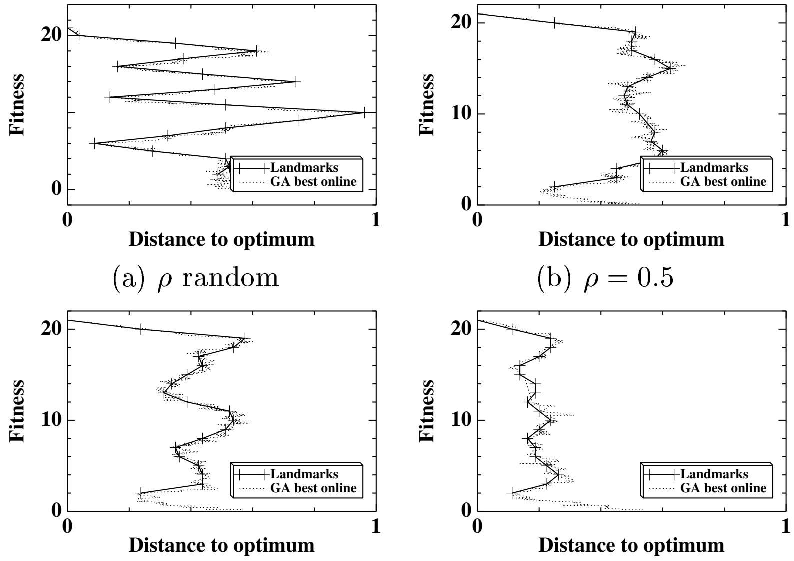
The values of strings consisting of two equal or consecutive keys are set to:

$$f(key^i key^i) = 2i, \qquad i \in \{1, \ldots, r-1\}, \tag{105}$$
$$f(key^{i+1} key^i) = 2i+1, \qquad i \in \{1, \ldots, r-1\}. \tag{106}$$

When a string consists of two keys which do not fall under (105) conditions, or key^r is involved, then either Eq. (103) or Eq. (104) is applied, priority being given to the key with highest index, ignoring index r.
This finishes the construction of the controlled path f depending on the set of keys $\{key^i\}_{i=1}^r$. Note that the optimum of f is at $f(key^r key^{r-1}) = 2r-1$.
The function f defines a path through $2(r-1)$ different areas. Each area is a schema of order $l/2$. The choice of taking $l/2$ as the order of the schemas is arbitrary, of course, as is the choice of 2 blocks of consecutive bit positions. As a matter of computational efficiency, it does not seems appropriate to generalize the schemas to arbitrary sets, although there is no theoretical objection.
The shortcuts made by the GA can be quantified: an increase of $n \geq 2$ in the fitness value of the best individual indicates that $n-1$ areas have been skipped. A GA is said to follow the controlled path when, over the run of the GA, the fitness value of the best individual does not increase from one generation to the next with a value strictly greater than 1.

I.4.2.3 An experiment with random keys

Let us now experiment with randomly chosen keys, and an GA with population size 100 with stochastic $1/l$ mutation and different crossover operators (one-point, two-points and uniform) at rates equal to 0.5.
We introduce one parameter ρ in order to tune the proximity of the keys in Hamming distance. Given a value $\rho \in [0,1]$, we generate a key by independently assigning 1's with probability ρ and 0's with probability $1-\rho$ as the key's bit values. In this way, we create a set of r keys. The expected normalized Hamming distance between two distinct keys is $2\rho(1-\rho)$. However, we are not guaranteed a minimal distance.
Figure 18 shows the results of the GA subject to four controlled paths, each created with a different value of ρ. The pictures show the paths in the fitness versus distance-to-optimum space. To give a visual idea of the path to be followed, we connected the crosses (+) which represent the 20 different landmarks, created with $r = 11$ keys. The best individual seen by the GA at each generation is represented by a dot. In this way, we can visually follow the GA's walk through the search space.
Picture (a) shows the case where each key is generated with a randomly chosen value of $\rho \in [0,1]$. No shortcuts are taken by the GA. With the value of ρ increasing, as in (b), (c) and (d), the expected normalized Hamming distance between two distinct keys decreases, and the keys become more similar to each other. In the case of $\rho = 0.9$, the keys have come so close that the GA can make a number of shortcuts. If we keep ρ at 0.9 and increase the number of keys, then the shortcuts dominate. The case of $\rho \approx 0.5$ is more interesting. Experiments show that the GA, with uniform or one-point crossover, reliably follows a path consisting of 2000 landmarks (1001 keys).
The results of the random key experiment are very encouraging. By explicitly designing the keys rather than randomly drawing them, the distance between any pair of keys can be made larger than a minimal value.



(a) ρ random
(b) $\rho = 0.5$
(c) $\rho = 0.7$
(d) $\rho = 0.9$

Fig 18 : The best individual known to the GA at each generation plotted in the fitness versus distance-to-optimum space of four controlled paths. The schema key^1* and the 19 landmarks, created with $r = 11$ keys, are interconnected and represented by crosses (+). In cases (b), (c) and (d), $2\rho(1-\rho)$ is the expected normalized Hamming distance between two distinct keys. With ρ increasing, the keys become more similar to each other. In case (d), the keys have come so close that the GA can make a number of shortcuts. In picture (a), each key is generated with a randomly chosen value of $\rho \in [0,1]$. No shortcuts are allowed to the GA.

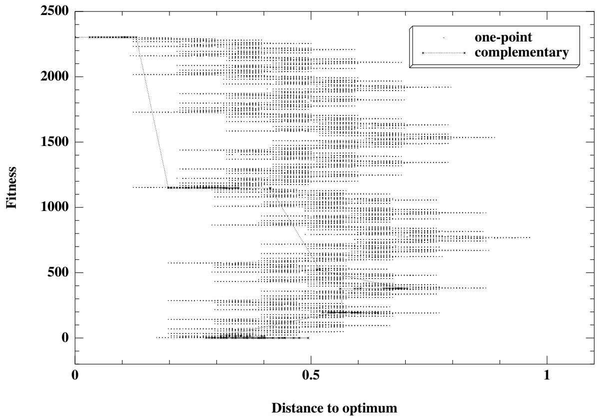


FIG 19 : *Controlled path with $l = 162$ and keys from $S(81, 8)$: the best individuals found so far over the course of a GA run plotted in the fitness–distance to optimum space. A GA using complementary crossover (in addition to stochastic $1/l$ bit flipping mutation) takes huge shortcuts, whereas a GA using one-point (resp. unifom) crossover with stochastic bit flipping mutation, takes none (fitness improvement at each step is always less than 1).*

I.4.3 CANDIDATE LONGPATHS FOR THE GA

All keys at a minimal distance from each other

It is easy to observe that Rudolph's long k-path contains a sub-path $S(l, k)$ whose points are all at least a Hamming distance k from each other. The points are at positions $1 + \alpha(k + 1)$ ($\alpha \in \{0, 1 \ldots 2^{(l-1)/k}\}$) in the path, and include none of the bridge points. The length of this sub-path is $|S(l, k)| = 2^{(l-1)/k}$.

The structure of the sub-path $S(l, k)$ is exactly the same as that of the long k-path with respect to the definition and probabilities of shortcuts; each shortcut starting from a string in $S(l, k)$ ends up in $S(l, k)$ again. If the (1+1)-ES enters the path $S(l, k)$, then at least one shortcut is required to reach the optimum without following the whole path. Hence, the convergence time is lower bounded by $T \geq el^k$.

Recall Rudolph's asymptotic bound of $\mathcal{O}(l^{(k+1)}/k)$ for the expected time of the (1+1)-ES on the long k-path. It is due to the fact that in the worst case, $(l - 1)/k$ shortcuts are needed to reach the optimum. In the case of the sub-path $S(l, k)$, the number of shortcuts needed to reach optimum remains the same, and hence the time spent by the (1+1)-ES on the sub-path is also upper bounded by $\mathcal{O}(l^{(k+1)}/k)$.

Combining the two bounds, we can conclude that for $k = \sqrt{l}$, the time spent by the (1+1)-ES on the sub-path is exponential for large l.

The controlled path with keys from $S(l/2, k)$

The sub-path $S(l/2, k)$ defined in the previous subsection perfectly satisfies the conditions for a choice of a set of keys: all strings in the sub-path are at a minimal distance of k from each other, which ensures that distinct areas are different enough.

Figure 19 shows the results of experiments with different algorithms on the controlled path with keys from $S(l/2, k)$.

The complementary crossover GA finds huge shortcuts, while both the one-point crossover GA and the (1+1)-ES fail to take any. Recall that shortcuts are detected by controlling the

degree of increase in fitness at each improvement step. In the case of the one-point crossover GA and the (1+1)-ES, the increase in fitness values is always smaller than 1, meaning that no shortcut happens.
As for convergence time, it takes about $11100\,10^3$ fitness evaluations for the one-point crossover GA, $260\,10^3$ for the complementary crossover GA, and $1500\,10^3$ for the (1+1)-ES. Hence, in the number of evaluations, (1+1)-ES is more advantageous than the one-point crossover GA.

I.4.4 A LONGPATH FOR THE (1+1)-ES

In this section we consider a slightly modified version of the controlled path with keys from $S(l/2, k)$. Points with key^i on the left-hand (respectively right-hand) side of the string are moved out of key^i's fitness level (and receive a fitness value of at most 1) unless both

- *First condition*: their right-hand (respectively left-hand) side is at distance $d \leq k$ from the next key in the path, which is key_i (respectively key_{i+1}), and

- *Second condition*: their right-hand (respectively left-hand) side is at distance $d \leq k+k/2$ from key_{i-1} (respectively key_i) .

These conditions allow one to control the first string of the modified OneMax process. As a result, the process remains at a minimal distance of k from all the other keys of the path (apart from the one it leads to, and the one it comes from). This fact is due to the geometry of the strings of the $S(l/2, k)$ path.

Now consider the modified OneMax process on the right-hand side to key^i (respectively left-hand side to key^{i+1}). Note that if this process starts at string $key^i key^{i-1}$ (respectively $key^i key^i$), then the set of points that the process can visit necessarily fall under both conditions.

As a consequence, these conditions don't change the dynamic of the modified OneMax process: Whatever the string of the path it starts from is, the process can only visit points that fulfill the conditions. Hence, the adjonction of these conditions don't change the expected convergence time of the modified OneMax.

I.4.5 The process $X(t)$

Let us denote by $\mathbf{e}_i = key^1 key^1$ the first string of the modified controlled path and by $\mathbf{e}_f = key^r key^{r-1}$ the last one.
The (1+1)-ES follows a process $X(t)$ starting at $X(0) = \mathbf{e}_i$. At time $t \in \mathbb{N}$, a mutated string $X'(t)$ is built from $X(t)$ by inverting each bit of $X(t)$ independently with probability ℓ^{-1}. If $X'(t)$ is fitter than $X(t)$, then $X(t+1)$ becomes $X'(t)$, otherwise $X(t)$ is copied into $X(t+1)$. The path's last string $\mathbf{e}_f$ is the unique absorbing state of the Markov chain X.
In the following, T denotes the first hitting time of the string $\mathbf{e}_f$ by the process X. We prove that T is exponential in ℓ and that the process ends before any shortcut happens.

Proposition 1 *Let us assume that $k = [(\frac{\ell}{2})^\alpha]$, were $[x]$ denotes the integer part of x.*
1. $\lim_{\ell \to \infty} \mathbb{P}[G1] = 1$, where G1 is the event "no shortcut to a fitter areas happens during the process X(t)".

2. If $\alpha \geq 1/2$, then

$$\lim_{\ell \to \infty} \frac{2}{\ell^{1-\alpha} \ln(2)} \ln \mathbb{E}[T] = 1, \tag{107}$$

where T is the first hitting time of the solution $\mathbf{e}_f$ by the process $X(t)$ starting at $\mathbf{e}_i = key^1 key^1$.

Before turning to the proof we present some results on the waiting times of particular types of shortcuts.

I.4.5.1 Waiting times for shortcuts on the modified OneMax side

Let us focus on what happens on the side of the string where the modified OneMax problem is being solved, starting from key^i and going to key^{i+1}.
There are, at most, $\frac{l/2-1}{k} - 1$ keys of $S(l/2, k)$ that are likely to be approached at distance k by the substring of X(t) that is solving a OneMax process to key^{i+1}. These keys are at distance k from key^{i+1}. We denote them here by key_j^{i+1}, for $j \in \{1, ..., \frac{l/2-1}{k} - 1\}$. Suppose key^i equals

$$0000\,0000\,0000\,0000\,0$$

and key^{i+1} equals

$$0000\,0000\,\mathbf{1111}\,0000\,0,$$

then the keys key_j^{i+1} obtained by possible shortcuts are

$$0000\,0000\,\mathbf{1111}\,\mathbf{1111}\,0,$$
$$0000\,\mathbf{1111}\,\mathbf{1111}\,0000\,0,$$
$$\mathbf{1111}\,0000\,\mathbf{1111}\,0000\,0.$$

Let us introduce the family $\mathcal{S}$ consisting of the $\frac{l/2-1}{k}$ possible shortcuts from above, individually denoted by $\mathcal{S}_j$, with $j \in \{1, \ldots, \frac{l/2-1}{k}\}$. A shortcut $\mathcal{S}_j$ can be seen as an operator which at time t inverts the (at least k) bits of string $X(t)$ such that the process jumps to key_j^{i+1}.
Note that for the shortcuts $\mathcal{S}_j$ (occurring on the modified OneMax side), the number and positions of the flipped bits (that lead to a fitter key) may vary at each step. Yet, each of these shortcuts flips at least k bits; and, at every step, the number of possible shortcuts is bounded above by $\frac{l/2-1}{k}$.

We also introduce the family $\mathcal{S}'$ which consists of all the shortcuts which invert at least the k bits corresponding to some $\mathcal{S}_j$.
Note that some of the shortcuts of the family $\mathcal{S}'$ do not give rise to a jump for the process $X(t)$. Yet, it is clear that $\mathcal{S} \subset \mathcal{S}'$ and $\mathcal{S}'$ actually contains many more shortcuts than $\mathcal{S}$. Moreover, the waiting times for one particular event of $\mathcal{S}$ and for an event of $\mathcal{S}'$ are almost the same when $k \sim (\frac{l}{2})^\alpha$; they follow an exponential law with mean of the order of $(l)^{(l/2)^\alpha}$.

Lemma 2 *If $l \gg 1$, then the waiting time $T_{\mathcal{S}'}$ for some shortcut of the family $\mathcal{S}'$ obeys an exponential distribution with mean $(l)^k k/(l/2-1)$.*

Proof. Although the $T_{\mathcal{S}'_j}$ are integer-valued, we describe their respective distributions using the notion of "densities", which is usually reserved for real-valued random variables. For example, the statement "$T_{\mathcal{S}'}$ has an exponential density with mean m" should be understood as

$$\mathbb{P}(T_{\mathcal{S}_j} = t) = \frac{1}{el^k} \exp -\frac{t}{el^k} \text{ for any integer } t$$

At each time $t \in \mathbb{N}$ and for any fixed j, the probability that some shortcut which inverts at least the k bits that lead to a particular fitter key key^j occurs is:

$$P_{\mathcal{S}'_j} = (l)^{-k}.$$

This proves that the statistical distribution of the first time $T_{\mathcal{S}'_j}$ when one of these shortcuts occurs is given by:

$$\mathbb{P}(T_{\mathcal{S}'_j} \geq (l)^k t) = \exp(-t).$$

The minimum of $\frac{l/2-1}{k}$ independent exponential variables with mean m obeys an exponential distribution with mean $m\frac{k}{l/2-1}$:

$$\mathbb{P}\left(\inf_{j=1,\ldots,((l/2)-1)/k} T_{\mathcal{S}'_j} \geq l^k t\right) = \mathbb{P}\left(T_{\mathcal{S}'_1} \geq l^k t\right)^{\frac{l/2-1}{k}} = \exp\left(-t\frac{l/2-1}{k}\right)$$

□

Remark I.4.1 *The above analysis only considers the OneMax side of the process X(t). Before studying the shortcuts' occurrence from both sides of the process, this remark investigates the influence of the presence of non-encoding genes in the string (as if we remove the events corresponding to shortcuts on the alternate part of the string, i.e the part containing a fixed key). We show that convergence time is then multiplied by a factor of* $e^{1/2}$ *asymptotically. This is a general result, which proves that the presence of non-encoding genes in the string, doesn't change the asymptotical order of convergence time.*

Proof.
Since mutations at different moments in time are independent, we can decompose T as a sum of improvement steps. We write $T = \sum_i \tau_j$, with τ_j the time required by some process to improve the fitness value. The variable τ_j^{joint} is like τ_j
but is used in the context of a joint process with a fixed key at one half of the string.
If a mutation occurs in the part of the string containing the key, there is no improvement in fitness. The probability of this event is

$$p_{\text{fail}} = 1 - (1 - \frac{1}{l})^{\frac{l}{2}}. \tag{108}$$

The probability of the event "exactly i failures" is

$$\mathbb{P}[i \text{ times}] = p_{\text{fail}}^{i-1}(1 - p_{\text{fail}}), \tag{109}$$

and we have

$$\mathbb{E}[\tau_j^{\text{joint}}] = \sum_i \mathbb{P}[i \text{ times}]\, i\, \mathbb{E}[\tau_j] = \mathbb{E}[\tau_j]\left(\frac{1}{1 - p_{\text{fail}}}\right). \tag{110}$$

For $l \gg 1$, this gives the desired result

$$\mathbb{E}[T^{\text{joint}}] = \left(\frac{1}{1 - p_{\text{fail}}}\right) \sum_i \mathbb{E}[\tau_j] = e^{1/2}\mathbb{E}[T]. \tag{111}$$

□

I.4.5.2 Waiting times for shortcuts on the controlled path

Let $\mathcal{S}^{\text{joint}}$ be the family of shortcuts on the controlled path. A shortcut ${\mathcal{S}^{\text{joint}}}_j$ of $\mathcal{S}^{\text{joint}}$ leads to a string on the controlled path, which is one key from $S(l/2, k)$ on one side and a substring fulfilling conditions 1 and 2 (in section I.4.4) on the other side.

Considering the facts that:

1. All keys from the skeleton $S(l/2, k)$ are at distance k – at least – from each other.
2. A substring on the modified OneMax side is at distance k – at least – from all the keys of the skeleton (apart from those involved in the OneMax solving).

We conclude that

1. A shortcut ${\mathcal{S}^{\text{joint}}}_j$ on the controlled path implies – at least – a k bit flip on either the right or left sides of a string on the path.
2. Each time, the number of possible shortcuts on the right (resp. left) side is bounded above by $\frac{l/2-1}{k}$.

This implies that the waiting time for a particular shortcut on the controlled path ($T_{\mathcal{S}_j'^{\text{ joint}}}$) is the minimum of two exponential variables with mean l^k: the waiting times for shortcuts at either sides of the the string (as the processes at right and left sides of a string are independent, due to the mutation definition). Hence

Lemma 3 *the law of* $T_{\mathcal{S}'^{\text{ joint}}}$ *is exponential with mean* $\dfrac{l^k}{2}\dfrac{k}{l/2-1}$.

Using the shortcuts of the family $\mathcal{S}'$ as defined above, the proof of Proposition 1 follows in the same manner as that of proposition I.3.2.

Remarks
1. When $\alpha < 1/2$, the convergence time is still exponential, but shortcuts occur. This can be proved by an estimation of T_{ws} using the standard large deviations principle for the sum of independent random variables [Str84, theorem 3–8].

2. A generalization of the latter statement (convergence time on the controlled path) to the $(1+\lambda)$-ES is easily made. In the same way as for the long k-path [GKS98], there is no gain in the number of evaluations when λ increases.
In the case of an elitist SGA, we have to consider the influence of the strings which due to crossover or mutation fell off the path, and are still in the population. This problem can be studied in a Markov chain model.

I.4.6 MORE MOTIVATION

The choice of a system of landmarks and keys, which is at the core of our controlled path, can be motivated from the physics of (spin) glass. Real glass are commonly modeled by spin glass, whose terminology we will mostly use.

Solid glass is constructed by rapidly cooling down a liquid from above its critical temperature. The result of this process is a material which consists of microscopic domains. Due to frustration (see e.g. the introduction of [Cho86]), spin domains, which did not have the time to properly relax into thermal equilibrium, are formed. If, on the contrary, the glass is cooled down slowly enough from its critical temperature, the domains have the time to merge together into one big crystal.

Each spin domain is characterized by a "set of spin correlations" [Hil90]. We can interpret such a set as a key; it is one of the possible solutions to the local frustration problem. The optimal solution with respect to free energy depends on the temperature of the spin glass; a huge majority of domains have sub-optimal free energy. Since the difference in energy between a sub-optimal state and the optimal state is small, rebuilding a domain's key into a different key is a costly process, which is either achieved by a strong change in temperature or by waiting a long time (in effect, by literally waiting ages). (A nice example of this phenomenon is the glass of medieval cathedrals, which is now reaching its final equilibrium state after 400 years of slow decay. This final state is much more crystalline; the glass has become very fragile.)

Two more aspects of the keys are worth mentioning. Firstly, neighboring domains have incompatible keys and hence cannot merge, as would be the case when frustration is absent. Secondly, due to the slow evolution of keys a glass records its own thermal history. As a consequence, a glass contains a mixture of many different keys.

The construction of our controlled path, using a sequence of alternating keys, may seem very contrived. The spin glass example, however, shows that a system of keys occurs in nature. Such a system is very effective for preventing shortcuts in the optimization process, which is also the reason why we are using keys.

Finally, remark that with spin glass, keys occur as a consequence of the frustration problem. This shows the need for further study of this concept in the context of function optimization. It is also worth investigating whether a longpath could be created using only sub-optimal solutions of a frustrated instance of an NP-hard optimization problem as keys.

I.4.7 CONCLUSION

We propose a generic way for constructing controlled paths for GAs. GA populations are lead through a number of consecutive areas (hyperplanes of order $l/2$) in the search space. While we have no formal proof yet of this controlled path being a possible longpath for population based EAs, experiments with uniformly chosen hyperplane keys show no shortcuts to fitter areas. This empirical observation holds for different EA instances using one point or uniform crossover and deterministic or stochastic $1/l$ mutation:

A modified OneMax problem leads the SGA from the beginning of the area to the beginning of the next one. This construction is related to the choice of genetic operators, hence different GA instances (yet to be determined) may not follow these controlled paths.

Moreover, we prove that the controlled path with keys from the *long k-path*, is a longpath for the (1+1)-ES using the stochastic $1/l$ bit flipping mutation, in the sense that convergence time is exponential and that the solution is reached before any shortcut (to fitter areas) happens. This provides a second longpath for both the deterministic and stochastic bit flipping mutations, with a path that contains much more points than the long k-path does.

Controlled paths provide an example of smooth and unimodal[44] fitness landscapes, on which GA convergence time can be exponential. An interesting but currently unanswered question is whether real world problems can present such a longpath structure? The analogy with spin glass shows that there exist physical systems which evolve very slowly towards an equilibrium state, through successive unstable states.

While controlled paths improve our practical knowledge about possible GA dynamics, it also appeals for the construction of GA instances (choice of operators and selection schemes) that don't follow the path. Recall that increasing the population size and selection pressure does not help convergence. On the other hand, selection schemes that maintain diversity in the population (e.g. crowding[DeJ75], sharing [GR87]) coupled with adaptive operators may help to design more powerful EAs. This shall be left for future work: "Can we break the inherent symmetry of a Controlled Path problem without *a priori* knowledge on its structure".

However, a first step towards such a goal is to design and implement practical tools that report information about actual EA dynamic during a given run. The commonly used convergence curve, translated in the fitness space, cannot detect whether a GA is following a smooth controlled path or whether the GA is walking up and down on a rugged landscape. This is the goal of Section II.2 of next chapter: describing and comparing EA behaviors.

[44] no local optimum but the global one exists

I.5 Conclusion of Chapter I

The first part (sections I.2 and I.3) of this chapter compares two types of binary mutation operators, namely the stochastic c/l mutation (all bits are flipped independently with the same probability c/l where l is the string length) and the deterministic 1-bit mutation, which for each bitstring, exactly flips 1 uniformly chosen bit. These operators yield the same average mutation rate for $c = 1$, but only the stochastic mutation allows simultaneous bit flips.
A systematic comparison has made possible the derivation of the first hitting times of the optimum, in different situations:

- random walks directed by mutation (i.e. without selection), by considering a (1,1)-ES algorithm.
- two extreme cases of unimodal functions (respectively Onemax and *long k-path*), using the elitist (1+1)-ES.

One of the outcomes of this comparison, is, the existence of controlled longpaths for (1+1)-ES using either mutation. This in turn raises the same issue for crossover operators and population based EAs. Section I.4 gives partial answers this issue.

The results of this chapter provide some lower and upper bounds on the complexity of simple search procedures in l dimensional spaces, for big l values:
The time to convergence of mutation only Hill Climbers, ranges from $l \ln l$ to an exponential of l (respectively on the unimodal OneMax and Long k-path functions). In the OneMax case, convergence is a factor $\frac{e^c-1}{c}$ quicker for the deterministic bit flipping mutation than for the stochastic mutation, whereas the opposite happens for the Long k-path problem (exponential versus polynomial time in favor of the stochastic mutation). On the other hand, random walks directed by either stochastic or deterministic mutations, require the same amount of time, when considering only 'active' steps.

within the class of unimodal fitness functions, the time to convergence of a hill climber directed by the commonly used stochastic mutation, ranges from linear to exponential in l. The stochastic c/l mutation is outperformed by the deterministic 1-bit-flip mutation in the first unimodal function case, but not in the second. On the other hand, the efficiency of these mutations is the same for random walks, where the goal is to reach a given target point.
[45]

We also demonstrated that the trajectory of EAs[46] follow a user predefined sequence of hyperplanes of order $l/2$, on *controlled path* fitness functions, purposely constructed. This improves our practical knowledge about possible EA dynamics.

Last but not least, a number of new results has been obtained,

1. The multiplicative constants for the complexity of the first hitting times on the Onemax problem (resp. the *long k-path* problem with fixed k). This allows to actually compare the hitting times with both mutations in the Onemax case, an original result.
2. The behavior of the stochastic bit flipping mutation is derived on *long k-path* for variable k values. This allows to prove the existence of controlled longpaths for this mutation.

[45] if we consider mutation steps that flip at least one bit in a bitstring

[46] We experimented different population based EA instances using crossover and both deterministic and stochastic bit flipping mutations

3. The complete distribution of hitting times is derived in the onemax case (resp. *long k-path*). This allows to compute the variance, unknown up to now, on these problems. A direct consequence is that parallel independent processes scarcely improve the time to convergence in the Onemax case (small normalized variance), while it significantly improve the time to convergence in the case of *long k-path* problems (the high variance makes it more likely that a run achieves rare short time of convergence).

This last result raises a natural question for future work. Knowing that parallel independent processes scarcely improve the time to convergence on a Onemax, can one expect to improve the time to convergence by using a positive migration rate between parallel processes, or populations eventually with crossover and niching? The statistical tools of physics are able to approximate the fitness distribution of successive populations on the onemax problem, at this point they might be technically able to address this question.

Chapter II

Indicators of EA Performance

Plan du Chapitre II

L'un des enjeux majeurs en AG, est d'identifier les fonctions faciles ou difficiles à optimiser. Plusieurs classes de fonctions fitness sont étudiées dans la suite, montrant que la notion de difficulté ne peut se limiter à des indicateurs sommaires du paysage de fitness. Le vrai problème reste la caractérisation d'une classe de fonctions 'faciles' (terme justement à definir) pour l'AG.

II.2 Base de Comparaison des Résultats de l'AG

Définir la notion de *comportements similaires de l'AG* est une question à part entière. Une méthode de comparaison de la trajectoire et des temps de convergence de l'AG est proposée. Les résultats de l'AG sur un problème donné sont d'abord comparés à une base de référence qui depend de l'AG en question.
Le recours à cette base permet de tenir compte du biais introduit par l'initialisation et les divers paramètres, quand il s'agit de comparer les résultats de l'AG sur avec des fonctions fitness ou des représentations différentes.

II.3 Indicateurs Statistiques des Paysages de Fitness

Cette section propose une étude intensive de quelques propriétés statistiques sommaires du paysage de performance (corrélation, épistasie), montrent une totale décorrélation de celles-ci avec les résultats (qualité, vitesse) de convergence de l'AG sur différentes classes de fonctions fitness.
La corrélation (resp. l'épistasie) caractérise une classe de fonctions bien trop petite, à savoir, celle des fonctions linéaires relativement à la distance à la solution (resp. à variables séparables). La figure 25 montre les relations entre plusieurs classes (plus larges) et faciles pour l'AG (ex. monotones, unimodales, non-trompeuses).

II.4 Relation Fitness–Distance

Cette section s'intéresse à la relation entre fitness et distance à l'optimum, au-delà de la seule valeur de corrélation. Deux classes de de fonctions partageant la même relation sont construites. Le comportement de l'AG au sein d'une même classe se trouve être complètement imprévisible, illsutrant que la connaissance de la relation entre fitness et distance à l'optimum est bien trop insuffisante pour caractériser le comportement de l'AG.

II.5 Comparaison des Procédures d'Initialisation

Le graphe de corrélation peut cependant aider à choisir une procédure d'initialisation, facteur important de rapidité et d'efficacité pour l'AG. En effet, ce graphe permet d'évaluer la diversité d'un échantillion d'individus, de par l'etendue et la distribution des distances et des performances.
Cette partie propose et compare des procédures d'initialisation originales basées sur différents critères d'uniformité dans le cadre binaire. Mais les résultats les plus spectaculaires sont obtenus avec les représentations de taille variable, où la procédure d'initialisation depend de beaucoup de paramètres interdépendants (section III.6.2).

II.6 Indicateurs Statistiques des Opérateurs d'Évolution

Si le comportement de l'AG dépend directement du paysage de fitness, on ne peut ignorer le rôle crucial des opérateurs sur ce paysage.
Plusieurs opérateurs sont comparés, pour identifier les propriétés essentielles, dans une logique de maximisation du profit à court terme.
Il est important de noter la dualité représentation/opérateurs : changer de représentation revient aussi à changer d'opérateur. La même approche peut servir pour comparer les différentes représentations (section III.6.3).

Publications: La section II.2 est une version modifiée d'un article publié [6], les sections II.3 et II.4 sont des versions modifiées de deux travaux (soumis) [7,11] réalisés en collaboration avec Bart Naudts. Enfin, les deux dernières sections (II.5 et II.6) sont des versions modifiées de deux articles publiés [3,4].

Cela explique certaines redondances (de l'état de l'art) de ces sections. Même si les versions modifiées limitent ces redondances et rajoutent des références croisées, les sections peuvent quasiment être lues indépendamment.

Overview of Chapter II

This chapter presents a series of work devoted to the analysis of indicators of landscapes difficulty in the framework of binary Evolutionary Algorithms. All subsections are edited versions of published (or submitted) work that can be read independently.

II.2 Ground Basis for Comparison of EA Behaviors

The concept of *similar behavior* for two different instances of EAs on two different problems is difficult to define. Some original indicators are proposed here, related to both the trajectory of the best individual in the population during a run, and the time that was necessary to reach the final best individual. Moreover, the genotypic distance to the solution is used as a helper function: the same indicators computed from the same initial populations are used as scaling factors, thus decreasing the amount of bias due to the different representations, initialization procedures, and parameters of the EAs to be compared.

II.3 Statistical Indicators of Fitness Landscapes

Some "indicators" of landscapes difficulty have been proposed in the GA's literature, such as Fitness Distance Correlation (FDC), epistasis variance and Site Wise Optimization indicator (SWO). Based on those indicators, different classes of functions are defined. Their intersections are carefully examined, resulting in partial order among those classes. Moreover, extensive numerical experiments demonstrate that there is no clear correlation between those classes and the dynamic behavior of common instances of GAs.

II.4 On Functions with a Given Fitness-Distance Relation

Beyond FDC, this subsection addresses the whole fitness-distance relation in connection with the behavior of EAs. Different functions are built starting from given fitness-distance plots. Experimental results using different instances of GAs (1-point or uniform crossover ...) and ESs, exhibit very different convergence behavior, demonstrating that even the whole fitness-distance relation is not a reliable tool to predict GA or ES behavior.

II.5 Initialization and Fitness–Distance Relations

There is, however, a niche for FDC as an indicator for GAs: the choice and tuning of the initialization procedure. This subsection first proposes alternative initialization procedures in the binary framework, based on original points of view on the uniformity criterion. The results of FDC predictions based on those different initialization procedures are then shown to accurately predict the results of actual experiments for different GA instances on some well known artificial test problems. Subsection II.2.6 will later emphasize the usefulness of FDC values for the choice of the initialization procedure in the case of variable length representations.

II.6 Statistical Indicators of Crossover Operators

Previous sections addressed the question of GA performance predictions without directly considering the evolutionary algorithm itself. This subsection puts the operators back in the game: Using first some indicators proposed in the literature, and considering uniform vs 1-point crossover operators, it is shown that even simple binary test function can fool the predictions such indicators suggest. A new indicator based on offspring ranking is proposed, that better matches the actual experimental results.

Publications: Section II.2 is an edited version of a published paper [6], the sections II.3 and II.4 are edited versions of two joint (submitted) works [7,11] with Bart Naudts. Finally, both last sections II.5 and II.6 are edited versions of two published papers [3,4] (References from List of papers after the Appendix).
This explains the possible (state of the art) redundancies in the following sections. Even if the edited versions limit such redundancies and add cross pointers between these works, the sections can almost be read independently from each other.

Chapter II
Indicators of EA Performance

II.1 Introduction

One of the major challenges in the EC field is to characterize which fitness functions are easy for the EA and which are not. Whereas the answer is straightforward in the case of the Hill Climber algorithm, given by the presence of local optima, no rigorous definition of the concept of difficulty is available in the framework of EAs. Yet, many papers implicitly use the notion of difficulty in many cases: When two instances of algorithm parameters are compared on some problem; when two different problems are compared for the same algorithm; or when tools are developed (and validated) in order to measure problem difficulty. Examples of such tools are fitness distance correlation [JF95], correlation length, operator correlation [MdWS91] , epistasis [Dav91a], schema variance [RS95a], hyperplane ranking [DWP95] ...

Early attempts to characterize difficulty [Gol93, Gol89, FM93] propose criteria based on Isolation (needle in a haystack), Deception (schema analysis) and Multimodality. It seems clear that these criteria certainly contribute to problem difficulty for EAs. However, there exist examples showing that the latter two criterion (deception and multimodality) are neither necessary nor sufficient for problem difficulty: Vose and Wright [VW95] present the extreme case of a fully non deceptive function that is difficult for a GA because of an exponential number (in terms of string length l) of stable suboptimal attractors. Wilson [Wil91] proposes a deceptive function, yet straightforward to optimize for a GA. Horn and Goldberg [HG95] construct an easy maximally multimodal function. They also stress the fact that modality and schema-GA-easy analysis are not at all sufficient for describing the behavior of the GA. Finally, longpath problems [HGD94] have been designed to show that unimodal functions can be difficult for a simple 1-bit-flip hill climber because of an exponential path length (and section I.3 proves that an instance of Rudolph's *long k-paths* is a longpath for $(\mu+\lambda)$-ES, and proposes a construction of candidate longpaths for GAs).
Other works unformally use the word 'difficult problem for GAs' when the fitness landscape presents a large number of local optima with large basins of attraction and high barriers between them. This is the case of the NK-landscapes when K value increases. This problem is widely studied in [Kau89], where problem difficulty is related to the repartition of local optima around the global optimum, indexed by their fitness and the size of their basin of attraction. This method, however, requires the knowledge of the whole fitness landscape and is rather time consuming.

On the other hand, many papers [DSG95, CL96] stress the nonsense of speaking of problem difficulty without considering the parameters of the algorithm at hand. Examples are given where the same landscape can be difficult or easy (in convergence speed) to optimize according to the parameters and operators of the algorithm. Along that same line, some tools have been proposed to *a priori* estimate the efficiency of different crossover operators (cf. section II.6 for a detailed state of the art). As in the case of landscape analysis indicators, these tools have been designed to capture key properties that may predict GA-behavior.

The first part of this chapter addresses the key issue of *a posteriori* comparison of EA results. Indeed, before being able to *a priori* measure EA-hardness, one should be able to *a posteriori* compare EA behaviors. The first step is then to extract scalable relevant information from EA run.
Section II.2 proposes different *behavioral measures* in order to characterize EA trajectory and convergence in the search space. The *Hamming fitness* is then proposed as a relative basis that scales behavioral measures, taking into account eventual parameters biases (such as over-fitting initialization that generates close to optimal individuals). The method easily extends for comparing different algorithms or representations.

Having specified how EA behavior and performance can be described, the second part of this chapter investigate some *a priori* predictors of GA performance, and identifies particular properties of landscapes that limits their reliability.
Section II.3 reviews, further investigates and extends some statistical indicators like epistasis variance, fitness distance correlation, proposed in the GA literature. The reference classes (the class the indicator measures a distance or a difference from) of the indicators are identified and compared. Instructive examples are constructed in order to test the reliability of these indicators for predicting *GA behavior*, within the class of site wise optimizable fitness functions.
Of course, the whole Fitness Distance plot (FD plot) gives much more information than the summary correlation value (FDC). Section II.4 investigates the sensitivity of GA (resp. ES) behavior to strings distribution inside the FD plot by studying particular longpath shaped FD relations.
This shows that interpreting the FD plot in order to predict GA (resp. ES) behavior can be very misleading, as various types of functions (easy or difficult for EAs) can share the same FD relation.
Nevertheless, section II.5 shows that if we consider a fixed fitness function, the *fitness* $\times$ *distance to optimum* plot may be useful for comparing initialization procedure. Original initialization procedures are constructed and compared, showing that, beyond GA-hardness, the FD plot can be an efficient tool for choosing problem dependent initialization procedures.

Having investigated the limitations (II.3) and the benefits (II.5) of the landscape based indicators, section II.6 investigates the reliability of some operator based indicators proposed in the literature, to describe operators efficiency. In light of these experiments, the *offspring ranking* method is suggested to compare genetic operators.

II.2 Ground Basis for Comparison of EA Behaviors

Overview This section recalls some of the commonly used methods for comparing EA performance on different problems. This points out the scaling issue when differently distributed fitness functions are compared. *Hamming fitness functions* are then proposed as a basis to relax the dependence of measures that characterize EA behavior (*behavioral measures*) on fitness scale or distribution, and to account for any EA parameters bias. Different behavioral measures are also proposed, distinguishing convergence time from EA radial and effective trajectory in Hamming space.

II.2.1 INTRODUCTION

In the EC field, much attention has been paid to understanding, to characterizing or to predicting EA behavior on a given fitness function. Such works often use some implicit classification of EA results. The most simple and common way to compare EA results on different problems or EA results with different parameter settings is to run numerous experiments for each setting, and to compare EA results with some user-defined criteria. Such criteria can be the average on-line and off-line fitnesses [SCED89], or the ratio of best fitnesses respectively found by EA and pure random search, after the same number of evaluations [RVSK97]. Many papers report the fraction of runs that finds the global optimum. This measure is alternatively named *success rate* (1989, Tanese's doctoral dissertation) and *robustness of evolution* [CJ91], with the difference that the latter considers the fraction of runs that find at least one –user-defined– *acceptable solution*, rather than the global optimum. The same paper also defines the *speed of evolution* as the number of generations necessary to discover an *acceptable solution*.

This trial-and-error experimental method is unfortunately time consuming. Nevertheless, the lack of efficient and robust measures of difficulty and/or adequation of settings for a given problem has made this experimental comparative method very popular since the early days of EAs.

As early as 1991, some attempt was made to formalize the properties that all comparison measures should follow [FM91] *(1) they should be normalized so that performance can be compared on different problems and with different optimizers, (2) they should reflect problem difficulty, and (3) they should indicate how far the optimizer got, not just whether or not it reached the global optimum.* However, a different point of view is encountered in [KG96] (1996) concerning SEARCH, a particular optimization algorithm: EA performance is not only related to some measure, but also to the way the measure evolves with some parameter (problem size, solution quality or success probability requirement). This is not the point of view considered here, and we stick to the requirements of [FM91] cited above.

Within the experimental comparison methods recalled above, one can distinguish two kinds of behavioral measures: those based on fitness values, and all others. Unfortunately, when different problems are compared, respective fitness distributions can be completely different. Hence, the ratio, or the amount of fitness improvement is completely unreliable. On the other hand, measures like the probability of success and the convergence time of successful runs have serious drawbacks. They do not give information about what happened during the run, and they do not take into account possible biases due to initialization or operators over-specialization to the problem at hand. For example, a given initialization can seriously bias the optimization process, by generating individuals very close in distance to the optimum of one of the compared problems (a caricatural example is given in section II.4.5 with function g'_1).

The goal of this section is to design some comparison method that allows us to overcome these drawbacks. The section organizes as follows.
Section II.2.2 defines our point of view on EA performance and motivate the proposed approach: using *Hamming fitness* as a basis to scale the behavioral measures of EA behavior when different problems, or different EA instances, are involved. Section II.2.2 specifies the definition of EA trajectory. Then, Section II.2.3 presents the comparison method for EA behaviors.
Finally, Section II.2.4 proposes different measures to characterize EA behavior distinguishing time and trajectory, and Section II.2.5 instantiates the method on some well known problems of EA literature. Special attention is paid to the comparison of GA and other ESs dynamics on the difficult Baluja binary F1 problem.

II.2.2 MOTIVATION

Before presenting the comparison method with *Hamming fitness* basis, this section discusses two different points of view on difficulty, and specifies the goal of this section.

Characterizing EA performance: Some statistics indicators of fitness landscape (as Epistasis and Fitness Distance Correlation ..) have been proposed to characterize problem difficulty, particularly for GAs. the next section (II.3) gives a detailed state of the art of these indicators, and constructs simple counterexamples that fool the indicators prediction of landscape difficulty for GAs. However, note that the reliability of an indicator is necessarily limited as it is a statistical summary ; Further, its reliability may depend on problem domains.

On the other hand, many papers [CL96], [DSG95] stress the nonsense of speaking of problem difficulty without considering all the parameters of the algorithm at hand: [CL96] presents two unimodal functions on the binary space. Depending on whether elitism is used or not, the same sampling technique (ex. steepest ascent hill climber) yields an exponential difference in convergence time. Authors explain that the potentialities of an operator are not necessarily exploited by the algorithm. Such differences are unmeasurable, of any landscape based indicator. Hence, considering the on-line behavior, relaxes this problem of the unused potentialities of the operator graph.

In that line, we proposes to consider the notion of difficulty as related to a given run of some EA instance. As a first step to refined the notion of EA difficulty, we propose a method to compare EA behaviors when applied to different fitness functions.

$\longrightarrow$ **The aim of this section** is not to define an order relation in the set of possible EA behaviors, as the notion of what is important may vary widely across problem domains. Rather, it simply proposes to extract *scalable* information that may be used later on (in a manner dependent on the user's prerequisites), to construct a total or a partial EA-performance relation that suites a particular problem domain or user's prerequisites.

On the other hand, EA *behavior* includes both temporal and spatial issues. If time is rather straightforward to define, it is not the case with spatial trajectory.

Definition of EA trajectory: The effective trajectory of an EA is the succession of

populations. To measure the length of this trajectory, one can use a distance between two populations that are considered as sets of elements of E, replacing the Hamming distance with the adequate distance function, for example the *Hausdorff's* metric [47].
But in the context of optimization, we may be interested in the trajectory of the upper tail of the successive populations, viewed as a sampling of the fitness distribution, rather than some average distance that hides what happens with the fittest individuals. Thus, EA's trajectory will be reduced to the fittest, so far, individual known to the EA at each generation. The following approach can be generalized to different definitions of the EA's trajectory, by simply replacing the Hamming distance with the adequate distance.

Moreover, different points of view can be considered as illustrated in Fig. 20:

- the distance between best individuals at each generation
- their difference in distance to optimum
- their distance to optimum

II.2.3 COMPARISON METHOD

Next section defines a number of *behavioral measures* (independent from fitness distribution) to characterize EA behavior and results on a given fitness function: In addition to convergence time, figure 20 illustrates some geometrical quantities derived from the trajectory of the algorithm.
It is not easy to compare two sets of such behavioral measures, as convergence time is very sensitive to initialization bias (initial population with close to optimal individuals), and trajectory measures are directly dependent on the choice of the genotypic distance.

Since EA behavior on *distance fitness* functions (the fitness is the distance [48] to optimum), is well understood, we propose to use it as an absolute basis to *scale* the behavioral measures, following the question: *What would this same algorithm have done –starting from the same population– if for each x the fitness function were simply the distance to optimum of f (or to the best individual found by the algorithm).*

More precisely, the comparison method consists in performing the following steps for fitness f (resp. g), then in comparing $\rho_m(f)$ and $\rho_m(g)$, where m is a behavioral measure of the algorithm as illustrated in Figure 20:

1. `Run EA with fitness` $f \longrightarrow$ `compute` $m(f)$

   Note s be the best found by the EA.

2. `Run the same`[49] `EA with fitness` $\mathcal{H}_s$[50] $\longrightarrow$ `compute` $m(\mathcal{H}_s)$

3. `Compute ratios` $\rho_m = m(f)/m(\mathcal{H}_s)$

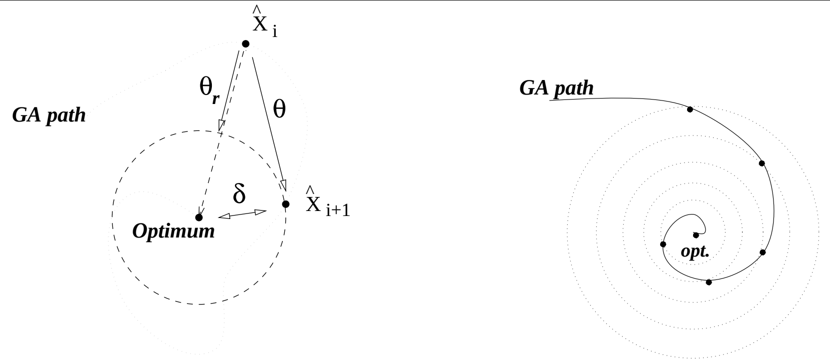
$\longrightarrow$ The *Hamming fitness* are proposed as an absolute basis to scale EA behavioral measures (convergence time and trajectory measures), and to take into account eventual parameters

[47] $\Delta(A,B) = max\{\delta(a,B) : a \in A\} + max\{\delta(b,A) : b \in B\}$, where $\delta(a,b)$ is any metric distance on E, and the distance of a point $a \in E$ to a set B is $\delta(a,B) = min\{\delta(a,b), b \in B\}$

[48] We suppose that the genotypic distance is implicitly *related* to EA operators, as for example the *Hamming* distance is related to the *1-bit-flip* mutation operator. The relevance of a distance in relation to genetic operators is addressed in the third chapter.

[49] EA with the same initial population and parameters as for f in step 1

[50] $\mathcal{H}_s(x) = l - d(x,s)$ is the Hamming fitness to s



(a) (b) Minimal θ_r

FIG 20 : *(a)* **Behavioral measures** *of EA trajectory. (b) Typical GA path with a* Hamming *distance fitness: θ is minimal, not θ_r.*

biases (such as over-fitting initialization that generates close to optimal individuals).

Using Hamming distance to compare problems or search methods is not a new idea, although scarecely used up to now. In [LR93] the author unformally proposed to characterize convergence from two different points of view:

1. how much progress is possible (related to average population fitness)
2. how much work is remaining (related to the population Hamming distance).

The author also remarks that *' using syntatactic properties of Hamming space in refining our notion of GA's trajectory through search space should lead to easier comparison with other search methods.'*

II.2.4 BEHAVIORAL MEASURES OF EAs

In the following, we present some measures that characterize on-line EA behavior (during a given run) when maximizing a fitness function f with a unique global optimum on the binary space.

Let $\hat{X}_i$ be the best-so-far individual at generation i, N_c the number of generations to reach $\hat{X}_i$, and d_n the Hamming distance normalized to string length.

The behavioral measures presented below, characterize on-line EA behavior (of a given run) to reach $\hat{X}_{N_c}$. If $\hat{X}_{N_c}$ is not the optimum, the behavioral measures characterize the dynamic of the EA to reach this local optimum. If $\hat{X}_{N_c} = X^*$, we get some useful information about optimization difficulty for the EA at hand.

1. **Convergence time:** $\mathbf{N_c} =$ Number of generations
2. **Distance to** $\hat{X}_{N_c}$: $\delta = \dfrac{\sum_{1,N_c} d_n(\hat{X}_i, \hat{X}_{N_c})}{N_c}$
3. **Trajectory distance:** $\theta = \sum_{1,N_c-1} d_n(\hat{X}_i, \hat{X}_{i+1})$

4. **Radial Trajectory distance:** $\theta_{\mathbf{r}} = \sum_{1,N_c-1} |d_n(\hat{X}_i, \hat{X}_{N_c}) - d_n(\hat{X}_{i+1}, \hat{X}_{N_c})|$

Note that θ and θ_r are not necessarily correlated:

θ_r can be minimal, with different θ values. This is the case for *Hamming fitness* functions, where experiments (of section II.2.5) show that θ typically equals 5 to 10 times θ_r. Note that θ_r is minimal and equals $d_n(\hat{X}_1, \hat{X}_{N_c})$, if the EA comes closer to $\hat{X}_{N_c}$ at each step (Fig. 20-b).

On the other hand, the unrealistic case of minimal θ necessarily implies a minimal θ_r: θ is minimum and equals $d_n\hat{X}_1, \hat{X}_{N_c})$, if each time an allele of the optimum is found, it remains fixed until the end of the run. Hence, there are no more than l improvement steps.

II.2.5 EXPERIMENTS

To illustrate the approach, we propose to focus on comparing the same function with different instances of GAs and ESs. For each function, only the most significant cases (in relation to the behavioral measures) are presented. In the last experiment, the case of the difficult Baluja F1 problem is studied for different algorithms.
Otherwise mentioned, a generational GA with a population size of 100, linear ranking selection, crossover rate 0.6 and $1/l$ mutation at rate [51] per individual 0.4, is used. Stopping criterion is 1000 generations with no improvement or 200 000 generations. Results are averaged over 10 runs.
The definition of the problems used below are recalled in Appendix A.

- **Longpath: large θ_r**

As expected, a longpath problem (see. I.3 for a detailed description) with $1/l$-bit flip mutation (Table 1) yields a particularly high Radial trajectory ($\rho_{\theta_r} = 7$): the GA -partially following the path- makes big movements getting closer then further to the optimum. These movements are responsible for the increase in average distance to optimum ($\rho_\delta = 2$), as well as for the large time ratio($\rho_{N_c} = 39$).
A longpath problem with a $2/l$-bit flip mutation shows a completely different behavior:

1. Trajectory measures (ρ_θ, and ρ_{θ_r}) are rather close to one, meaning that GA trajectory is similar to that obtained with a Hamming fitness to optimum. In fact, the search begins almost as a linear function to the first point on the path, which is two bits distant from optimum.

2. Average distance to optimum ($\delta = 0.66$) is even smaller than one. Once it found the first point on the path, it takes the GA a long time – relative to the time it would take with a Hamming function – to find the right 2-bit mutation to optimum. Hence, the GA spends a relatively large amount of time searching in close areas to the optimum.

- **F1g,F3g: influence of problem size**

The Gray coded F1 problem (defined in A) is very sensitive to the choice of problem and population size. For example, Tab. 2 shows that convergence time ratio ρ_{N_c} increases at a super-linear rate (wrt. problem size). Yet, F1g trajectory is -in average distance- as close to optimum as the Hamming run is ($\rho_\delta = 1$). Hence, the big radial trajectory ($\rho_{\theta_r} >= 3$) is not

[51] The overall mutation rate per bit is hence $0.4/l$.

	N_c	δ	θ	θ_r
longpath $1/l$**-mut**	2466(914)	0.2(0.08)	5.7(1.6)	2.7(1)
Hamming	63(9)	0.1(0.01)	2.9(0.6)	0.3(0.02)
ρ	**39**(15)	**1.9**(0.7)	**2**(0.6)	**7.5**(2.8)
longpath $2/l$**-mut**	271(142)	0.07(0.01)	3.5(0.7)	0.5(0.1)
Hamming	65(11)	0.1(0.01)	2.4	0.3(0.02)
ρ	**4.3**(2.4)	**0.6** (0.13)	**1.2** (0.4)	**1.5**(0.3)

Table 1: *GA results on the 91-bits longpath problem, using mutation at* $1/l$ *and* $2/l$*-bit-flip probability.* ρ *is the ratio of the two upper lines (longpath and Hamming behavioral measures).*

	N_c	δ	θ	θ_r
45-F3g 100-GA	**2.1** (0.3)	**1.4** (0.16)	**2.3** (1.1)	**3** (0.47)
90-F3g 100-GA	**2** (0.3)	**1.3** (0.14)	**1.6** (0.44)	**2.9** (0.29)
90-F3g 200-GA	**2** (0.35)	**1.3** (0.13)	**1.8** (0.41)	**2.9** (0.52)
45-F1g 100-GA	**16.6** (14)	**1.1** (0.29)	**1.9** (0.87)	**3.1** (0.86)
90-F1g 100-GA	**47.4** (24)	**1.1** (0.15)	**1.8** (0.3)	**4.5** (0.4)
90-F1g 200-GA	**34.8** (27)	**1.1** (0.24)	**2.2** (0.86)	**5** (1.3)

Table 2: *Influence of problem size (*-F1), with different population sizes (*-GA). Only ratios of the behavioral measures are presented. At the difference with F1g, F3g ratios scarcely vary.*

due to big deviations far away from optimum (as for longpaths), but due to small oscillating movements.

On the other hand, Gray F3 problem (defined in A) is an easy problem that has a behavior close to that of Hamming functions for different population and problem sizes (Tab. 2). The difference in convergence time $\rho_{N_c} = 2$ is mainly due to small oscillating movements as witnessed by radial trajectory $\rho_{\theta_r} = 3$.

• Ugly and Royal-Road problems: initialization bias

Two initialization procedures are compared on Ugly [Whi91] and Royal Road (RR) [MH93] problems (see. description in A). The second initialization – defined and studied in section II.5 of this chapter – is biased to enhance the density of ones in a bitstring. Although both problems have the same optimum (all ones string), the resulting initialization bias is not the same:

The second initialization procedure generates individuals closer to optimum than the standard initialization does. For instance, initial distance to optimum drops from about 0.35 (init1) to 0.15 (init2) (values given by θ_r values of the Hamming problems in Fig. 3).

As expected, when using the second initialization, convergence is a lot faster for both Royal Road and Ugly problems (Tab. 3). But, using the same initial population, *Hamming fitness* functions also converge faster, but only ρ_{N_c} for the Royal Road problem decreases. This is not surprising since the 2nd initialization enhances the emergence of the RR building blocks (blocks of 1s). On the other hand, the Ugly landscape is invariably deceptive with both initializations, hence starting closer to the optimum scarcely changes the time ratio ($\rho_{N_c} = 7$).

	N_c	δ	θ	θ_r
Ugly init1	**7.9** (4)	**0.9** (0.2)	**1.2** (0.5)	**2** (0.5)
Ugly init2	**6.8** (4.2)	**0.9**(0.3)	**2.4** (1.1)	**3.1** (0.9)
royal init1	**16.3** (8.7)	**1** (0.2)	**0.7** (0.2)	**1.3** (0.2)
royal init2	**7.2** (7.8)	**1.2** (0.2)	**1.8** (0.7)	**2.4** (1.1)

Table 3: *Influence of initialization on 90-bits Ugly and 64-bits royal problems. Only ratios are presented. The second initialization is biased to enhance density of ones in a bitstring. Note that we easily compare different problem sizes here.*

	N_c	δ	θ	θ_r
F1b* (1+200)-ES : ρ	**32**(45)	**0.8** (0.2)	**1.8** (0.5)	**1.2** (0.2)
F1b* (1,200)-ES : ρ	**49** (42)	**0.78** (0.3)	**1.7** (0.48)	**1.2** (0.3)
F1b* 200-GA : ρ	**4.3** (1.8)	**1.8** (0.6)	**3.1** (1.5)	**6.9** (3)
F1b 100-CX-GA	1356 (490)	0.3 (0.05)	11 (1.9)	3.1 (0.6)
Hamming	807 (320)	0.06 (0.018)	1.3 (0.6)	0.3 (0.04)
ρ	**1.9** (1.1)	**5** (1.7)	**12** (8.8)	**11** (2.7)

Table 4: *Study of Baluja's binary F1 problem with $l = 45$. F1b* means all runs failed to find the optimum. In which case, behavioral measures are calculated with respect to $\hat{X}_{N_c}$. Last case shows results with complementary crossover (CX): 8 GA runs out of 20 found the optimum, and results are averaged among successful runs only.*

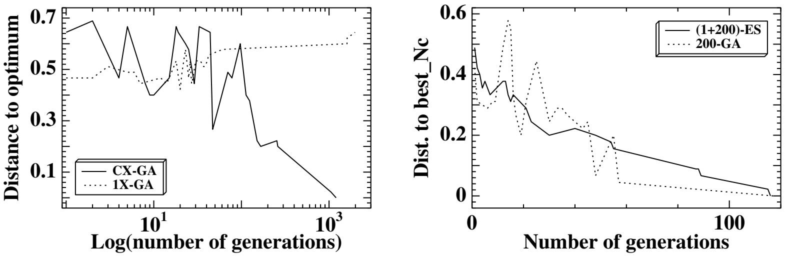


FIG 21 : *Binary F1 problem (a) GA trajectory with one point and complementary crossover plotted in distance to optimum: one point crossover GA gets stuck in a local optimum. (b) ES and GA trajectories plotted in distance to local optimum: ES trajectory is almost monotone decreasing.*

- **F1b: comparison of different algorithms**

Different algorithms are studied on Blauja's binary F1 problem with $l = 45$. Most algorithms (denoted F1b* in Tab. 4) fail to find the optimum. Only a GA using a one-point complementary crossover (CX the crossover between a string and its bit-wise complement) finds the optimum. Note that this crossover seriously weakens the *Hamming fitness* run to optimum (807 generations!!), resulting in a small ρ_{N_c} of 2 in average.
Fig. 21-a shows GA trajectory over the course of the GA. 1C-GA gets stuck in a local optimum far away from the optimum. As witnessed by the particularly high $\rho_{\theta_r} = 7$, its trajectory makes big movements both further and closer to its final best individual (Fig. 21-b). On the other hand, both evolution strategies show a trajectory similar to the trajectory obtained with a Hamming fitness (ρ_θ is close to one).
This suggests that the building blocks of the optimum are paritally contained in the bitwise complements of some high fit sub-optimal points of the search space.

II.2.6 SUMMARY AND FURTHER DIRECTIONS

Different behavioral measures have been proposed to characterize EA behavior, involving both time and spatial EA trajectory. Hamming fitness functions are proposed as a basis to scale the behavioral measures, and to account for EA parameters biases. For instance, the ratio of the behavioral measure obtained with the fitness f and that obtained with the *Hamming fitness* function, directly takes into account eventual parameter bias or overfitting.
As the comparison is independent from fitness values, optimizing f or $g(f)$ for any monotone increasing function g, give the same results.
Moreover, this method can be used to compare different problems at different problem sizes and with different algorithms. For example, from the temporal point of view, 45-bits F1g problem is as difficult as 64-bits royal problem, and (4 times) more difficult than the 2-bits mutation 91-bits longpath problem.
The case of the F1b problem is very instructive: it reveals a caricatural overfitting when the complementary crossover GA is used. On the other hand, mutation-only evolution strategies yield a low θ_r value, and globally a similar to *Hamming fitness* behavior. Yet, this situation reflects the presence of strong local optima and certainly not problem easiness. A close look at the solutions clearly shows they vary in fitness and in genotype from one run to the other.

The method presented in this section can easily be generalized to different representations, simply by replacing the *Hamming fitness* function with the appropriate *distance fitness* function (see. section III.5.4 for the choice of the distance). Alternatively the basis can be defined with a phenotypic distance. Then, a possible comparison of results obtained with genotypic and phenotypic basis, may give insights into the dynamic of the search linked to representational issues.

However, the choice of Hamming functions as a basis is rather arbitrary, and may be too limited when used alone. It only allows us to measure the difference of EA behavior from its behavior on some fixed reference basis. A better characterization of EA behavior can be obtained from the comparison with a full sequence of well known basis functions. In this case, we get a sequence of values for each considered behavioral measure, and we could identify the basis that yields the most EA-similar behavior. The NK-Landscapes [Kau89], for example, can provide such a sequence indexed by K values (note that Hamming functions correspond to a null K value).

These issues are left to further works, we shall concentrate now on *a priori* prediction of GA-behavior, and first, investigate the reliability of some fitness landscape indicators to predict GA-hardness, when restricting to particular classes of fitness functions, for instance, monotone functions, and fitness functions that invlove second order epistatic interactions (NNI). In the following, we consider a very rough notion of GA-hardness, mainly defined as non-GA-easiness. This statement is specified in section II.3.2.

II.3 Statistical Indicators of Fitness Landscapes

Overview Having precised how EA behavior and performance can be described, we now investigate some *a priori* predictors of GA performance. The goal of this section is to draw a precise picture of landscape-indicators' properties and reliability when facing different classes of easy fitness functions.

We review, investigate and extend a number of indicators of fitness landscapes, amongst which Epistasis variance (EPIV) and Fitness Distance Correlation (FDC) are the most widely known. The reference classes of these indicators (the class of functions the indicator measures a distance from) are generalized, in order to construct a new indicator (swo), which is insensitive to non-linear fitness scaling.

Further, we construct a class of parametrized site wise optimizable functions that blind the indicators values. This is due to the presence of a wide amount of irelevant information in the random sample, and appeals for alternative sampling schemes.
This study points out the need to identify further GA-easiness function classes –beyond monotone fitness functions– in order to design more efficient GA-hardness indicators.

We finally consider a real world epistasic fitness function, involving only second order interaction terms, for which all indicators give irrelevant values with respect to the behavior of GAs. This example shows that we still need more work in order to detect the difference between bad epistasic interaction for GAs from non-harmful epistasic interactions.

II.3.1 INTRODUCTION

A number of statistical indicators of fitnesses (resp. operators) landscapes –originated from empirical observations–, have been proposed to characterize classes of fitness functions with similar GA behavior. We mention *epistasis variance* [Dav91a], and *fitness distance correlation* [JF95]. Although there have been some recent experiments which suggest that the quality of these predictions is not always so reliable (e.g., in [Alt97, QVSG98] for fitness distance correlation and in [RVSK97] for epistasis variance), a thorough investigation of the properties of indicators like epistasis variance and fitness distance correlation has not been carried out yet.

Although there have been some recent experiments which suggest that the quality of these predictions is not always so reliable (e.g., in [Alt97, QVSG98, KNS98] for fitness distance correlation and in [RVSK97, RW95a] for epistasis variance), a thorough investigation of the properties of these indicators has not been carried out yet.
In fact, most of the related works were limited to the construction of separate examples and counterexamples, and as far as we know, no comparison has been made between the properties of these indicators, or of their reliability on classes of basic GA easy problems.
The goal here is to investigate the properties of correlation and epistasis indicators on different "GA-easy" binary problems. Indeed, the first step to identify GA-difficult functions, is to be able to identify GA-easy ones.

The whole analysis is based on an interpretation of the statistical indicators as a function defined on the space of fitness functions. The **reference class** of a given indicator m is then defined as the class of fitness functions the indicator measures a distance (or a difference) from. Alternatively, we can design indicators as distances from a purposely chosen reference class.

In Section II.3.2, we first define, in the context of this paper, what it means for a fitness function to be **easy for a given GA**: Four classes of GA behaviors are identified based on the GA's convergence quality and convergence speed.

The first part of Section II.3 starts by discussing epistasis variance, *epistasis correlation*, introduced in [SRV97], and fitness distance correlation. The respective reference classes of these indicators (first order and linear functions) are compared. Considering the position of the reference classes of these indicators within other GA-easy function classes (site wise optimizable and site wise decidable functions), first points out the inability of these indicators to deal with **constantness** and non-linear **scaling**, and second, motivates the need to design indicators related to more general reference classes.
The second part of Section II.3 is devoted to this generalization. It presents an indicator arising from it, called the *site-wise optimization indicator* (of which a preliminary version appeared in [BNV97]).

In Section II.3.4 we present more general function classes than those related to the above indicators, which are well-known to consist of (or at least, contain) easy functions: the unimodal (or steepest-ascent optimizable) functions and the GA-schema-easy functions (in the sense of schema competitions). Within these GA-easy function classes, we draw some instructive examples which obviously mislead all the previously discussed indicators: These examples involve a tunable amount of local deception that does not change GA behavior, but, due to **averaging**, completely blinds the indicators values.
This points out the need for more work in identifying further GA-easy function classes, which in turn may lead to designing more reliable GA-hardness indicators.

The experiments of Sect. II.3.6 compute the indicators of Sect. II.3.3 on random samples for both easy and difficult fitness functions. The goal of this Section is to show to which extent the –random sample based– values produced by the four indicators can be unrelated to the convergence behavior of commonly used GA instances (with ranking and proportional selection). In some cases, epistasis variance gives low values to almost unsolvable problems and higher values to completely trivial ones.

In the first part of Section II.3.7, we discuss how the results of the previous subsections apply to Reeves and Wright's Anova tables [RW95b, RW95a], which give much more detailed information about the epistatic structure of a fitness function than the epistasis indicators do.
The last part of Section II.3.7 comments on how to draw the samples which are used to approximate the values of the indicators for a given fitness function. We propose (for further investigation) two alternatives to random sampling: Metropolis-generated samples and strings sampled from successive generations of a GA run.

II.3.2 DEFINITIONS, CONVENTIONS

II.3.2.1 Notations

Throughout this text, the finite alphabet $\{0,1\}$ is denoted by Σ. The notation $s[i|\alpha]$ represents the string $s \in E$ after replacing site i with α. If $\alpha \in \Sigma$, $\alpha^{(n)}$ will denote the concatenation of n copies of α (i.e. $\alpha^{(n)} = \alpha\alpha\ldots\alpha$).

A schema (or hyper-plane) is a string in Σ'^l, with $\Sigma' = \Sigma \cup \{\#\}$, the alphabet augmented with a wildcard symbol. The fitness of a schema is defined as the average of the fitness values of the strings belonging to that schema. The order of a schema is the number of non-wildcard symbols in the schema [Hol75, Gol89].

Consider a fitness function f with global optimum s^*. Let m be a real valued function that takes as arguments a number of points of E and their respective fitnesses:

$$(x_i,\ f(\varphi(x_i))\,)_{i=1\ldots k},\ x_i \in E,$$

where φ is the mapping function between phenotypes and genotypes φ is the identity function here ($E = \{0,1\}^l$), but it is important to note that the indicator value depends on the choice of φ (i.e. on the choice of the representation).

In the following, we speak of the indicator value given by a fitness function, although we are aware that it depends on the representation.

Definition II.3.1
Exact and approximate indicators *If an indicator m is computed on the whole search space, it is said to be 'exact'. Otherwise, if it is computed on a sample of the space, it is said to be 'approximate'.*

Definition II.3.2
Reference class of indicator m *The reference class is defined as the class of functions characterized by an extreme value of the indicator: for instance a null value (for epistasis variance and SWO indicators), or a value of one (for fitness–distance correlation and epistasis correlation).*

Note that the epistasis variance (see II.3.3) indicators the Euclidean distance from its reference class, (i.e. the class of first order functions).

Definition II.3.3
First order functions *A fitness function is said to be a first order (or fully separable) function when it is of the form*

$$s \mapsto \sum_{i\in\{1,\ldots,l\}} g_i(s_i), \qquad \textit{with } g_i : \Sigma \to R, \quad \textit{for all } i \in \{1,\ldots,l\}.$$

We write $f \in \mathsf{1ORD}$ when f is a first order function.

Definition II.3.4
Linear functions *A fitness function is said to be* linear *when it is of the form*

$$s \mapsto C_1 - C_2\, d(s, s^*),$$

where C_1 and C_2 are constants.

Definition II.3.5
Monotone functions *A fitness function f is called* monotone, *which is denoted by $f \in \mathsf{MON}$, when it satisfies*

$$d(s^1, s^*) < d(s^2, s^*) \implies f(s^1) > f(s^2) \qquad \textit{for all } s^1, s^2 \in E.$$

Definition II.3.6
Unimodal functions *A fitness function f is unimodal ($f \in \mathsf{UNI}$), when it has no local optima, apart from the global optimum.*

II.3.2.2 Experimental conditions

Throughout this section we consider maximization problems, and use the same genetic algorithm to observe the hardness of fitness functions: Unless otherwise stated, the string size is 80, an the population size is fixed to 80 strings. We use uniform crossover, and $1/l$ mutation both with probability 0.5, and linear ranking selection. Results (convergence time) are averaged out of 21 runs with different randomly sampled initial populations. We show the standard deviation in the form of error bars.

Although many works stress the limitations of proportional selection, the experiments conducted in this section with ranking-GA are systematically compared to the results obtained with proportional selection. This gives more generality to the results of the section, and, give further evidence for the limitations of proportional selection on some chosen 'GA-easy' function classes.

II.3.2.3 Convergence quality classes

Although it is beyond the scope of this section to give a full account of the subject, we specify the statement " ... are easy for the GA" that will be used here.
In the following, we give a rough and intuitive hierarchy of the *convergence quality* of a given GA and fitness function. We are aware that whether a given fitness function belongs to one convergence quality class or not can heavily depend on the choice of parameters of the GA. In this work, we stick to the GA with ranking selection defined above to classify fitness functions in this section.

Definition II.3.7
Quality convergence classes *For a given GA and fitness function, we distinguish four convergence classes:*

1. *The GA detects the optimum of the fitness function independently of the choice of the randomly sampled initial population. The variance on the number of iterations needed to detect the optimum is comparatively* [52] *small. This behavior is called* stable convergence.

2. *Each run of the GA still detects the optimum of the fitness function, but now the variance on the number of iterations needed for detection is comparatively large (w.r.t. to* Hamming fitness *as in section II.2). The GA exhibits* unstable convergence.

3. *The GA sometimes detects the optimum of the fitness function after a reasonable number of iterations, but it takes a number of runs because the problem is sensitive to the choice of initial population.*

   This behavior can be seen as follows: there are a number of easy paths to the global optimum, but if the GA is not in one of them then it gets trapped in a local optimum from which it does not escape.

4. *The GA does not perform better than random search. It cannot detect the optimum of the fitness function in a reasonable length of time, no matter how often it is run.*

Within the first convergence quality class, *convergence speed* can be used as a refinement in the hierarchy of GA-easiness. This allows us, for example, to distinguish between monotone functions and longpaths for the GA defined in I.4 which typically show first convergence

[52]We consider that the variance is small if it is at a lower order of magnitude than convergence time.

quality but extremely low convergence speed (exponential in string size). (The latter is not to be confused with the longpath defined in [HGD94] and studied in I.3 which is a longpath for a hill climber in Hamming space).

In the same way, a partial order can be made within the second convergence quality class, simply by comparing the mean and standard deviations of convergence time for two functions. A more reliable comparison may be obtained using some statistical test (as the t-test) between two sets of convergence times.

Note, finally, that this classification is not related to landscape ruggedness or difficulty, but is based on an experimental trial-and-error method. Nevertheless, the examples we use to show that the indicators cannot even detect GA-easiness (as the RSAO functions in II.3.4.3), mainly belong to the first convergence class, and obviously show a GA convergence behavior that is 'similar' to GA convergence behavior on *Hamming fitness* functions. Hence they would fit in the GA-easy classification of section II.2.

II.3.3 MONOTONICITY

This section reviews, further investigates and extends some statistical indicators which reference classes are contained in the class of monotone fitness functions (with respect to the distance to optimum). These indicators are Epistasis variance and Epistasis correlation, Fitness Distance Correlation, and a new proposed indicator (swo) which reference class is slightly more general than monotone fitness functions. The relation between the reference classes of all these indicators are investigated, exhibiting the sensitivity of some indicators to fitness scale, and the difference between Epistasis variance and Epistasis correlation in the presence of local constantness in the fitness landscape.

II.3.3.1 Epistasis indicators and first order functions

Epistasis in biology denotes the dominance of one gene upon other genes. But in the context of optimization and evolutionary algorithms, epistasis denotes the interaction between the alleles in the expression of the fitness function. The full epistatic model can be expressed according to the order of interaction [RW95b, RW95a]:

$$\begin{aligned} f(s) = \quad & constant + \sum_{i=1}^{l} (\text{ effect of } s_i) \\ & + \sum_{i=1}^{l-1} \sum_{j=i+1}^{l} (\text{ interaction between } s_i \text{ and } s_j) \\ & + \qquad \ldots \\ & + (\text{ interaction between } s_1\ s_2 \ldots \text{ and } s_l) + \text{ random error} \end{aligned}$$

Reeves and Wright [RW95a] distinguish positive and negative interactions between genes. Positive interactions mean that the contribution of a pair of genes in the fitness expression reinforces the respective contributions of each gene (the 'interaction between s_i and s_j' term has the same sign than the 'effect of s_i' and the 'effect of s_j' terms). While epistasis indicators (detailed below) cannot distinguish between positive and negative interactions, the influence of these interactions on problem difficulty can be very different.

This issue can be related to the scaling problem of epistasis: This is illustrated in Example II.3.10, where a tunable amount of positive interactions correspond to a straightforward problem for the GA.

- **Epistasis variance:**

Davidor [Dav91a] proposed to compute the epistasis of a binary string $s \in E$ with respect to a fitness function $f : E \to R$ and a sample P as $\varepsilon_P(s) = f(s) - \xi_P(s)$, by

$$\xi_P(s) = \sum_{i=1}^{l} \left(\bar{f}_{P_i} - \bar{f}_P\right) + \bar{f}_P, \tag{112}$$

where $\bar{f}_P$ is the average sample fitness and $\bar{f}_{P_i}$ is the average fitness in the schema that have allele s_i at the i^th position:

$$\bar{f}_{P_i} = \frac{1}{|\{t \in P\,;\, t_i = s_i\}|} \sum_{\substack{t \in P \\ t_i = s_i}} f(t), \qquad \bar{f}_P = \frac{1}{|P|} \sum_{t \in P} f(t)$$

Davidor also defined the *epistasis variance* of f w.r.t. P as proportional to the Euclidean distance between f and the value ξ_P, i.e.

$$\varepsilon_P^2(f) = \frac{1}{|P|} \sum_{s \in P} \varepsilon_P^2(s) = \frac{1}{|P|} \sum_{s \in P} (f(s) - \xi_P(s))^2. \tag{113}$$

It has been proved independently in [BNV97], in [SRV96] and implicitly also in [RW95b] that ξ_P is the first order projection of f. Hence, ξ is the first order function (Def. II.3.3) which is closest to f in Euclidean distance.

Some normalization is required to achieve the equality of the epistasis variance of f and of αf, for $\alpha > 0$. Manela and Campbell [MC92] propose to divide the epistasis variance by the variance of the fitness function. Alternatively, [BNV97] first normalize the fitness function, using the Euclidean norm, before computing the unscaled epistasis variance. This approach guarantees that epistasis variance takes values in $[0, 1]$.
In this section we use the Euclidean normalization to compute the epistasis variance over the sample P:

$$\mathsf{epiv}_P(f) = \frac{\sqrt{\sum_{t \in P} (f(s) - \xi(s))^2}}{\sqrt{\sum_{t \in P} f(t)^2}}. \tag{114}$$

By $\mathsf{epiv}(f) = \mathsf{epiv}_E(f)$ we denote the epistasis variance computed over the *whole* search space. The **reference class** of epistasis variance is given by

$$\mathsf{epiv}(f) = 0 \text{ iff } f \text{ is a first order function.}$$

- **Epistasis correlation:**

Epistasis correlation, proposed in [SRV97], computes the (statistical) correlation between a fitness function f and its first order approximation ξ over a sample P:

$$\mathsf{epic}_P(f) = \frac{\sum_{t \in P} (f(t) - \bar{f}_P)(\xi(t) - \bar{\xi}_P)}{\sqrt{\sum_{t \in P} (f(t) - \bar{f}_P)^2} \sqrt{\sum_{t \in P} (\xi(t) - \bar{\xi}_P)^2}}, \tag{115}$$

with

$$\bar{f}_P = \frac{1}{|P|} \sum_{t \in P} f(t), \qquad \bar{\xi}_P = \frac{1}{|P|} \sum_{t \in P} \xi(t). \tag{116}$$

We use $\mathsf{epic}(f)$ to denote the epistasis correlation of f computed over the whole search space. Being a correlation coefficient, $\mathsf{epic}_P(f) = \mathsf{epic}_P(af+b)$ for all $a, b \in R$. Epistasis correlation values are in the range $[0,1]$. The **reference class** of epistasis correlation is a subset of first order functions, and is given by

$$\mathsf{epic}(f) = 1 \text{ iff } f(s) = \sum_{i \in \{1,\ldots,l\}} g_i(s_i) \;;\; \forall(\alpha,\beta) \in \Sigma^2, \alpha \neq \beta \Rightarrow g_i(\alpha) \neq g_i(\beta) \tag{117}$$

Unlike epistasis variance with Euclidean normalization, epistasis correlation is, as stated above, translation invariant. Moreover, epistasis correlation has the immediate advantage over epistasis variance of also having a correlation plot that is interpretable over the mere distance produced by the other.

- **Discussion**

While epistasis variance was an attempt at detecting or measuring the epistasis (amount of interaction between sites) of a fitness function, it merely detects the absence of epistasis. This was already pointed out in [Dav91b]; it is obvious that this is also the case for epistasis correlation.

Moreover, the epistasis variance indicator suffers from a more technical problem which follow from the choice of its reference class: It is easy to verify that the first order functions contain all constant functions. Consequently, epistasis variance will return the value 0 both for constant fitness functions and other first order functions (following Equation 117). This is counterintuitive because the GA deals with the two types of functions very differently. The latter are easily optimized, while constant areas in the fitness landscape decrease the search efficiency of the GA, as explained below.
Consider the special case of the following *needle in a haystack* problem. [BNV97] proved that maximal epistasis variance is obtained for a function which is zero everywhere except in two bit-complementary strings. Of course, epistasis variance approximations are very unlikely to find exactly those two strings, and will report zero epistasis variance, because of constantness.

We can demonstrate this approximation problem by computing the exact and approximated values for the Royal Road functions (see, e.g., [MFH93]). Rather unexpectedly, we also came across an approximation problem with epistasis correlation.

Example II.3.8 (Constantness problem of epistasis indicators) *The slightly generalized Royal Road functions [MFH93] are defined for $\Sigma = \{0,1\}$ and $\{1,\ldots,l\} = \{0,\ldots,2^n-1\}$, with $n > 1$. Define, for each $n > 1$, $m \leq n$ and $0 \leq i < 2^{n-m}$, the schema*

$$H_i^{n,m} = \#^{(2^m i)} 1^{(2^m)} \#^{(2^n - 2^m(i+1))}. \tag{118}$$

Then we construct, for all $m \leq m' \leq n$, the Royal Road function

$$RR(n,m,m')(s) = \sum_{k=m}^{m'} 2^k |\{i \;;\; s \in H_i^{n,k}\}| \qquad \text{for all } s \in E. \tag{119}$$

Given the fact that epistasis variance with L^2-normalization cannot evaluate a constant zero function, we translate the Royal Road functions by 0.01 before carrying out the approximations. This introduces a small but negligible difference between the exact and approximated values.

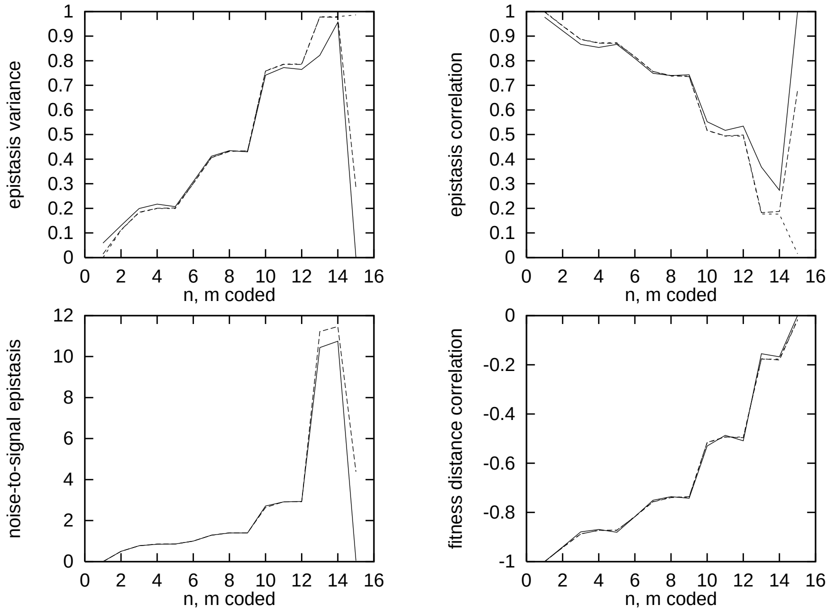


FIG 22 : Epistasis variance, epistasis correlation, noise-to-signal epistasis and fitness distance correlation of Royal Road functions with 16 sites. On the x-axis, the **15 Royal Road functions** are ordered as follows (values for m and m'): (0,0), (0,1), ... , (0,4), (1,1), (1,2), ... , (1,4), ... , (4,4). The full line indicates the sample size 2^8 approximations, the dashed line the sample size 2^{12} approximations, and the dotted line the exact values. Note that the latter has not been computed for the noise-to-signal epistasis indicator.

Close to constantness (high m value), exact and approximated values of the normalized epistasis variance of the Royal Road functions cease to agree. This is shown in Fig. 22. Note that the figure also shows a disagreement between the exact and approximated epistasis correlation values. Although the algorithm returns zero when it finds the variance of the fitness function or the variance of its linear approximation equal to zero, it does not do so when these values are close to zero (unless the fitness function is normalized, one cannot know what close to zero *means). In this case, the product of the two variances, which is of order* 10^{-31}, *closely resembles the covariance, resulting in a correlation coefficient of 1.*

The *noise-to-signal epistasis indicator* (presented below) is just an example of another way to measure the distance between a fitness function and the first order functions. As it turns out, it makes the constantness problem suffered by epistasis indicators more explicit, to the extent that this indicator is useless in practice.

Definition II.3.9
Noise-to-signal epistasis indicator *Define, for all samples P, sites $i \in \{1, \ldots, l\}$ and*

symbols $\alpha \neq \beta \in \Sigma$ *(here 0 and 1),*

$$\delta f_i(s) = |f(s[i|\alpha]) - f(s[i|\beta])|\,, \tag{120}$$

$$\sigma^2(P,i) = \frac{1}{|P|-1}\sum_{s\in P}\left(\delta f_i(s) - \overline{\delta f_i}\right)^2, \tag{121}$$

$$\nu(P,i) = \begin{cases} \dfrac{\sigma(P,i)}{\overline{\delta f_i}} & \text{if } \sigma(P,i) \neq 0, \\ 0 & \text{otherwise}, \end{cases} \tag{122}$$

$$\tag{123}$$

where $\overline{\delta f_i}$ *is the average of* $\delta f_i(s)$ *over the sample* $(s \in P)$.

The noise-to-signal epistasis indicator *of a fitness function* f *can then be defined as* $\mathsf{nts}(f) = \mathsf{nts}_E(f)$, *where*

$$\mathsf{nts}_P(f) = \frac{1}{l}\sum_{i\in\{1,\ldots,l\}} \nu(P,i).$$

It follows trivially that

$$\mathsf{nts}(f) = 0 \text{ if and only if } f \text{ is a first order function}$$

Note that this indicator is insensitive to translation and linear scaling (i.e., $\mathsf{nts}(af+b) = \mathsf{nts}(f)$, for any $a, b \in R$). The choice made in Equation 122 is responsible for the bad behavior on near constant functions.

A serious weakness of the epistasis indicators is their sensitivity to non-linear scaling, something a genetic algorithm with ranking selection is entirely insensitive to. Further, will show that a proportional selection GA is sensitive to scaling in a manner completely unrelated to the epistasis values, which can take any value of their range when presented with a monotone fitness function.

Example II.3.10 (Scaling problem of epistasis indicators) *For each positive* $n > 0$, *let*

$$v(i) = (i/l)^n \text{ for all } i \in \{0, \ldots, l\}$$

We construct a monotone fitness function NLM_n *by setting*

$$\mathrm{NLM}_n(s) = v(\#1(s)),$$

with $\#1(s)$ *the number of 1's in* s.

The higher we take this control parameter n, *the more the fitness value of optimum* $1\ldots 11$ *gets isolated from the mass of the fitness values, which results in, for example, a fitness distance correlation going to 0 for both* n *and* l *going to infinity. Figure 23 shows the results of the experiment: both epistasis variance and epistasis correlation take all values between 0 and 1. Note that proportional selection sees a stronger and stronger pressure towards strings with more ones, which quickens the convergence, while epistasis indicators are getting higher and higher (in distance from their reference class).*

We can conclude that the epistasis indicators cannot characterize GA-easy functions (i.e. nor high epistasis variance nor low epistasis correlation implies GA-hardness). Other examples have lead Mane-la and Campbell [MC92], amongst others, to come to the same conclusion.

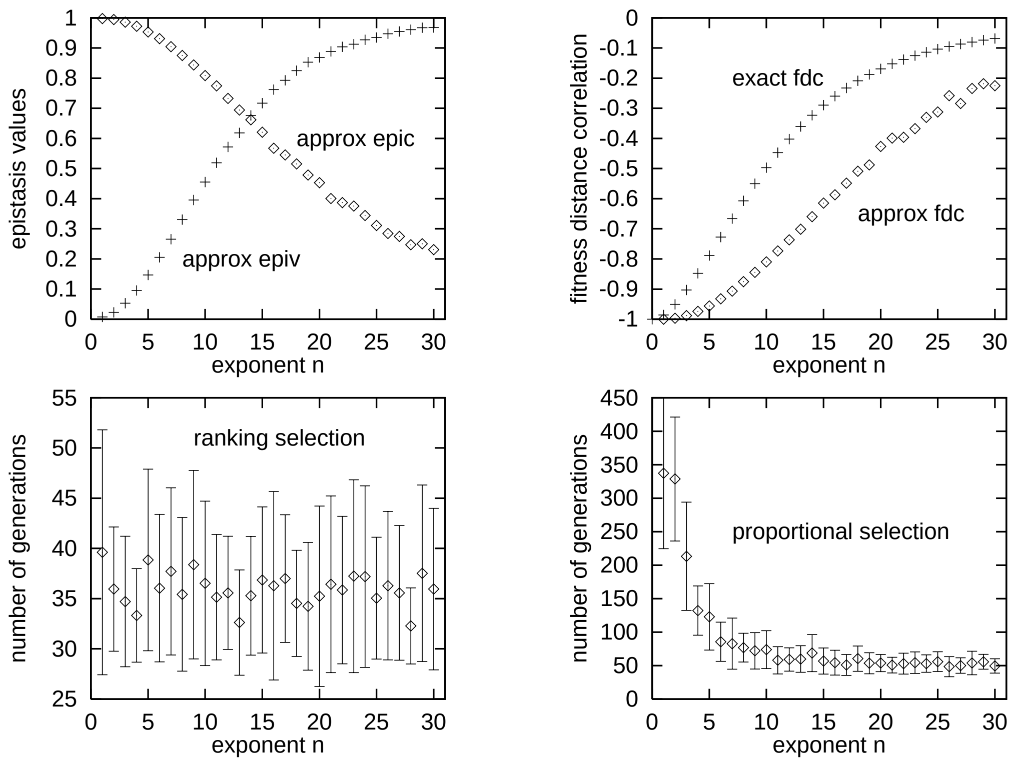


FIG 23 : *Epistasis variance, epistasis correlation, fitness distance correlation and average number of generations to detect the optimum of the* $\mathbf{NLM}_n$ **monotone functions**, *as defined in Sect. II.3.3.1, for n ranging from 1 to 30. The approximations are averaged over 21 values using samples of size* 2^{12}. *In the bottom left picture, we show the results obtained with the GA described in Sect. II.3.2 (linear ranking selection, uniform crossover, ordinary mutation). In the bottom right picture, proportional (roulette wheel) selection is used. Because of the small population size (80 individuals), the low crossover rate of 0.5 and a mutation rate which is kept at* $1/3l$ *all the time, there is not enough selection pressure for the cases* $n = 1$ *and* $n = 2$ *to converge (quality class 3 convergence). Note that the GA behaves better with the values of the indicators getting worse.*

II.3.3.2 Fitness distance correlation and linear functions

The fitness distance correlation, proposed in [JF95], computes the (statistical) correlation between a fitness function f and the distance to the global optimum s^* of f, over a sample P. The correlation coefficient is defined as:

$$\mathsf{fdc}_P(f) = \frac{\sum_{t\in P}(f(t) - \bar{f_P})(\mathrm{d}(t) - \bar{\mathrm{d}_P})}{\sqrt{\sum_{t\in P}(f(t) - \bar{f_P})^2}\sqrt{\sum_{t\in P}(\mathrm{d}(t) - \bar{\mathrm{d}_P})^2}}, \tag{124}$$

where $\bar{f_P}$ and $\bar{\mathrm{d}_P}$ are the means respectively of fitnesses and distances over the sample P. Notationally, we use $\mathsf{fdc}(f)$ to denote the fitness distance correlation value of a fitness function f.

The **reference class** of the *fitness distance correlation* is the class of linear functions (Def. II.3.4):

$$\mathsf{fdc}(f) = -1 \text{ iff } f \text{ is a linear function}$$

Hence, correlation values close to -1, characterize problems that are 'close' to *linear functions*. It is easy to see that these linear monotone functions are contained in first order functions, the reference class of the epistasis indicators.

Discussion: Fitness distance correlation is also sensitive to non-linear **scaling**. Figure 23 shows exact and approximated values for the NLM_n functions (defined in II.3.10). A first suggestion for curing this sensitivity is to perform a ranking on the fitness values before computing the correlation coefficient. The limitation of this solution is that it is only useful when there are not more different fitness values than there are distances to the global optimum. The optimal correlation of -1 would then be obtained for all functions of the form $s \mapsto C_1 - C_2\,\mathrm{d}(s,s^*)^n$, for example.

At the opposite of epistasis indicators, fitness distance correlation detects the presence of constantness in the landscapes, and reports a 0 correlation for constant functions.

On the other hand, defined as an **average**, fitness distance correlation can be blinded by the presence of a large proportion of irrelevant deception. The example II.3.21 will illustrate this problem.

The **fitness–distance relation** of the whole search space provides a lot more information than the average correlation value. For instance, the sample lay out and diversity in the correlation plot can be useful to explain genetic performance [MF98] or to choose an initialization procedure (Section II.5).

But, beyond diversity considerations, interpreting the correlation plot can be extremely misleading, mainly because two points which are close together in the plot do not have to be close in Hamming space. This phenomenon is studied in greater detail in Section II.4, and caricatural examples are given in Section II.4.5, where GA behavior convergence quality (Def. II.3.7) ranges from 1 to 4 on fitness functions with a fixed (longpath–shaped) Fitness Distance relation.

An immediate conclusion is that the fitness distance relation (and correlation) cannot be reliably related to GA convergence behavior. The examples of Sect. II.3.6 will give more evidence to this fact, when correlation is computed on random samples.

Remark: Note that Figure 1 also shows that the exact and approximated fitness distance correlations disagree for exponents $n > 1$ and sample sizes of the order of 2^{12}. Moreover, the values of epistasis correlation and fitness distance correlation for NLM_n are similar. The same phenomenon occurs in example II.3.8, where values for the Royal Road functions are computed.

II.3.3.3 The site-wise optimization indicator: one step beyond monotonicity

In this section, we construct a class which is a generalization of both the first order functions and the linear monotone functions, and we propose an associated indicator which is insensitive to scaling by strictly monotone increasing fitness functions.

The proof that a first order function can be characterized by exactly $l(|\Sigma|-1)+1$ parameters, as a direct consequence of the following proposition.

Proposition II.3.11
Let f be a fitness function. Then

$$f \in \mathsf{1ORD} \iff \textit{if } \beta \in \Sigma, \textit{ then } f(s[i|\beta]) - f(s[i|\alpha]) \textit{ is independent of } s \in E \textit{ for all } i \in \{1,\dots,l\} \textit{ and } \alpha \in \Sigma \setminus \{\beta\}.$$

Proof. Let f be in $\mathsf{1ORD}$, i.e., there exist $g_i : \Sigma \to R$, with $i \in \{1,\dots,l\}$, such that $f(s) = \sum_{i\in\{1,\dots,l\}} g_i(s_i)$ for all $s \in E$. Let β be in Σ. Then for any $s \in E$, any $i \in \{1,\dots,l\}$ and $\alpha \in \Sigma \setminus \{\beta\}$,

$$f(s[i|\beta]) - f(s[i|\alpha]) = g_i(\beta) - g_i(\alpha), \tag{125}$$

which is independent of s. To prove the other implication, we will first construct the g_i maps. Choose $\beta \in \Sigma$, $j \in \{1,\dots,l\}$ and $s \in E$, let $b = \beta\beta\dots\beta$, and set

$$\delta_i^\alpha = f(s[i|\alpha]) - f(s[i|\beta]) \tag{126}$$

for all $i \in \{1,\dots,l\}$ and $\alpha \in \Sigma \setminus \{\beta\}$. Then define

$$g_j(\beta) = f(b) \tag{127}$$
$$g_j(\alpha) = f(b) + \delta_j^\alpha \qquad \text{for all } \alpha \in \Sigma \setminus \{\beta\}, \tag{128}$$
$$g_i(\beta) = 0 \qquad \text{for all } i \in \{1,\dots,l\} \setminus \{j\}, \tag{129}$$
$$g_i(\alpha) = \delta_i^\alpha \qquad \text{for all } \alpha \in \Sigma \setminus \{\beta\} \text{ and } i \in \{1,\dots,l\} \setminus \{j\}. \tag{130}$$

We prove that $f(s) = \sum_{i\in\{1,\dots,l\}} g_i(s_i)$ by induction on the Hamming distance between s and b. The base of the induction, Hamming distance zero, follows immediately from the construction. Let us now assume that $f(t) = \sum_{i\in\{1,\dots,l\}} g_i(t_i)$ for all t at Hamming distance $k < l$ from b, and take $s \in E$ such that $d(b,s) = k+1$. Then we pick a $k \in \{1,\dots,l\}$ such that $s_i \neq \beta$, and we have

$$\sum_{i\in\{1,\dots,l\}} g_i(s_i) = f(b) + \sum_{i\in\{1,\dots,l\}\cdot s_i\neq\beta} \delta_i^{s_i} \tag{131}$$
$$= \Big(f(b) + \sum_{i\in\{1,\dots,l\}\cdot i\neq k\wedge s_i\neq\beta} \delta_i^{s_i}\Big) + \delta_k^{s_k} \tag{132}$$
$$= f(s[k|\beta]) + \delta_k^{s_k} \tag{133}$$
$$= f(s),$$

which completes the proof. □

To generalize the reference class of epistasis variance, we propose to relax the requirement of a constant difference of fitness values $(f(s[i|\beta]) - f(s[i|\alpha]))$ in proposition II.3.11 above, with the requirement that the difference always has the same sign.

Taking this into account, we propose a reference class denoted SWD, the class of *site-wise decidable* functions:

Definition II.3.12
Site-wise decidable functions *f is* site-wise decidable *if for each site there exists a symbol that we can substitute in every string to possibly improve the fitness value of the string:*

$$\exists s^* \in E \cdot \forall t \in E, \forall i \in \{1, \ldots, l\} \cdot f(t[i|s^*_i]) \geq f(t). \tag{134}$$

It is easy to verify that all first order functions are site-wise decidable. One obvious generalization of fitness distance correlation's reference class (linear functions), is the class of the monotone functions. Hence, the condition "if one comes closer to the optimum by n units, the fitness has to increase proportionally to n", is relaxed with the one-directional condition "if one comes closer to the optimum, the fitness has to increase".
Before we investigate the relationship between site-wise decidable and monotone functions, let us first consider the *site-wise optimization* algorithm. It has been introduced as the bit-setting optimization algorithm in [Wil91].

Algorithm-SWO:

Input: a fitness function $f : E \to R$ and a string $s \in E$.

Algorithm: Construct a string $m \in E$ by performing the following steps for each site $i \in \{1, \ldots, l\}$:

-- Construct the set

$$A_i = \{\, \alpha \in \Sigma \,;\, f(s[i|\alpha]) \geq f(s[i|\beta]) \text{ for all } \beta \in \Sigma \,\}$$

-- If $s_i \in A_i$, then assign $m_i := s_i$. Otherwise, assign m_i to an arbitrary element of A_i.

Output: The string m, denoted SWO(f, s).

Definition II.3.13
Site-wise optimizable function *We call a function f* site-wise optimizable *(denoted $f \in$ SWO) if the output of Algorithm-SWO is optimal regardless of the input string.*

It follows that f is site-wise optimizable and has a unique global optimum if and only if there is a $m \in E$ such that the output of the site-wise optimization algorithm is m regardless of the input string s.

The close relationship between site-wise optimizable and site-wise decidable functions is best understood by introducing yet another property for fitness functions.

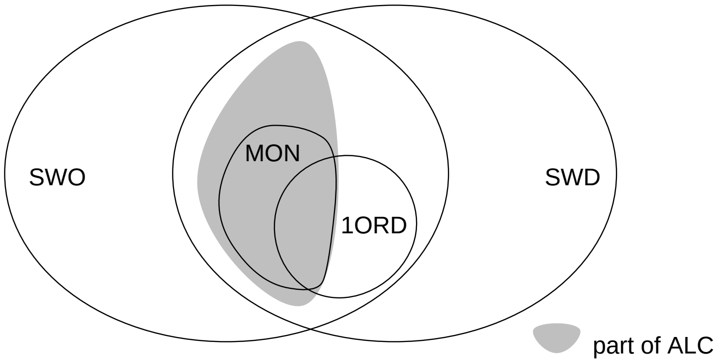


FIG 24 : *The position of the first order and monotone functions within the classes of site-wise optimizable and site-wise decidable functions. PropositionsII.3.25, II.3.26, II.3.27, II.3.28 and examples II.3.22,II.3.23, II.3.29 of subsection II.3.5 the inclusions of this diagram. Note that the absence of local constantness also separates the site-wise optimizable functions with one global optimum from those with more (i.e., the shaded area also represents* SWO ∩ UGO*).*

Definition II.3.14
Absence of local constantness *We call the following property the* absence of local constantness *(*ALC*)*

$$\forall s \in E\, \forall i \in \{1, \ldots, l\}\, \exists \alpha \in \Sigma\, \forall \beta \in \Sigma \setminus \{\alpha\}\, f(s[i|\alpha]) > f(s[i|\beta]). \tag{135}$$

Figure 24 shows the relations between most of the function classes introduced so far. The proofs of the inclusions can be found in subsection II.3.5.

We propose the site-wise optimizable fitness functions with a unique global optimum as a generalization of the reference classes of epistasis and fitness distance correlation indicators. They do not possess local constantness, which caused the approximation problems of epistasis indicators. Moreover, they are only slightly more general than the monotone functions.
We will now construct a new indicator that captures the difference between a fitness function and the class of site-wise optimizable functions with unique global optimum, using the fact that these functions can be optimized by the site-wise optimization algorithm. (This indicator was first described in [BNV97], where it was denoted by τ, the bit decidability indicator).
Let us define the *width* of a population (or sample) P to be the average Hamming distance between its members, i.e.,

$$w(P) = \frac{1}{|P|^2} \sum_{p,q \in P} \mathrm{d}(p, q) \tag{136}$$

The following proposition gives the width of the search space E.

Proposition II.3.15
The width $w(E)$ of the search space equals $l/|\Sigma|$.

Proof. Let $\alpha \in \Sigma$ and $i \in \{1, \ldots, l\}$. There are exactly $|\Sigma|^{l-1}$ strings in E with a value of α at site i. Exploiting this symmetry, we restrict our calculation to the average Hamming distance between $|\Sigma|^{l-1}$ α's and $(|\Sigma|-1)|\Sigma|^{l-1}$ other symbols, and multiply the result by $l|\Sigma|$. This yields

$$w(E) = l|\Sigma| \frac{|\Sigma|^{l-1}(|\Sigma|-1)|\Sigma|^{l-1}}{|\Sigma|^{2l}} = \frac{l}{|\Sigma|}$$

□

If we denote by $\mathrm{SWO}(f, P)$ the population formed by applying the site-wise optimization algorithm on the members of P iindividually, then $w(P) = 0$ for all samples P if and only if f has a unique global optimum and is site-wise optimizable. A natural way to define the *site-wise optimization indicator* is then:

Definition II.3.16
Site wise optimization indicator *The* site-wise optimization indicator *is given by*

$$\mathsf{swo}_P(f) = \frac{w(SWO(f, P))}{w(P)} \tag{137}$$

The range of the (exact, not approximated) site-wise optimization indicator of function f $\mathsf{swo}(f) = \mathsf{swo}_E(f)$ is the closed interval $[0, 1]$. A value of 1 is obtained, amongst others, by constant fitness functions. This has a trivial interpretation: there is no information for the site-wise optimization algorithm to map a string s to anything else other than s itself (i.e., the values of the sites cannot be decided at all).

Proposition II.3.17
For any fitness function f, we have $0 \leq \mathsf{swo}(f) \leq 1$.

Proof. We have to show that no population P of $|\Sigma|^l$ elements can have a width strictly greater than $l/|\Sigma|$. Again we exploit the symmetry in the definition of the width of the population, by calculating the highest value that one single site can contribute to the total width. Let x_α be the number of α's at site i, formally

$$x_\alpha = |\{s \in P \ ; \ s_i = \alpha\}| \qquad \text{for all } \alpha \in \Sigma \text{ and } i \in \{1, \ldots, l\}. \tag{138}$$

Obviously $\sum_{\alpha\in\Sigma} x_\alpha = |\Sigma|^l$, and we have to optimize

$$\sum_{\alpha\in\Sigma} x_\alpha \Big(\sum_{\beta\in\Sigma\setminus\{\alpha\}} x_\beta \Big) = 2 \sum_{\alpha<\beta\in\Sigma} x_\alpha x_\beta. \tag{139}$$

Using Lagrange multipliers we obtain the system of equations

$$2 \sum_{\beta\in\Sigma\setminus\{\alpha\}} x_\beta + \lambda = 0 \qquad \text{for all } \alpha \in \Sigma, \tag{140}$$

$$\sum_{\alpha\in\Sigma} x_\alpha = |\Sigma|^l, \tag{141}$$

from which $x_\alpha = |\Sigma|^{l-1}$ follows for all $\alpha \in \Sigma$. This concludes our proof. □

As a second improvement on the epistasis indicators and fitness distance correlation, we have that the site-wise optimization indicator is invariant to non-linear fitness function scaling, i.e., $\mathsf{swo}(g(f)) = \mathsf{swo}(f)$ for all strictly monotone increasing functions g. As a consequence, $\mathsf{swo}(\mathrm{NLM}_n) = 0$ for all n.
Exact site-wise optimization values for the Royal Road functions are easily obtained, see [BNV97]. They only depend on the block length (and *not* on the number of blocks, which is in proportion to the string length). The approximation (vs. exact indicators) quality is very good.

II.3.4 GA-EASINESS BEYOND MONOTONICITY

In the previous section we demonstrated that fitness distance correlation and the epistasis indicators can take any value when presented with monotone fitness functions that all share the same convergence behavior (convergence quality class 1 according to the classification of Sect. II.3.2). In order to find other such function classes, and to test the site-wise optimization indicator, we will have to look beyond the monotone fitness functions.
Another reason for looking beyond monotonicity is that even the class of monotone fitness functions is too small to play the role of a reference class: there exist many other fitness functions for which the GA shows quality class 1 behavior.

In Sect. II.3.4.1 and II.3.4.2, we present two larger fitness function classes related to GA-easiness: the unimodal (or steepest-ascent optimizable) functions and the schema-GA-easy (or non-deceptive) functions. Section II.3.4.3 presents a subclass of the unimodal functions, called the *restricted steepest-ascent optimizable* functions. Examples will be drawn from this subclass that fool the predictive quality of all the indicators of the previous section.

Figure 25 sketches the relationship between the site-wise optimizable functions and the three function classes introduced in the rest of this section.
From this point on, we will note UGO as the class of fitness functions that have a unique global optimum.

II.3.4.1 Steepest-ascent optimization and unimodal functions

The steepest-ascent optimization algorithm (see, e.g., [Wil91]) is the standard implementation of a hill climber on the space $\Sigma^{\{1,\ldots,l\}}$. It is defined analogously to the site-wise optimization algorithm.

Algorithm-SAO:

<u>Input:</u> a fitness function $f : E \to R$ and a string $s \in E$.

<u>Algorithm:</u> Modify the string s by repeatedly performing the following steps:

-- Record, for all sites i and symbols α, the triples

$$(i, \alpha, f(s[i|\alpha]) - f(s))$$

-- Assign $s_i = \alpha$ where i and α are chosen such that the third component of the triple (i, α, x) reaches the highest possible value (this is the steepest ascent).

-- Stop when no triple with strictly positive third component has been found.

Output: The string s, denoted $\mathtt{SAO}(f, s)$.

Note that this algorithm always stops after a finite number of changes in s. We call a fitness function $f : E \to R$ *steepest-ascent optimizable* if the application of this algorithm leads to the optimum of f regardless of the input string.
It is easy to verify that a fitness function f is steepest-ascent optimizable if and only if it is unimodal. Also, that all site-wise optimizable functions are steepest-ascent optimizable [Wil91]. The same does not apply to site-wise decidable functions, which is shown in the following example.

Example II.3.18 ($\mathsf{SWD} \not\subseteq \mathsf{UNI}$) *Let* $\Sigma = \{0,1\}$ *and* $\{1,\ldots,l\} = \{0,1,2\}$. *Assign arbitrary fitness values to* f *such that*

$$f(000) > f(001) = f(010) = f(100)$$
$$f(100) > f(011) = f(101) = f(110) = f(111)$$

The function f *is site-wise decidable (filling in 0 does not decrease the fitness value) and 111 is a local optimum. Note that Ex. II.3.23 shows that* $(\mathsf{SWD} \cap \mathsf{UNI}) \setminus \mathsf{SWO}$ *is not empty.*

To complement Wilson's example of a steepest-ascent but not site-wise optimizable function, we present examples of fitness functions in $(\mathsf{UNI} \setminus \mathsf{SWO}) \cap \mathsf{ALC}$ and $(\mathsf{UNI} \setminus \mathsf{SWO}) \setminus \mathsf{ALC}$:

Example II.3.19 ($(\mathsf{RSAO} \setminus \mathsf{SWO}) \cap \mathsf{ALC} \neq \emptyset$) *As in [Wil91], but slightly modified for* ALC *membership. Let* $\Sigma = \{0,1\}$ *and* $\{1,\ldots,l\} = \{0,1,2\}$. *Assign arbitrary fitness values to* f *such that*

$$f(111) > f(110) > f(101) > f(011) > f(100) >$$
$$f(010) > f(000) > f(001).$$

To avoid ALC *membership, take Wilson's example. We can tune the values of this function to obtain or avoid* schema-GA-easy *membership.*

It is worth noting that there also exist easy functions at the other end of the modality spectrum. Horn and Goldberg [HG95] show that there exists easy yet maximally multi-modal fitness functions.

II.3.4.2 Schema-GA-easy functions

The term *GA-easy*, which we refer to as *schema-GA-easy*, originated in the study of deception and schema competitions, started mainly in [Gol87]. In this context, [Whi91], [DW91], [LV91] and [Gre93] amongst others, are relevant. Briefly, a fitness function f is called schema-GA-easy whenever the winners of the competition of any order k less than l contain the global optimum of f.

It is straightforward to prove that all monotone fitness functions are schema-GA-easy. An investigation of the statements $\mathsf{SWD} \subset$ schema-GA-easy and $\mathsf{SWO} \subset$ schema-GA-easy requires a more precise definition of schema-GA-easiness: we could, for example, state that no competitions can remain undecided, or that the winners can only lead to one unique optimum instead of many different ones. In this way, all first order functions could be seen as schema-GA-easy, or only the subset $\mathsf{1ORD} \cap \mathsf{ALC}$. For the remainder of this text, we say that a function is schema-GA-easy when all competitions (of any order) are decided in favor of the same optimum. In subsection II.3.5 we prove that $\mathsf{SWO} \cap \mathsf{UGO} \subset$ schema-GA-easy.

It is shown in [Wil91] that schema-GA-easy fitness functions are not necessarily steepest-ascent optimizable. A function which is not schema-GA-easy is called *deceptive* (of some sort and order). A simple example of a deceptive function which is steepest-ascent optimizable is the following.

Example II.3.20 ($\mathsf{UNI} \not\subseteq$ schema-GA-easy) *Let* $\Sigma = \{0,1\}$ *and* $\{1,\ldots,l\} = \{0,1\}$. *Assign the following fitness values:* $f(00) = 5$, $f(01) = 1$, $f(10) = 4$ *and* $f(11) = 3$. *The function is deceptive and steepest-ascent optimizable.*

Vose and Wright [VW95] present the extreme case of a (fully) non-deceptive function which is difficult for the GA, showing that schema-GA-easiness does not imply GA-easiness. Depending on the choice of mutation rate, this function belongs to quality classes 2 or 3.

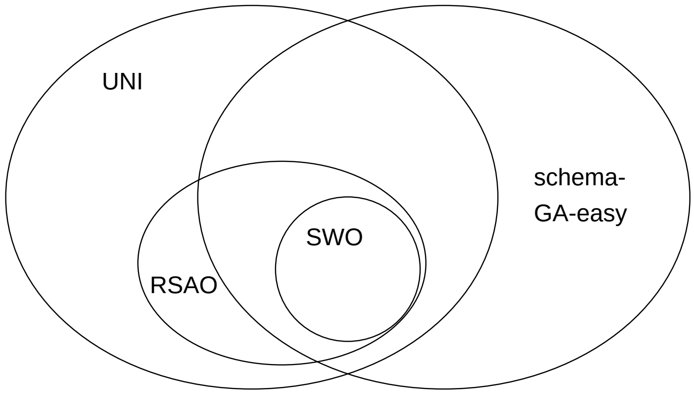


FIG 25 : Pictorial description of the relationship between four classes related to GA-easiness in the case of functions with unique global optimum. In the intersection of the unimodal and schema-GA-easy functions we find the class of site-wise optimizable functions, itself containing the monotone functions. The restricted steepest ascent optimizable functions are trivially contained in the unimodal functions. Propositions II.3.30, II.3.31 and examples II.3.19, II.3.20 in subsection II.3.5 prove the inclusions of this diagram. The longpath problem presented in [HGD94] is an example of a unimodal function which is not restricted steepest ascent optimizable nor schema-GA-easy.

II.3.4.3 Restricted steepest-ascent optimizable functions

Algorithm-RSAO

```
The restricted steepest-ascent optimization algorithm is a
   restricted version of the steepest-ascent optimization algorithm
   in the sense that, once the value at a site has been modified,
   it cannot be modified again.
```

It follows that this algorithm does not perform more than l steps. It can be considered as a compromise between site-wise and steepest-ascent optimization.
We call a fitness function f restricted steepest-ascent optimizable, and we write $f \in \mathsf{RSAO}$, if and only if this algorithm reaches optimum independently of the input string s. Obviously, $\mathsf{SWO} \subset \mathsf{RSAO} \subset \mathsf{UNI}$.
All the restricted steepest-ascent optimizable functions that we have come across so far, and certainly those forthcoming in Sect. II.3.21 and II.3.6.3, showed high convergence speed and optimal convergence quality.
In the following we present a subclass of restricted steepest-ascent deceptive functions, which obviously mislead all indicators of Sect. II.3.3.

Example II.3.21 Deceptive mixtures of OneMax and zeromax: *By mixing a OneMax and a number of zeromax problems we obtain a class of deceptive yet restricted steepest-ascent optimizable functions.*
Let $\Sigma = \{0, 1\}$*,* $\{1, \ldots, l\} = \{0, \ldots, l-1\}$ *and let* n*, with* $0 < n < l$*, be a control parameter. We construct a fitness function* f_n *by first putting an* $l-1$*-site zeromax problem on the schema* $0\#^{l-1}$*, followed by an* $l-2$*-site zeromax problem translated by* $2l$ *on the schema* $10\#^{l-2}$*. We continue adding translated zeromax problems until an* $l-n$*-site zeromax problem, translated by* $2(n-1)l$*, is put on the schema* $1^{n-1}0\#^{l-n}$*. Finally, we put an* $l-n$*-site OneMax problem on the schema* $1^n\#^{l-n}$ *and translate it by* $2nl$*.*

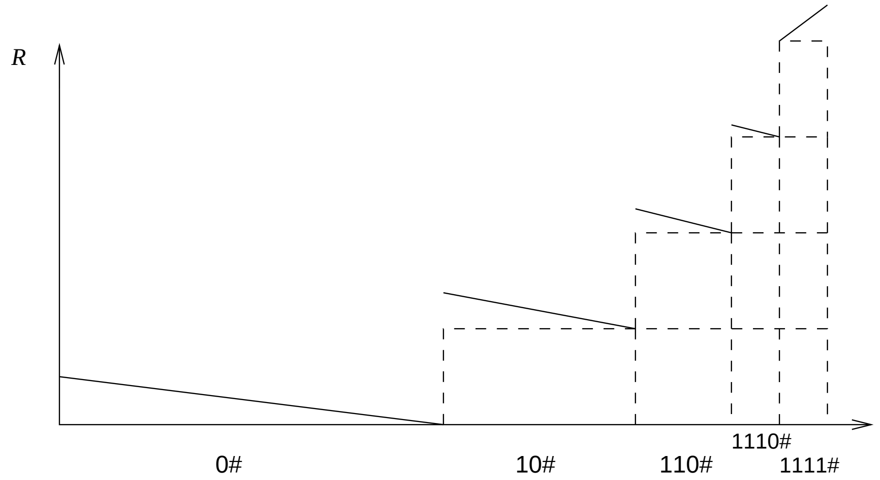


FIG 26 : Description of the deceptive mixture of a OneMax and 4 zeromax problems in the unitation space. The mixture of a OneMax and four nearest neighbor problems (as in Sect. II.3.6.3) is obtained by substituting the zeromax functions and adjusting the height of the translations according to the optimum of the nearest neighbor functions.

As an example, the function f_4 is sketched in Fig. 26. In short,

$$\begin{aligned}
f_4(0s_1 \ldots s_{l-1}) &= \textit{ number of zeros in } s_1 \ldots s_{l-1} \\
f_4(10s_2 \ldots s_{l-1}) &= \textit{ number of zeros in } s_2 \ldots s_{l-1} + 2l \\
f_4(110s_3 \ldots s_{l-1}) &= \textit{ number of zeros in } s_3 \ldots s_{l-1} + 4l \\
f_4(1110s_4 \ldots s_{l-1}) &= \textit{ number of zeros in } s_4 \ldots s_{l-1} + 6l \\
f_4(1111s_4 \ldots s_{l-1}) &= \textit{ number of ones in } s_4 \ldots s_{l-1} + 8l
\end{aligned}$$

The experiments of the next section demonstrate that GA behavior on these functions belongs to the first convergence quality class (definition II.3.7).
However, the fitness–distance relation with such functions is far from linear, because of the deceptive tendencies present in sub-parts of the search space. For example, half the space $0\#^{l-1}$ have a correlation coefficient of -1. It's also obvious that there is a significant amount of first order epistasis. The fitness can vary significantly $(+/-2l)$ if the adequate locus is inverted, whereas it scarcely varies if the wrong locus is inverted.

II.3.5 PROOFS

Some of the proofs of the relations between function classes (deferred to this section to improve readability) are presented below:

Example II.3.22 (SWO $\not\subseteq$ SWD **)** *Let $\Sigma = \{0,1\}$ and choose values for f such that*

$$f(000) = f(100) = f(110) = f(111), \tag{142}$$
$$f(011) < f(001) < f(101) < f(010) < f(000). \tag{143}$$

Clearly f is site-wise optimizable:

$$SWO(f,010) = SWO(f,001) = SWO(f,011) = 100 \tag{144}$$

and $SWO(f,101) = 110$. But it is not site-wise decidable, for $f(010) < f(000)$ and $f(101) < f(111)$.

Example II.3.23 ($\mathsf{SWD} \not\subseteq \mathsf{SWO}$) *Let $\Sigma = \{0,1\}$ and choose values for f such that $f(00) > f(01) > f(10) = f(11)$. The conditions for site-wise decidability are satisfied, but $SWO(f, 11) = 01$, which is suboptimal.*

Proposition II.3.24
$\mathsf{SWD} \cap \mathsf{ALC} \subset \mathsf{SWO}$.

Proof. Assume that there exist $s^1, s^2 \in E$ such that $m^1 = \mathrm{SWO}(f, s^1)$ and $m^2 = \mathrm{SWO}(f, s^2)$ are different. Let i be one of the sites in which they differ. According to the the algorithm, we have (the ALC property of Def. II.3.14 is responsible for the strict inequalities)

$$f(s^1[i|m_i^1]) > f(s^1[i|m_i^2]) \qquad \text{and} \tag{145}$$
$$f(s^2[i|m_i^2]) > f(s^2[i|m_i^1]), \tag{146}$$

which contradicts the fact that f is site-wise decidable. □

Proposition II.3.25
$\mathsf{SWO} \cap \mathsf{ALC} \subset \mathsf{UGO}$.

Proof. By induction on l, the number of sites. Assume first that $l = 2$, e.g., $\{1, \ldots, l\} = \{0, 1\}$. Suppose f is multi-modal, with optima at s^1 and s^2. The ALC membership forces the optima to be at Hamming distance 2 from each other (i.e., $s_0^1 \neq s_0^2$ and $s_1^1 \neq s_1^2$). We also have that $f(s_0^1 s_1^2)$ is strictly less than $f(s^1)$. Hence $\mathrm{SWO}(f, s_0^2 s_1^1) = s_0^1 s_1^2$ cannot be optimal, which is in contradiction with $f \in \mathsf{SWO}$. This proves the base of the induction.
The induction step is proved similarly. We are given that all fitness functions $g \in \mathsf{SWO} \cap \mathsf{ALC}$ defined on l sites are unimodal. Let $f \in \mathsf{SWO} \cap \mathsf{ALC}$ be defined on $l+1$ sites, and assume it is multi-modal. There can only be two optima, because all optima have to be at Hamming distance $l+1$ from each other. (This follows from the induction hypothesis.) Let s^1 and s^2 be these optima, with $s_i^1 \neq s_i^2$ for all $i \in \{1, \ldots, l\}$. Fix $i \in \{1, \ldots, l\}$. Again we have $f(s^1[i|s_i^2]) < f(s^1)$, as $f \in \mathsf{ALC}$, and we have a contradiction because $\mathrm{SWO}(f, s^2[i|s_i^1]) = s^1[i|s_i^2]$ which is suboptimal. (The latter equality follows from the induction hypothesis: if the value at site i equals s_i^1, then the algorithm maps the value of site $j \in \{1, \ldots, l\} \setminus \{i\}$ to s_j^1.) □

Proposition II.3.26
$\mathsf{SWO} \cap \mathsf{ALC}^C \subset \mathsf{UGO}^C$.

Proof. Let f be in SWO and not in ALC. It follows from the latter fact that there exist $s \in E$, $i \in \{1, \ldots, l\}$ and $\alpha \neq \beta \in \Sigma$ such that

$$f(s[i|\gamma]) \leq f(s[i|\alpha]) = f(s[i|\beta]) \tag{147}$$

for all $\gamma \in \Sigma$. As a consequence, $\mathrm{SWO}(f, s[i|\alpha])$ will have value α at site i, and $\mathrm{SWO}(f, s[i|\beta])$ value β at that site. The outcomes are optimal and different. □
Example II.3.23 shows that there exist unimodal $f \in \mathsf{ALC}^C$.

Proposition II.3.27
$\mathsf{MON} \subset \mathsf{SWD}$.

Proof. Let s^* be the unique global optimum of the monotone fitness function f. Let $t \in E$ and $i \in \{1,\ldots,l\}$. We have to show that $f(t[i|s_i^*]) \geq f(t)$. When $s_i^* = t_i$, this is the case. $s_i^* \neq t_i$, then $\mathrm{d}(t[i|s_i^*], s^*) < \mathrm{d}(t, s^*)$, from which $f(t[i|s_i^*]) \geq f(t)$ follows. □

Proposition II.3.28
$\mathsf{MON} \subset \mathsf{SWO}$.

Proof. Let s^* be the unique global optimum of the monotone fitness function f. Let $t \in E$ and $i \in \{1,\ldots,l\}$. We know that $\mathrm{d}(s[i|s_i^*], s^*) < \mathrm{d}(s[i|\beta], s^*)$ for all $\beta \neq s_i^*$, hence $f(s[i|s_i^*]) > f(s[i|\beta])$ for all $\beta \neq s_i^*$, from which follows that the site-wise optimization algorithm will pick s_i^* as the best symbol. □

Example II.3.29 ($\mathsf{SWO} \cap \mathsf{UGO} \not\subseteq \mathsf{MON}$) *Let $\Sigma = \{0,1\}$ and $\{1,\ldots,l\} = \{0,1,2\}$. Take the following values for f:*

$$\begin{array}{ll} f(000) = 8 & f(100) = 6 \\ f(001) = 4 & f(101) = 2 \\ f(010) = 7 & f(110) = 5 \\ f(011) = 3 & f(111) = 1 \end{array}$$

One verifies that $f \in \mathsf{SWO} \cap \mathsf{UGO}$, but f is not monotone since $f(001) < f(110)$.

Proposition II.3.30
$\mathsf{SWO} \subset \mathsf{UNI}$

Proof. Let $f \in \mathsf{SWO}$. This means that, under f, the site-wise optimization algorithm transforms each string $s \in E$ to one of the global optima. Assume that f contains a local optimum $m \in E$. Then necessarily m is transformed onto itself by the algorithm, a contradiction. □

Proposition II.3.31
$\mathsf{SWO} \cap \mathsf{UGO} \subset$ schema-GA-easy

Proof. Let $f \in \mathsf{SWO} \cap \mathsf{UGO}$, and let s^* be the unique optimum of f. Consider a schema competition of order $n < l$, with the wildcard symbols at positions $I = \{i_1,\ldots,i_n\} \subset \{1,\ldots,l\}$. We have to show that the schema $H(s^*)$, with wildcard symbols at positions in I and symbols of s^* at positions in $\{1,\ldots,l\} \setminus I$ (i.e., $H(s^*)_j = s_j^*$ if $j \notin I$), wins the competition, in favor of the schemas $H(s)$ which have the symbols of $s \in E$ at the non-wildcard positions. Let $t \in E$. Then, because of site-wise decidability and absence of local constantness, the string formed by putting the symbols of t at the wildcard positions of $H(s)$ has a strictly smaller fitness value than the string formed by putting the symbols of t at the wildcard positions of $H(s^*)$. This implies that the schema $H(s^*)$ wins the competition (average of pointwise strictly smaller sequence of numbers). □

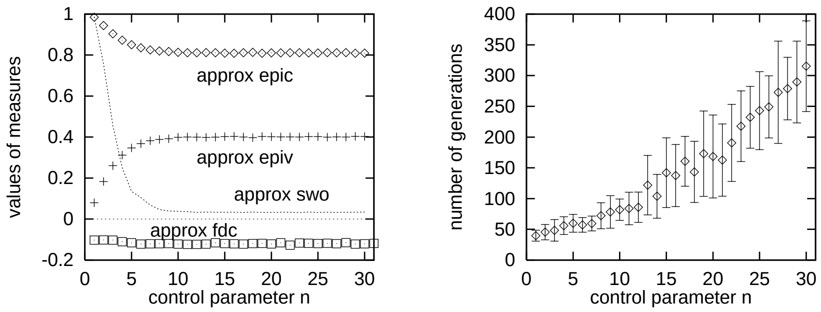


FIG 27 : Epistasis variance, epistasis correlation, fitness distance correlation, site-wise optimization values and number of generations to optimum for the **deceptive mixtures of a OneMax and n zeromax problems**. The string size is 80. The approximations are averaged over 21 values using a random sample of size 2^8 for swo and of 2^{12} for the other indicators. See Sect. II.3.2 for a description of the GA. The same GA with proportional (roulette-wheel) selection fails to detect the optimum within a reasonable length of time due to a too low selection pressure.

II.3.6 EXPERIMENTAL VALIDATION

While the previous section mainly focused on GA-easy function classes, this section considers both easy and hard fitness functions. First, some examples from restricted steepest-ascent function class (Sect. II.3.4.3) are instantiated. The values of the indicators for these functions are calculated on random samples, and compared to observed GA-behavior. Second, difficult-to-untractable examples are considered, from the well-known class of nearest neighbor problems [MPV87, PBS97], which involve only second order epistatic interactions.
The goal of this section is to show the extent to which the –random sample based– values produced by the four indicators can be unrelated to the convergence behavior of commonly used GA instances, irrespective of the latter function being an easy or a hard one.

II.3.6.1 Deceptive mixtures of OneMax and zeromax

Figure 27 shows that the functions f_n belong to the first convergence quality class; the GA can be said to converge fast on them when linear ranking selection is applied. Yet, fitness distance correlation for all the values of n and the site-wise optimization indicator for the first values of n are blinded by the large proportion of completely irrelevant deception. Both epistasis indicators predict a near to first order function for low values of n, and stabilize to some – rather good – intermediate value for higher values of n. Note that none of the indicators sees the convergence speed dropping with n increasing.
A noticeable feature is that proportional selection does not exert enough pressure for the GA to converge, especially when n value increases.

II.3.6.2 Nearest neighbor interaction functions

A one-dimensional nearest neighbor interaction (NNI) function f with cyclic boundary conditions is determined by $3l$ constants, denoted by h_i (the second order components), and k_i and l_i (the first order components), for $i \in \{1, \ldots, l\}$. For the sake of notation, we will assume that $h_l \equiv h_0$ and $s_l \equiv s_0$. The fitness value of a string $s \in E$ is given by

$$f(s) = \sum_{i \in \{1,\ldots,l\}} h_i(1 - s_i)s_{i+1} + \sum_{i \in \{1,\ldots,l\}} (k_i s_i + l_i(1 - s_i)). \tag{148}$$

The origin of these functions lies in statistical physics. The functions correspond to the instances of the anti-ferromagnetic one-dimensional nearest neighbor interaction model, whose energy functional is

$$H = - \sum_{i \in \{1,\ldots,l\}} J_i \sigma_i \sigma_{i+1} - \sum_{i \in \{1,\ldots,l\}} \tau_i \sigma_i. \tag{149}$$

Here the interaction constants J_i take non-positive values, the external magnetic field strengths τ_i take arbitrary values and the spin variables σ_i take values in $\{-1, 1\}$. Functions of this form have been extensively studied; an overview can be found in [MPV87]. Closer to our subject, Prügel-Bennett and Shapiro [PBS97] have made a theoretical study of the GA's dynamic on simple random Ising systems, which are theoretically harder to deal with than the class we are considering, yet are close enough to our class to transfer the results.
It is important to realize that NNI problems can be optimized in linear time by a *divide and conquer* algorithm. It is also possible to calculate their optimum using analytical and numerical techniques from statistical physics.

• **A subclass of NNI functions:** We define a subclass of NNI problems by setting the first order component to zero and the values of the interaction component h_i to $1+(a(i+1) \bmod b)$, for positive $a, b \in \mathbb{N}$:

$$f(s) = \sum_{i \in \{1,\ldots,l\}} \big(1 + (a(i+1) \bmod b)\big)\ (1 - s_i) s_{i+1} \tag{150}$$

Functions in this class range from fast and perfect convergence – quality class 1 and high speed (see Sect. II.3.2) – over occasional fast convergence (quality class 3) to occasional slow convergence (also quality class 3). Figure 28 shows the values of the four indicators for a selection of 29 NNI problems from the class parametrized by a and b. It is clear that none of the indicators can make a distinction between the different classes of convergence quality. The influence of the selection operator on the GA's performance is drastic. Instances whose optimum is occasionally found with proportional selection are found to be trivial for a GA with ranking selection, and vice versa.

• **Balanced NNI functions:** We can make the NNI problems extremely hard for both the GA and simulated annealing by applying a technique we call *balancing*. We want to achieve that for each site i, the equation $h_{i-1} + 2k_i = h_i + 2l_i$ is satisfied. This can easily be accomplished by changing the first order component of the function. Translated to the traditional notations from statistical physics, balanced NNI's correspond to instances of the nearest neighbor interaction model without external magnetic field (i.e., with energy functional $H = -\sum_i J_i \sigma_i \sigma_{i+1}$).

The GA is unable to detect the optimum of any of the balanced versions of the NNI problems presented in Fig. 28, no matter how long or how often it is run. We refer to [JSC91] and [PBS97] for the reasons behind this.
On the other hand, all indicators (not presented) except epistasis variance show extremely high values (in distance to their respective reference class), a fact which is not difficult to prove. Let us outline the proof for the case of the site-wise optimization indicator:

The expected contribution of a site of a given string equals half of the second order component of the previous site plus the complete first order component of the current site. This is because

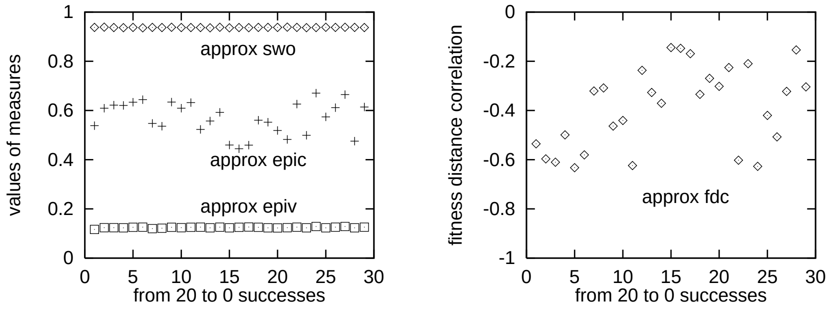


FIG 28 : Epistasis variance, epistasis correlation, site-wise optimization values and fitness distance correlation for a **selection of 29 unbalanced nearest neighbor interaction problems** on 80 sites, as defined in Sect. II.3.6.2. The problems are ordered according to the number of runs which reached the optimum. The success rate is optimal for the six leftmost problems, and is zero for the two rightmost. The approximations are averaged over 21 values using a random sample of size 2^8 for swo and of 2^{12} for the other indicators. We refer to Sect. II.3.2 for a description of the GA. Applying the same GA but with proportional selection results in a different ordering. This does not change the conclusions about the indicators, however.

the probability that there is a 0 at the previous site equals the probability of there being a 1, in which case the second order component can never be included.
If a problem is balanced, the expected contribution of a site is independent of the value at that site. Hence, the sample containing the strings modified by the site-wise optimization algorithm, contains as many strings with a 0 on a given site as there are strings with a 1 on that site; which results in a site-wise optimization value of 1.

II.3.6.3 Mixtures of OneMax and nearest neighbor interaction functions

The construction of these mixtures is similar to that of the deceptive mixtures defined in Sect. II.3.21, and Fig. 26 can be used as a guide to it.

In this section, we substitute the zeromax problems by a balanced nearest neighbor interaction problem with $a = 5$ and $b = 7$. To ensure that the mixtures are restricted steepest ascent optimizable, we have to adjust the translations such that the fitness ranges of each schema involved in the definition are disjoint. This is not a problem since we can easily compute the optimum of the balanced problem on $l - n$ sites.

• **Results:** Figure 29 shows that the new mixtures (OneMax+NNI) still belong to the first convergence quality class (Def. II.3.7), as the ranking selection GA converges rapidly. Again, proportional selection does not exert enough pressure for the GA to converge.

Considering the mixture of a OneMax and n zeromax problems of Sect. II.3.6.1 and Fig. 27, the substitution of the zeromax problems by the NNI problems (Fig. 29) only effects the values of the site-wise optimization indicator, which stabilize to a high value of 0.983 with n increasing. Recall that in the previous case (OneMax+ZeroMax), when the control parameter n becomes greater than, e.g. 6, the site-wise optimization indicator cannot be fooled anymore by the deceptive components.
Fitness distance correlation and the site-wise optimization indicators are blinded by the ruggedness of the landscape which is completely ignored by the GA concentrating on the translations. Again, as in the case of Sect. II.3.6.1, the epistasis indicators stabilize to an

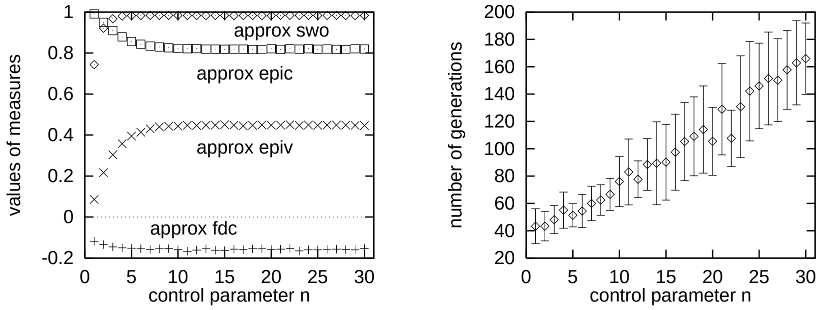


FIG 29 : Epistasis variance, epistasis correlation, fitness distance correlation, site-wise optimization values and number of generations to optimum for **mixtures of a OneMax and n balanced nearest neighbor interaction functions**. The string size is 80. The approximations are averaged over 21 values using a random sample of size 2^8 for swo and of 2^{12} for the others. See Sect. II.3.2 for a description of the GA. As with the other mixtures, the same GA with proportional (roulette-wheel) selection fails to detect the optimum within a reasonable length of time due to a selection pressure that is too low.

intermediate value, when n increases; and none of the indicators sees the convergence speed dropping with n increasing.

Note that the epistasis variances obtained for the mixture of OneMax and NNI problems (around 0.4) in Fig. 29 are even higher than those obtained for the NNI problems (around 0.1) in Fig. 28, although the former problems are a lot easier for a GA to solve than the latter ones (quality 1 vs quality 3 and 4). The same phenomenon of misleading prediction is observed with fitness distance correlation. The two other indicators, epistasis correlation and the site-wise optimization indicator, hardly see any difference between the two cases.

II.3.7 RELATED WORK

This section relates the results of the previous sections to another attempt at improving the epistasis indicator, and proposes some guidelines for further work.

II.3.7.1 The Experimental Design perspective

Reeves and Wright's Anova tables [RW95b] contain a lot more information about the epistatic structure of a fitness function than the epistasis indicators do. In short, these tables also provide, apart from the deviation from the first order functions, information about the higher order interactions present in the fitness function.

While this technique gives more quantitative information about the epistatic structure of the fitness function, and certainly has to be preferred over epistasis variance or correlation for that matter, the question remains whether there is a correspondence between the epistasis information and the convergence behavior of the genetic algorithm. As can be deduced from our experiments with nearest neighbor interaction problems, it is not the amount of interaction which influences the convergence. Rather, one has to search for the presence of symmetry or frustration within the interaction. The latter phenomena already occur in second order interaction problems. We refer to [NV98] for a more detailed discussion of second order interaction.

Interaction does not necessarily yield a negative effect on the convergence behavior of the GA. We refer to the NLM_n functions defined in Sect. II.3.3.1: the more interactions we add (i.e., the higher the exponent n), the faster the GA with proportional selection converges.

We conclude that quantitative information about interactions is not a priori related to the convergence behavior of the GA.

II.3.7.2 Random sampling versus Metropolis sampling

It is very easy to make up fitness functions which yield close to minimal observed values for the indicators, yet are very hard to optimize for the GA. Consider the OneMax problem modified by inserting a balanced nearest neighbor interaction problem in the subspace $1^{(n)}\#^{(l-n)}$, for some large enough n and l. Random samples of size less than 2^n will typically only carry information about the OneMax component of the fitness function, which results in minimal observed values for all indicators presented in this section (and any other indicators based on random samples).

Using the Metropolis algorithm, which is the basic algorithm behind simulated annealing, it is possible to construct samples of the search space with the property that the mean fitness value of the sample equals a given value f_β (where β is the inverse temperature; it suffices to know that theoretically there is a one-to-one correspondence between temperature and average values). In this way, we can obtain information about the structure of the fitness landscape as the GA sees it when the average fitness value of its population has already surpassed the average fitness value of the function.

Note that this technique also gives an estimation of the number of strings which share a given fitness value. The *density of states* [REA96] of a fitness function is computed from these data. It is presented as a absolute optimization hardness indicator.

Similar to the technique of obtaining information at a higher fitness value level is the technique of computing the values of the indicators using the information contained in successive generations of a GA run. This typically gives better insight in the structure of the search space than, for example, the exact values of the indicators, as the latter can be biased by a large and completely irrelevant part of the search space.

II.3.8 Random sampling versus on-line sampling

Many papers [CL96], [DSG95] stress the nonsense of speaking of problem difficulty without considering all the parameters of the algorithm at hand: [CL96] presents two unimodal functions on the binary space. Depending on whether elitism is used or not, the same sampling technique (ex. steepest ascent hill climber) yields an exponential difference in convergence time.

Authors explain that the potentialities of an operator are not necessarily exploited by the algorithm. Such differences are unmeasurable, of any static landscape based indicator.

Considering samples obtained from successive GA populations, may relax this problem of unused potentialities of the operator graph.

In that streamline, section II.2 of this chapter proposes to precise the notion of GA performance from the analysis of a given run of the GA at hand ...

II.3.9 CONCLUSIONS

This paper widely investigates and extends some statistical tools for landscape analysis, mainly showing that they are too weak to detect a host of functions which are easy for the GA. The "GA-easy" statement is defined relatively to the ranking based GA considered in this paper. However, as the notion of GA performance can vary according to problem domains and user's prerequisites, the reliability of indicators also depends on problem domains. The conclusions of this section are directly related to the class of functions involved here, and more importantly, to the definition we consider for the statement "GA-easy" (first convergence quality-class here).

The lesson from these experiments, is that Epistasis variance and correlation, fitness distance correlation and the site-wise optimization reliably detects first order functions resp. linear and monotone functions, when presented with the relevant information. Otherwise, we can find fitness functions which are very easy for the GA and whose indicators values can take any arbitrary value.

We also summarize a number of key properties of the fitness landscapes undetectable by these indicators, for instance constantness, scaling and averaging. Using these properties, we construct and locate – within known GA-easy functions – subclasses of functions that illustrate the indicators limitations. These examples also define a test context for further GA-hardness indicators. Hence, we build some requirements for new and more efficient indicators.

To sum up, the key problem in designing a summary statistic is to find a characterization of a broad class of fitness functions on which the GA at hand behaves similarly. Clearly, monotonicity and unimodality –investigated in this section– are too small to satisfy this requirement. Note that finding a good definition of "similar behavior" is also a major problem (that can be addressed using the ratios of behavioral measures of section II.2).

On the other hand, the tremendous influence of selection pressure (among other parameters) on the convergence behavior of the GA gives further evidence for the calls for on-line measures (behavioral measures of section II.2) and for additional indicators that take into account GA parameters as those of section II.6.
However, before turning to that direction, the following Section investigates the fitness–distance relation, beyond correlation value. In fact, the correlation plot contains much more information than the mere correlation value.

II.4 On Functions with a Given Fitness–Distance Relation

Overview The whole Fitness Distance plot (FD plot) gives much more information than the summary correlation value (FDC) studied above. Hence, this section investigates whether an interpretation of the FD plot can yield reliable information about GA's behavior. We present a generic method for constructing fitness functions which share the same fitness versus distance-to-optimum relation (FD relation). Special attention is given to FD relations which show no local optimum in the correlation plot, as is the case for the relation induced by Horn's longpath. We give an inventory of different types of GA behavior found within a class of fitness functions with a common correlation plot. We also show that GA behavior can be very sensitive to small modifications of the fitness–distance relation.

II.4.1 INTRODUCTION

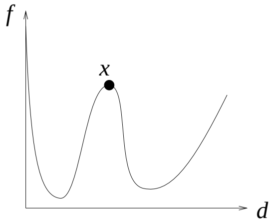
A number of summary statistics have been proposed to characterize classes of fitness functions with similar GA behavior. We mention fitness distance correlation [JF95], correlation length and operator correlation [MdWS91], epistasis [Dav91a], schema variance [RS95a] and hyperplane ranking [DWP95]. These summary statistics originated from empirical observations and their efficiency is widely discussed (e.g., [RVSK97, Alt97, QVSG98]).

Studies of the above properties and statistics have mostly been limited to the construction of examples and counterexamples. An interesting direction, scarcely taken, is to construct and study classes of fitness functions which share the same property or statistic value, and to understand the differences in GA behavior on these functions.

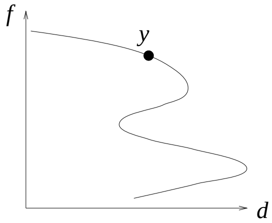
Exemplar for the latter approach are the NK-landscapes [Kau89] which provide a simple way to construct classes of functions with a given amount of epistasis (order of interaction between genes). The study of NK-landscapes gives useful information about the relation between the structure of the landscape and the order of epistasis. Note that the notion of epistasis is not to be confused with Davidor's epistasis indicator, which cannot distinguish between different orders of epistasis (see. II.3.3.1).

Fitness distance correlation indicators the degree to which the fitness values increase with the individuals approaching the optimum in Hamming distance. The results of section II.3 show that the values produced by this statistic are often unrelated to the GA's behavior. To summarize, unless one restricts its scope to some narrow function class, the only reliable information that the fitness distance correlation gives about the difficulty or structure of the fitness landscape is –at most– the sign of the correlation coefficient value.

Of course, the whole fitness distance plot gives more information than the summary statistic (correlation value). Section II.5 shows how the sample lay out in the fitness–distance space and diversity comparisons can be reliable for choosing problem-dependent initialization procedures. But further interpretation of this plot can be dangerous, since only partial information about the structure of the fitness landscape is displayed. A local optimum in the fitness landscape is not necessarily visible in the fitness distance plot. For example, the point x in figure (a) below is both a local optimum in the fitness landscape and in the correlation plot. The point y in figure (b) might or might not be a local optimum in the fitness landscape, depending on the distribution of the neighbors of y.



(a) peak



(b) no peak

Hence "continuous" lines in the FDC correlation plot do not necessarily define a feasible path to the optimum of f, although in an intuitive way this may seem to be the case. They may even hide intractable local optima.

In order to study the effects of invisible information in the plot (e.g., the presence of local optima, the number of genes represented by one point, etc.), we will focus on fitness–distance relations which show no local optimum in the correlation plot, as is the case in figure (b). What difference can we expect in GA behavior for a fixed FD relation?

This section is organized as follows: Section II.4.2 introduces some conventions and the main results of this section. Section II.4.3 presents a generic method for constructing fitness functions which follow a given fitness–distance relation, with Horn's longpath [HG95] as the main example. The method is then used in Section II.4.4 to construct a class of functions whose members share the same FD relation. We make an inventory of the different types of GA behavior observed in this class, and study the effects of small modifications in the FD relation. Finally, Section II.4.5 addresses the same issues, but with a consequent modification of the out-shape of the FD relation. It gives further evidence for the fact that the correlation plot can be completely unreliable to distinguish a straightforward easy problem from an intractable one.

II.4.2 MAIN RESULTS

Experimental conditions: Unless explicitly stated, we use a generational GA with population size of 50, linear ranking selection, uniform crossover at rate 0.5, and $1/l$ mutation with a rate of 0.5 per individual. The stopping criterion is set at 100,000 fitness evaluations without improvement of the best value, with a maximum of 1,000,000 fitness evaluations. All results are averaged over 50 independent runs.

Notations: we propose the following hierarchy of the *convergence quality* of a given GA and fitness function (more details can be found in section II.3.2):

1. The GA detects the optimum of the fitness function at each run (with a small [53] variance on convergence time).

2. Each run of the GA still detects the optimum of the fitness function, but now the variance on the number of iterations needed for detection is comparatively large.

3. Only some GA runs detect the optimum of the fitness function.

4. The GA does not perform a better than random search. It cannot detect the optimum of the fitness function.

[53] We consider that the variance is small if it is at a lower order of magnitude than convergence time.

Summary of the results: Section II.4.4 studies a class of fitness functions f_k sharing the same fitness-distance relation of Eq. 158, sketched in Fig. 32. It shows that according to the strings distribution in the plot (choice of k value in Eq. 159), GA convergence can be either stable and straightforward (convergence quality 1), or unstable (convergence quality 2). Moreover, GA trajectories in the correlation plot vary according to the choice of the rule. However, changing the rule yields a consequent **change of strings density** in the correlation plot, which partially explains the observed differences.

A mostly interesting result is obtained with the function f'_1, which follows the same FD relation than f_1, and is equal to f_1 for points with below average distances (to optimum) only. Yet, most GA runs fail to find the optimum of f'_1 (convergence quality 3, vs. 2 for f_1). This is due to the presence of an intractable local optimum w.r.t. *hamming* distance, in the middle of the *upper path* (the dot line with highest fitness values in the correlation plot) of f'_1.

On the other hand, a **small perturbation of the fitness values** allows us get rid of local constantness in the landscape. Applied to f_1 GA convergence quality (initially 2) is 1. Whereas GA behavior is still the same (quality 3) when the perturbation is applied to f'_1.

To sum up, GA convergence quality takes values 1 (f_k, $k >> 1$), 2 (f_1) and 3 (f'_1), on the same correlation plot. A **small translation** of the fitness-distance relation **along the distance axis,** transforms f'_1 into an intractable problem (quality 4). Perturbing fitness values of the plot (to get rid of local constantness) does not change the convergence behavior with f'_1, however, with f_1 convergence quality is 1.

Section II.4.5 considers a more general fitness-distance relation, where the whole correlation plot is covered. It shows similar results to those of the first section: convergence quality 1, 3 and 4 can be observed with functions sharing the same correlation plot. A different rule is proposed that completely changes GA trajectory in the correlation plot.

II.4.3 GENERAL CONSTRUCTION

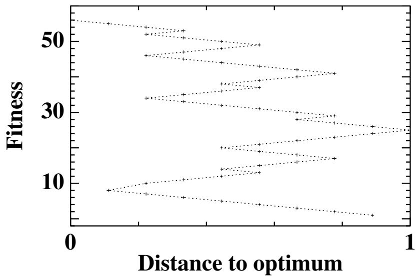
In this section, we propose a way of constructing fitness functions which share the same fitness–distance relation. The shape of this relation, depicted in Fig. 30, is similar to that induced by Horn's long path [HG95]. Only successive points on Horn's long path are within Hamming distance 1 from each other; they define a unique feasible path for a hill climber.
Let us drop the requirement of proximity of neighboring points on the path, and constrain the relation between the fitness of a string and its distance to the optimum to functions of the form

$$H(\mathit{fitness}) = \mathit{distance\text{-}to\text{-}optimum}, \tag{151}$$

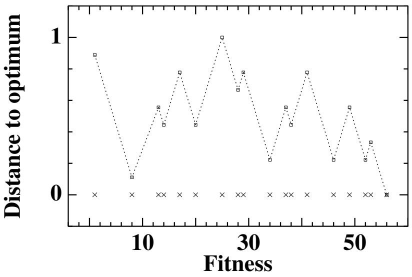
with

$$H : [0, f_{\max}] \to \{0, \ldots, \ell\}, \tag{152}$$
$$H^{-1}(0) = \{f_{\max}\}. \tag{153}$$

In this case, no local optimum is visible in the correlation plot, and we obtain a long-path shaped FD relation (each fitness value is associated with a unique distance). Note that the choice of optimum is arbitrary; unless otherwise mentioned, it is set at $s^* = 11\ldots11$.



(a) Correlation plot



(b) Relation H

FIG 30 : *(a) Correlation plot of Horn's long path problem. (b) FD Relation H for Horn's long path problem. The points marked on the fitness axis are the extrema $f_1, \ldots, f_K$ of the function H.*

All fitness functions satisfying FD relation H can now be described by assigning values to a *rule* function, which maps the strings to natural numbers. *rule*(s) indicates which branch of the plot (at a given distance from optimum) will be chosen as the fitness of string s. It obeys

$$\forall s \in E : rule(s) \in \{1, \ldots, \#H^{-1}(\mathrm{d}(s, s^*))\}. \tag{154}$$

A fitness value can be assigned to each string by following the convention that lower *rule* values correspond to lower fitness values, i.e., if $rule(s) = 1$ and $d = \mathrm{d}(s, s^*)$, then the lowest fitness value of the set $H^{-1}(d)$ is assigned to s. If $rule(s) = k$, with $k \leq \#H^{-1}(d)$, then the k-th greatest fitness value of the set $H^{-1}(d)$ is assigned to s.

Example of Horn's long path rule: As an example, we give the *rule* function for Horn's long path, assuming that we have already chosen an adequate function H, such that the obtained correlation plot is that of the long path (see Fig. 30). We denote the extrema of H by $f_1, f_2, \ldots, f_K$. The optimum of the long path is reached in $110\ldots0$.
Horn, Goldberg and Deb [HG95] give a recursive algorithm which either returns the position of a string in the long path or indicates that this string is off the path. Let *pos*(s) be the position of s in the path. It ranges from 1 to $3.2^{(\ell-1)/2} - 1$, which is the length of the path when ℓ is odd. The special value $pos(s) = 0$ indicates that s is off the path.
Armed with this *pos* function, we define the *rule* function by

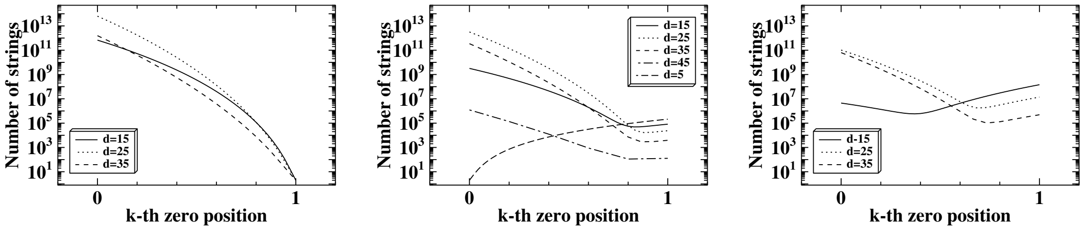
$$rule(s) = \sum_{i=1}^{K} i\,[p(f_i) < pos(s) < p(f_{i+1})], \tag{155}$$

where $p(f_i)$ corresponds to the position in the path of the (unique) string at fitness f_i. The brackets are used for the indicator function. The $p(f_i)$ are determined recursively by

$$p(f_1) = 1, \tag{156}$$
$$p(f_{i+1}) = H(f_i) - H(f_{i+1}) + p(f_i). \tag{157}$$

II.4.4 CASE STUDY OF A FITNESS–DISTANCE RELATION

The first part of this section uses the method presented in the previous section to construct a class of fitness functions whose members share the same fitness–distance relation. The properties of some of the fitness landscapes in this class are investigated.



(a) $k=1$ (b) $k=5$ (c) $k=9$

FIG 31 : *Number of strings in logarithmic scale versus normalized k-th zero position in* $[k, \ell + k - d(s, s^*)]$, *with* $\ell = 50$.

The second part of the section gives an inventory of the different types of GA-behavior found in this class. The effects of small modifications in the FD relation and different distributions of the individuals are also studied.

The construction of function class $\{f_k\}$: Let us take sinusoidal assignments of values for the function H, parametrized by the number of periods κ:

$$H(x) = \frac{\ell/2}{x_{\max} - \pi} \left(x \sin(x) + x_{\max} \right), \tag{158}$$

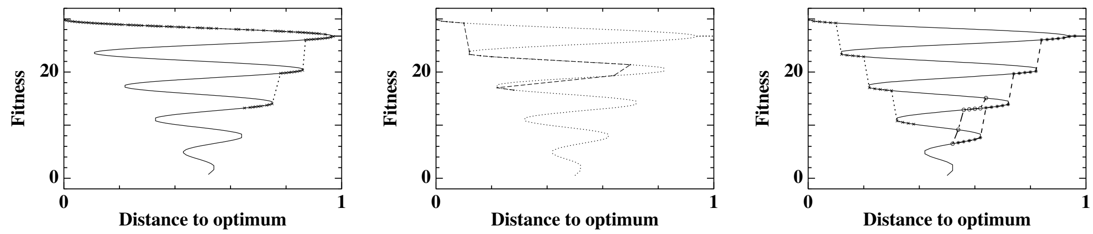
for $x \in [0, x_{\max}]$, with $x_{\max} = 2\pi\kappa - \pi/2$. After a suitable restriction of its domain, H satisfies the conditions of Eqs. 152 and 153. For the sake of symmetry, and to avoid a bias towards above average distances, the function H is slightly translated to the left, which centers the plot around $\ell/2$ (see Fig. 32). The fittest string $s^* = 11\ldots11$ is at the leftmost position. The string represented by the rightmost position of the plot, which is in Hamming distance closest to $00\ldots00$, and on the sine curve, is noted s^0.
Having defined the relation H between fitness values and the distance of the strings to the optimum, we proceed by defining a parametrized class of *rule* functions, denoted $\{r_k\}$, hereby obtaining a class of fitness functions $\{f_k\}$ satisfying the FD relation H.

Notations: Given a string s, we denote by $Z_k(s)$ the position of the k-th zero of s. As the range of $Z_k(s)$ is $\{k, \ldots, k + \ell - \mathrm{d}(s, s^*)\}$, we call $(Z_k(s) - k)/(\ell - \mathrm{d}(s, s^*) + 1)$ the *normalized k-th zero position.*
The rule r_k is defined in such a way that the further the k-th zero is in a string, the higher the fitness of this string is:

$$r_k(s) = \left\lfloor \frac{Z_k(s) - k}{\ell - \mathrm{d}(s, s^*) + 1} \# H^{-1}(\mathrm{d}(s, s^*)) \right\rfloor + 1, \tag{159}$$

for all strings s and positions $k \in \{1, \ldots, \ell\}$. Since this rule takes values between 1 and $\#H^{-1}(\mathrm{d}(s, s^*))$ inclusively, it follows the conditions of Eq. 154, and therefore, defines a fitness function f_k following the fitness–distance relation H of Eq. 158.
Figure 31 shows the distribution of strings at distance d from the optimum in function of the normalized k-th zero position.
Note that the rule is defined in such a way that the further the k-th zero is in a string, the higher the fitness of this string is. Hence, the plot also characterizes the distribution of strings in the fitness–distance plot. We observe that

- there are more low-fit strings at distance $\ell/2 + a$ than at distance $\ell/2 - a$. Similarly, there are less high-fit strings at distance $\ell/2 + a$ than at distance $\ell/2 - a$;



(a) 85% of the GA-runs (b) 15% of the GA-runs (c) possible HC behavior

FIG 32 : *The GA and hill-climber trajectories (best individual so far for each generation) plotted on the correlation plot for the function f_1 associated with rule r_1. In (a) the GA trajectories go straight up to s^0, then follow the difficult upper path to converge to the optimum. In (b) the GA does not go through the difficult upper path, and converges rather quickly. Three sorts of one-bit-flip hill-climber behavior are shown in (c). Of all the runs, about 50% take the path along the left side to s^*, 25% take that along the right side to s_0, and 25% get rapidly stuck in a local optima inside the cone of H.*

- if $k = 1$ then, at a given distance from optimum, the number of strings decreases when fitness increases (Fig. 31(a)).
- as k increases, the number of strings with high fitness values increases (Fig. 31(b,c)).
- when $k > (\mathrm{d}(s, s^*) + 1)/2$, there are more strings at top than at bottom fitness values (Fig. 31(b) with $\mathrm{d}(s, s^*) = 5$ or Fig. 31(c) with $\mathrm{d}(s, s^*) = 15$).

Note that strings at distances beyond $\ell/2$, with minimal k-th zero position ($Z_k(s) = k$), define an easy path up to s^0. In fact, within these points, the fitness function is determined by the Hamming distance towards a point of s^0. The same thing happens symmetrically: strings at distances shorter than $\ell/2$ define an easy path (determined by the Hamming distance) to s^*. These two paths will be called the **paths along the left-hand and right-hand side** in the next sections. We also define the **upper path** as the points of the correlation plot that link s^0 to $s*$ via highest fitness values.

Hill-climber behavior on the function class $\{f_k\}$: The f_k-landscape of the one-bit-mutation hill-climber contains many local optima, while experiments show almost none for the landscape of the GA defined in Sect. II.4.2. Figure 32(c) presents the different kinds of hill-climber trajectories:
1. For $k = 1$, the one-bit-mutation hill-climber and the $1/l$ mutation (1+1)-, (1+50)- and (50+50)-evolution strategies show the same behavior: either they take the easy left-most path to the optimum, or they get stuck in some local optimum (which is often one of the strings in s^0).
2. As k increases (not shown), most HC runs (e.g., 70% when $k = 5$) get stuck in a local optimum within 100 fitness evaluations. On the other hand, the GA and the evolution strategies perform increasingly well.

GA-behavior on the function class $\{f_k\}$: Let us now concentrate on the behavior of the GA on the functions $\{f_k\}$. Two types of GA-trajectories can be observed in the plots of Fig. 32(a,b):
1. Some runs go inside the fitness–distance plot, as in Fig. 32(b). They always find the optimum within 100 generations.

2. The other runs follow the right side and then take the upper path from right (s^0) to left (s^*), as in Fig. 32(a). We note that:

1. When $k = 1$, this path is followed by the majority of runs. It takes the GA a very long time (1000 up to 5000 generations) to move from the right side of the upper path to the left side. Convergence is quicker on the left half of the upper path.

2. With k increasing, the convergence time on the upper path drops below 400 generations, but the number of runs which follow this path decreases.

The distribution of strings in the fitness–distance plot partially explains the observed GA-behavior, since the GA is mainly attracted to dense and high-fit areas. We distinguish again between $k = 1$ and k increasing:

1. When $k = 1$, the dense areas define an easy path along the right-hand side to s^0 because of the conic shape of H. Therefore, most GA-runs take this path to s^0. Then, due to elitism, the GA necessarily follows the upper path from s^0 to s^*.

2. With k increasing, the number of high-fit strings increases, and the search on the upper path becomes easier.

The difficulties on the upper path are due to *isolation*. For all points s on the upper path, at a distance $\mathrm{d}(s, s^*) > \ell/2$, the next fitter strings s' are still on the upper path, but at a shorter distance from the optimum. The range of possible values of $Z_k(s')$ is therefore larger than that of $Z_k(s)$. Moreover, given the shape of H, there are probably more fitness values to allocate for s' than for s. A direct consequence of these facts, is that $Z_k(s')$ is likely to be greater than $Z_k(s)$. Hence s and s' are likely to be at a large Hamming distance from each other. Between the strings are low-fit areas.
This picture is reversed for below-average distances from the optimum (i.e., for string s with $\mathrm{d}(s, s^*) < \ell/2$). The number of different fitness values either decreases or remains constant. Each time it decreases, the search becomes easier.

We have now shown that different types of GA behavior arise, on the same fitness–distance relation H, according to the distribution of strings over the possible function values (the parameter k tunes this distribution).
In the next sections II.4.4.1 and II.4.4.2, we stick to the case of $k = 1$, and consider two modifications of the rule of Eq. 159. The first modification removes the plateaus in the fitness landscape which are responsible for the problem of isolation described above. The second modification adds symmetry to the distribution of strings with respect to the distance $\ell/2$.

II.4.4.1 Removing the plateaus with the function f_1^+

The minor modification made to f_1 to obtain f_1^+ is aimed at showing the obvious limitations of any statistical indicator of the fitness landscape which is based on fitness values. It also shows the sensitivity of GA's trajectory to local features of the landscape.
We construct f_1^+ by modifying f_1 (which corresponds to rule r_1 defined in Eq. 159):

$$f_1^+(s) = f_1(s) + \sum_{j=2}^{\ell/5} \frac{Z_j(s)}{\ell}\, 10^{-j+1}, \qquad (160)$$

for all strings $s \in \{0,1\}^\ell$. The effect of the additional terms is a small cloud around each point in the fitness–distance plot: the flat areas have been removed by the small differences (less

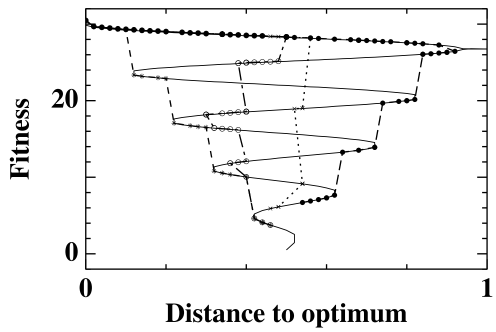


FIG 33 : *Typical (1+1)-ES trajectories shown on the fitness–distance plot of the function f_1^+. This function is straightforwardly optimized by the (1+1)-ES. Note that the small clouds around each point of the plot are invisible.*

than 10^{-1}) in fitness values. Yet, the macroscopic overall fitness–distance relation remains the same: the density of strings at each fitness level ($\pm 10^{-1}$) are the same. The fitness differences between any two strings in different fitness–distance positions remain the same, at precision 10^{-1}.
The Hamming landscape of f_1^+ keeps its many local optima. The one-bit-flip hill-climber shows the same three kinds of trajectories, as plotted in Fig. 32(c) for f_1.
But evolution strategies behave differently. While the (1+1)-ES finds the upper path of f_1 intractable, it swiftly optimizes the upper path of f_1^+ (convergence within 6000 fitness evaluations). Moreover, only 30% of the runs (vs. 50% for f_1) take the path along the right-hand side, as shown in Fig. 33.
The GA behaves similarly to ES: it converges within 300 generations when following the right-hand side and the upper path of f_1+ (vs. 1000 to 5000 generations for f_1). The GA can also easily escape from the absorbing right-hand side of f_1, and go inside the cone of H.
It is important to note that any statistical *a priori* indicator based on fitness values would give roughly the same results for f_1^+ and f_1. Yet, the GA's speed of convergence is completely different for both functions. The upper path, initially intractable for the (1+1)-ES, becomes trivial to optimize for this algorithm. The observed trajectories of GA and evolution strategies are different too: in the case of f_1^+ they often go inside the cone, rather than taking the absorbing path along the right-hand side when confronted with f_1.

II.4.4.2 Adding symmetry to the distribution with f_1'

Again we make a small modification to the fitness function f_1, this time by changing the rule r_1. The result is a fitness function which satisfies the fitness–distance relation H, but is intractable for the GA due to a local basin of attraction that is too large.
Recall that the rule r_k of Eq. 159 distributes more high-fit strings at distances beyond $\ell/2$ than before $\ell/2$, as can be seen in Fig. 31. To make symmetric the distribution of individuals in the plot with respect to $\ell/2$, we propose an alternative rule for above average distances.

Let us use $O_k(s)$ to denote the position of the k-th one in s. Then we define:

$$r'_k(s) = \begin{cases} \left\lfloor \dfrac{Z_k(s) - k}{\ell - \mathrm{d}(s, s^*) + 1} \# H^{-1}(\mathrm{d}(s, s^*)) \right\rfloor + 1, \\ \qquad\qquad\qquad \text{when } \mathrm{d}(s, s^*) \leq \ell/2, \\ \\ \left\lfloor \dfrac{O_k(s) - k}{\ell - \mathrm{d}(s, s^*) + 1} \# H^{-1}(\mathrm{d}(s, s^*)) \right\rfloor + 1, \\ \qquad\qquad\qquad \text{when } \mathrm{d}(s, s^*) > \ell/2. \end{cases} \tag{161}$$

Associated with the rules $\{r'_k\}$ is a class of fitness functions $\{f'_k\}$, whose members follow the same fitness–distance relation H as the $\{f_k\}$. The functions f'_k and f_k coincide for strings at distance $d < \ell/2$ from optimum.

Due to the conic shape of H, f'_1 still contains an easy path to s^0, but only via strings with minimal k-th one position. Experiments show that 70% of all GA-runs take this easy path to s^0, similar to the case of f_1. From s^0 they start on the upper path towards the optimum. In the case of f'_1, though, the search stops in the middle of the plot, around strings with $\mathrm{d}(s, s^*) = \ell/2$.

In fact, strings on the upper path with below average distance have their k-th zero at the highest position, and are very close to the string $s_{10} = 11\ldots1100\ldots00$. Strings with an above average distance have their k-th one at the highest position too, and hence are close to the bitwise complement of s_{10}, which is $s_{01} = 00\ldots0011\ldots11$. It is thus not surprising that the runs which start the upper path in s^0 get stuck in some intractable local optimum around $\ell/2$.

It is interesting to note that, while involving a strong local optimum, the upper path exhibits a locally ideal fitness–distance correlation (value -1). On the other hand, the local optimum in the GA landscape does not exist in the landscape of a GA with complementary crossover. Removing the plateaus helped to transform the upper path of f_1 into a (1+1)-ES-easy function. This is not the case for f'_k: there is no feasible path linking s_{01} to s_{10} via strings with either a high k-th zero position or a high k-th one position. Thus, removing flat areas does not necessarily make fitness landscapes easier.

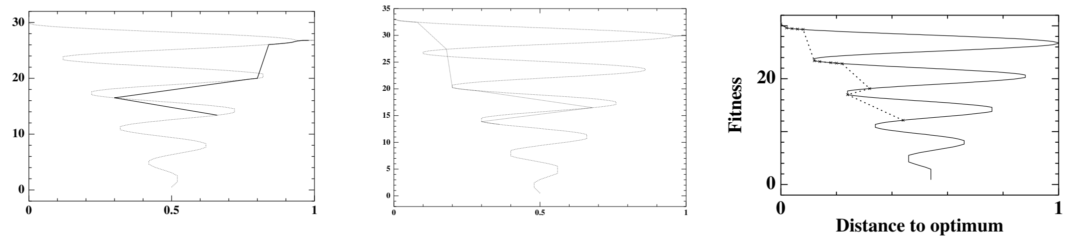
We also tested a minor changes in the FD relation H of Eq. 158:

1- We replaced the sine function by a cosine to get the mirror image of H over the vertical $\ell/2$ axis as shown in Fig. 34-b. Where for FD relation $H$and function f'_k, most (64%) GA-runs take the path along the right, and get stuck in s_{01}, experiments show that for the mirror of H, denoted H^c, opposite results hold: few (40%) GA-runs take the path along the right side. On the other hand, since the distribution of strings in f_k is asymmetric with respect to $d\ell/2$ (there are more low-fit strings at above average distances), it is not surprising that moving to H^c scarcely changes the GA's behavior on f_k (with H70% runs take the right path, versus 50% with H^{sym}).

2- We also tested the original sine function (not centred over the vertical $\ell/2$ axis) as in Fig. 34-c, then only two runs out of fifty follow the right+upper path with function f_1.

Note that the conic shape of H results in two attractors, s^0 and s^*, for the GA, which generally starts at average distances from s^*. One can expect that a different shape of the FD relation Hwill completely change the GA's behavior.

The following Section proposes a case study of the GA-behavior on functions with a FD relation which gives a better coverage of the fitness–distance plot.



(a) H (b) H^c (c) H_0

FIG 34 : *Typical GA behavior according to the reference function choice for the fitness-distance relation: (a) Function H (b) The mirror image of H over the vertical $\ell/2$ axis (c) The original sine-based reference function, asymmetric with respect to the vertical $\ell/2$ axis.*

II.4.5 SECOND FITNESS–DISTANCE RELATION

In the following, we consider a new fitness–distance relation G almost equal to a sin function, as shown in Fig. 35. This addresses a more general case of fitness–distance relations (better coverage of the fitness–distance plot).

The construction of function class $\{g_k\}$: The fitness–distance relation G and the same rule of eq. 159 (position of the k-th zero) define a new fitness function g_k.
Three kinds of GA behaviors have been observed with g_1 (Figure 35):

1. The GA tends to increase its fitness rapidly inside the plot (35-c,d), ending in the difficult upper path to optimum. According to the point where it reaches the top path, convergence speed varies from (d) 800 to (c) 18000 generations.

2. The GA follows the easy bottom path, and quickly reaches the optimum (50 to 100 generations, Figure 35-a).

3. The GA starts on the easy bottom path, then finds a fitter individual at a higher path before reaching the optimum. The search switches there and convergence time increases (400 generations in case 35-b).

If the number of periods of the sine function increases (5 for example), only the first behavior case is observed: GA trajectory goes inside the plot ending in the upper path. Then, in most cases, it gets stuck somewhere on the upper path. On the other hand, function g_1^+, as defined in Eq. 160, is trivial for (1+1)-ES, yet intractable for a 1-bit-flip HC.
Hence convergence quality increases from 2 (3 periods) to 3 (5 periods) with the number of periods.

The construction of function class $\{g'_k\}$: Consider now the number $N(s)$ of contiguous blocks of ones or zeros in a string s at distance d from optimum. The feasible range of N(s) equals:

$$I(s) = \begin{cases} [2, 2d+1] & \text{if } 0 < d \leq (n-1)/2 \\ [2, d+1] & \text{if } (n-1)/2 < d < (n+1)/2 \\ [2, 2(n-d)+1] & \text{if } d \geq (n+1)/2 \end{cases} \tag{162}$$

Note that $I(s)$ length is symmetric with respect to $d = l/2$. Therefore the following rules define a symmetric repartition for strings in the fitness–distance plot.

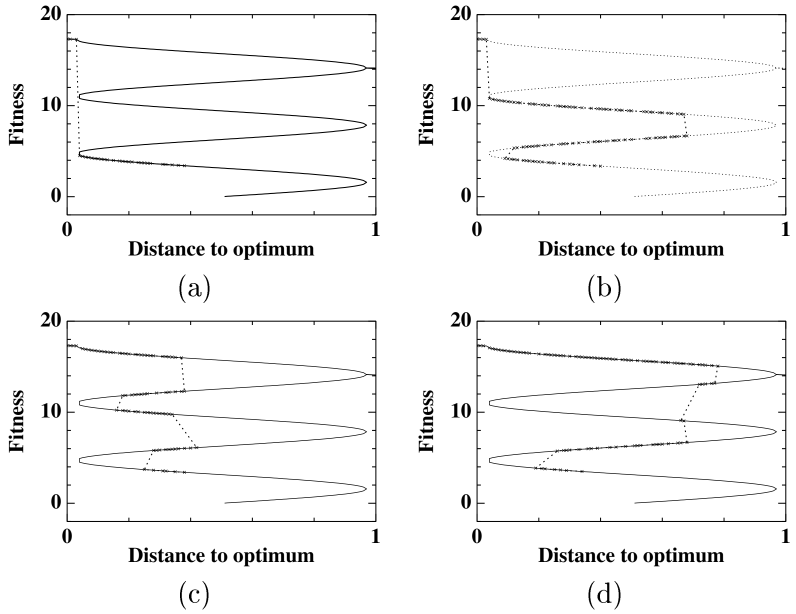


FIG 35 : *Position of 1st zero position in feasible range: different GA behaviors.*

Function g'_1

$$r(s) = \frac{N(s) - 1}{|I(s)|}\ |G^{\ -1}(d)|,$$

where $d = d(s, s^*)$, $I(s)$ (Eq. 162) the range of feasible values of $N(s)$, $N(s)$ the number of contiguous blocks of alleles in s, and G is the reference function plotted in Fig. 35.

This rule privileges individuals that have a big alternation of ones and zeros: The more there are alternate one-zero blocks (relative to feasible range), the better the fitness is.

Experiments show that the problem is intractable for a HC, but is obvious to solve for a GA. This is due to the initialization procedure: initial population is sampled around $d = l/2$, and already contains high fit individuals. Hence, the GA starts almost on the upper path around $d = l/2$, then following this path, converges within 100 generations.

GA behavior (trajectory and convergence speed) is much like a OneMax one (experiments at different problem sizes confirm this fact), and the right part of the space is never visited.

Function g'_2

$$r(s) = \frac{(l - N(s))}{|I(s)|}\ |G^{\ -1}(d)|,$$

with the same notations as with g'_1.
This rule rewards individuals at the opposite end of the rule $rule_1$: High fit candidates have a small number of blocks. The fewer blocks (relatively to feasible range), the better the fitness is.

GA behavior is completely different, and depends on the number of periods. With 2 periods, all GA runs converge to optimum within 1000 generations. But, as the number of periods increases, a high proportion of runs fail to find the optimum.

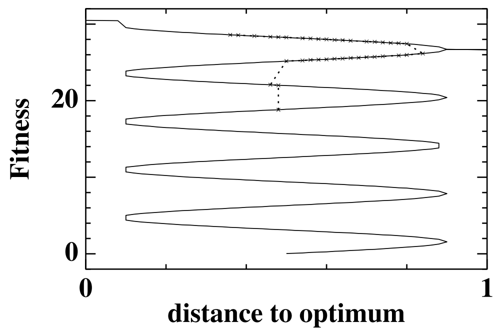


FIG 36 : *Number of one and zero blocks in feasible range* $({}'_2)$*, with 5 periods: most GA runs converge to some local optima. And increasing the population size and stopping criterion, don't help convergence.*

With 5 periods, no run out of 50 could find the optimum: The initialization procedure yields only low fit individuals. The search often goes towards s^0, following some gentle slope inside the plot, as shown in Fig. 36. Then, it reaches the upper path at some high distance from optimum, and stops some where on this path.

II.4.6 DISCUSSION

We presented a case study of GA behavior on a number of functions whose correlation plot shows no local optimum in the fitness–distance space. The overall conclusion is that convergence quality varies from 1 to 4 according to the strings distributions (invisible) in the plot.

The fact is that two neighboring points in the correlation plot are not necessarily close in Hamming distance to each other: The extreme case of function f_1' shows how a locally linear relation between fitness and distance to optimum (the upper path) can hide an intractable local optimum in the Hamming space. On the other hand correlation plots that cover the entire fitness–distance space, may contain easy paths to the optimum, eventually resulting in a OneMax like behavior (g_1').

However, the path that is chosen by the GA is heavily dependent on the position of the initial population in the correlation plot. For instance, a small translation of the fitness–distance relation along the distance axis completely changes the dynamic path of f_1' for example.
Hence, the uniform bitstring initialization is sometimes responsible for the failure of the GA (it causes the GA to take the difficult right path in the case of f_1', and concentrate the search within strings having an alternate number of blocks of ones and zeros in the case of g_2').

This shows how the commonly used uniform initialization can present serious drawbacks. The next Section proposes alternative initialization procedures and gives some guidelines for choosing, in each problem case, the adequate initialization.

II.5 Initialization and Fitness–Distance Relations

Overview Last section shows that interpreting the FD plot in order to predict GA behavior can be very misleading, as various types of functions (easy or difficult for GAs) can share the same FD relations. This section shows that if we consider a fixed fitness function, the *fitness* $\times$ *distance to optimum* plot may be useful to choose problem dependent initialization procedures:
In binary Genetic Algorithms, the standard initialization procedure, with a 0/1 equi-probable choice for every bit, is rarely discussed, as the resulting distribution probability over the whole bitstring space is uniform. However, uniformity is relative to a point of view on the search space. First, considering the measure given by the density of 1's, the *Uniform Covering* initialization procedure is naturally designed. Second, taking into account the probability of the appearance of sequences of identical bits leads to designing another alternative initialization procedure, the *Homogeneous Block* procedure.

These procedures are compared with the standard initialization procedure on several artificial problems. First, *a priori* comparison is achieved using Fitness–Distance Correlation. Further, actual experiments demonstrate the accuracy of these FDC-based comparisons, and emphasize the usefulness of the two proposed procedures as an alternative to the standard initialization.

II.5.1 INTRODUCTION

Initialization is recognized to be a crucial issue in all Evolutionary Algorithms (GAs) in general, and in Genetic Algorithms (GAs) in particular. Even if all theoretical convergence results on Evolutionary Algorithms are proved whatever the initial population (see e.g. [Rud97b, Cer94]), all GA practitioners have experienced that a bad initialization procedure can, in the best case modify the online performance (i.e. increase the time-to-solution), and, in the worst case, prevent convergence toward the global optimum.
Very few works actually address this critical issue. In particular, the bitstring-uniform standard initialization procedure of GAs is hardly discussed, due to its optimality with respect to the criteria of both uniformity over the search space, and genetic diversity.
The situation can be quite different for some real world problems. For instance in the context of inverse problems in Structural Mechanics, where each individual represents a partition of a 2-dimensional domain into two different materials (see. chapter III), the solution can be known to contain far more 0's than 1's (small inclusions of one material into another), and is likely to contain contiguous blocks of 0's and 1's. In that specific case, the standard uniform sampling procedure can be improved by incorporating this domain knowledge into the initialization procedure (chapter III section III.3).
We argue that such alternative criteria can be useful in the general binary framework, to measure the uniformity and diversity of a random population, leading to define alternative initialization procedures. The *Uniform Covering* procedure is based on considerations on the density of 1's and 0's in the bitstring. The *Homogeneous Block* procedure is designed to increase the probability of the occurrence of sequences of identical bits in the bitstrings.
An important issue is then the choice of an initialization procedure for a given problem. The proposed solution is based on the Fitness Distance Correlation (FDC) introduced in [JF95]. This framework is extended to the (more realistic) case where the global optimum of the problem is unknown. This allows us to define an a priori estimate of the validity of any random initialization procedure for a given problem. The accuracy of this estimate is studied both from the heuristical point of view, and experimentally on standard problems from the GA literature.

This section is organized as follows. First, some related works are reviewed and discussed in Section II.5.2. Section II.5.3 then discusses some drawbacks of the standard bitstring-uniform initialization; it introduces two other randomness criteria and the corresponding initialization procedures. Section II.5.4 extends FDC framework to problems for which the global solution is unknown. Section II.5.5 uses the extended FDC to study the impact of the different initialization procedures on a number of problems of GA literature, and estimate their a priori validity. These estimates are confronted with the actual results of intensive GAs runs, demonstrating both the interest of the alternative initialization procedures proposed, and the reliability of the FDC predictions for the choice of initialization procedures.

II.5.2 RELATED WORK

Early work [Gre87] demonstrates the potential strength of incorporating domain specific knowledge into GAs. This can be achieved by seeding the initial population with fit individuals, or with individuals lying in regions of the search space known to be of some interest. Such ideas have been later implemented [SG90]. These ideas however heavily rely on the user, and its expertise in the problem domain.

In [dG90], an iterative scheme is proposed, involving a sequence of fitness functions. A randomly initialized population is evolved by a GA using the first fitness, the resulting population is used as the initial population for a GA using the second fitness, and so on. [SX93] used that idea for constrained optimization problems. In these works, the idea is to purposely bias the initial population of the GA, but here the tool is another GA. However, this approach is highly problem specific, and requires the design of the appropriate sequence of fitness functions, from the user.

A more general approach to improve the initialization procedure of a GA is that of [Bra91]: the initial population is built by taking the best of N randomly chosen individuals. Another general approach is that of [MDG$^+$95], whose re-start operator is called to re-initialize the population whenever premature convergence is detected. These latter ideas, however, do not discuss which random initialization procedure is used either during the initial deterministic selection or for the re-initialization.

A more recent work addressing precisely the initialization procedure is [Iba96], where a new initialization procedure is proposed in the context of Genetic Programming, enhancing the diversity of the initial population regarding some specific criteria. We also refer to [FM96, MF97] where an operational research heuristic is used to build the initial population of the TSP problem, and to [SS98] where a special initialization procedure is designed to achieve mesh numbering by evolutionary algorithms.

The work presented here goes along the same line: the initialization procedure of bitstring GAs has hardly been discussed, because it is optimal with respect to the usual criterion of uniformity in the search space. However, several degrees of randomness have been defined (e.g. in [Knu81], Chapter 3.5: *What is a random sequence?*), among which uniformity is not the sole criterion. Next Sections will observe uniformity and diversity from points of view that differ from the usual bit-wise consideration.

II.5.3 BITSTRING INITIALIZATION PROCEDURES

This section first discusses the "uniformity" of the standard random procedure generally used during the initialization phase of Genetic Algorithms: Other points of view on uniformity show some drawbacks of that standard procedure, and lead to designing other initialization procedures.

The Bitstring-Uniform procedure: Almost all GAs use the *Bitstring-Uniform* procedure (BU) as initialization procedure. It assigns value 0 or 1 with probability 0.5 to every bit. The argument favoring that procedure is its uniformity in the binary space: using the BU procedure, all points of $\{0,1\}^l$ have equal probability $1/2^l$ being drawn. Moreover, the *bit-wise* diversity is maximal too: at every bit position, there will be, in average, as many 0's as 1's. Consider now the point of view of the number of 1's in each bitstring. Denote $\sharp 1(b)$ the number of 1's in bitstring b. The probability distribution of the sum of l independent random boolean variables having probability p to draw 1 follows the well-known binomial law:

$$P[\sharp 1 = k] = C_l^k p^k (1-p)^{l-k}$$

For large values of n (e.g. $l \geq 30$), the binomial law can be approximated by a Gaussian law of mean lp and variance $lp(1-p)$. Hence for the Bitstring-Uniform initialization procedure ($p = 0.5$), $\sharp 1$ lies in a very restricted range: For instance, for $l = 900$, 99% of all bitstrings generated by the standard Uniform procedure have a number of 1's in $[411, 489]$. Hence, from the point of view of the number of 1's in the bitstrings, the BU initialization procedure is certainly not uniform.

Uniform Covering procedure: To address the issue raised above, the *Uniform Covering* initialization procedure (UC) is designed as follows. For each bitstring, select a density of 1s uniformly in $[0,1]$, then choose each bit to be 1 with that specific probability. More formally, the procedure is given as:

```
for each bitstring do
  randomly choose P, uniformly in [0,1]
  for each allele position do
    randomly set bit to 1 with probability P
  done
done
```

The UC procedure of course leads to a uniform probability distribution of $\sharp 1$. But, on the other hand, the probability for a bit at a given position l in a random bitstring to be 1 is 0.5, and the expectation of the total number of 1's in the population is exactly 1/2 of the total number of bits: the UC procedure still achieves maximal bit-wise diversity.
But let us now consider another alternative point of view, considering the size of sequences of identical bits that are likely to occur in a bitstring.
When using the Bitstring Uniform procedure, sequences of k 1's happen at a given locus with probability[54] $1/2^k$. With the Uniform Covering procedure, this probability is $1/(k+1)(= \int_0^1 r^k dr)$.
Depending on the *a priori* regularity of the solution (in term of sequences of identical bits), it might be useful to further bias the Uniform Covering procedure to favor the emergence of homogeneous sequences.

Homogeneous Block procedure: An homogeneous block of bits is characterized by its length n and the position of its center c: "adding" the block of 1's of characteristics

[54] This is **not** the probability that a given bitstring contains a sequence of k 1's. The expectation of that probability must be computed, e.g. by recurrence on the bitstring length.

(c, n) to a bitstring $(b_i)_{i \in [1,l]}$ amounts to set to 1 all bits b_j with $max(1, c - n/2) \leq j < min(l, c + (n+1)/2)$.
The basic idea of the *Homogeneous Block* initialization procedure (HB) is to start from a bitstring initialized to a default value (e.g. 0), and to gradually "add" homogeneous blocks of the other value (1). The critical issue then becomes the number of such blocks, and the way their characteristics are randomly drawn.
In the same vein as for the UC procedure, the number of blocks is adjusted depending on a uniformly drawn target density of 1's: Blocks are added to an initial bitstring of all 1's, until that target density value is reached. (the last block is retained or rejected if the density of 1's of the bitstring after addition of this block, is closer to the target density than before). The block characteristics are chosen as follows: A local length L_{loc} is randomly chosen, uniformly in $[0, L_{max}]$, where L_{max} is user-defined (typically $l/10$). The center c and the length n of each block are chosen uniformly, respectively in $[0, l]$ and $[0, L_{loc}]$. Finally, to avoid loops that are too long in the case of high target density and small local length L_{loc}, a maximum number of blocks is authorized (typically l). The overall procedure can be sketched more formally as

```
for each bitstring do
  set all bits to 0
  randomly choose T, uniformly in [0,l]
  randomly select Lloc, uniformly in [0,Lmax]
  set #1 to 0
  until #1 > T do
    randomly choose c, uniformly in [1,l]
    randomly choose n, uniformly in [1,Lloc]
    add block (c,n) to bitstring
    increase #1 accordingly
    if number of blocks > l
       stop
  done
  if (#1-T) > (T-#1_if_last_block_removed)
    remove last block
done
```

Due to the tight control over the density of 1's, the bit-wise diversity should be close to optimal. However, because of the limit on the number of blocks, that procedure is slightly biased toward bitstrings having more 0's than 1's. A way to counter-balance that bias is to draw half of the bitstrings using 0 as the default value, and the other half by reversing 0 and 1.

Comparing initialization procedures: The most simple and common way to compare some parameters settings of Evolutionary Algorithms on a given problem is to run numerous experiments for each one of the settings, and compare averages of on-line or off-line fitnesses. This experimental method is indeed rather time consuming. However, the lack of efficient and robust indicators of difficulty and/or adequation of parameters for a given problem has made this experimental comparative method very popular since the early days of GAs (see e.g. [SCED89]).

In addition to such *a posteriori* comparisons, some studies have tried to propose heuristical approaches to the *a priori* estimation of problem difficulty, as is widely discussed in the chapter's introduction. Sections II.3 and II.4 discuss the reliability of some of these *a priori* estimations of problem difficulty, and give further hints for the limitations of any summary statistic of the fitness landscape to predict problem difficulty.

Nevertheless, the potential usefulness of some of these heuristical methods to compare the parameters settings of Evolutionary Algorithms on the same problem, has not been investigated yet. Among these, the Fitness Distance Correlation [JF95] seems suited to comparing initialization procedures: it is based on a random sample of individuals built using some initialization procedure.

Remark: Note that the goal here is not to compare landscape difficulty for a given GA, nor to take into account parameters biases, as with the *Hamming fitness* basis comparison method proposed in section II.2 (it shows for instance that the Royal Road landscape is easier with the UC procedure which enhances the emergence of its building blocks).
Rather, we only test the adequation of GA parameters setting with the problem at hand. Hence, a biased initialization is a better initialization. This matches the GA practicers' point of view, where the goal is to obtain a good solution more or less quickly.

II.5.4 FITNESS DISTANCE CORRELATION

This section extends the definition of fitness–distance correlation (FDC) studied in section II.3 (see Eq. 124), to the case where the global optimum is unknown, and finally assesses FDC as a tool to compare initialization procedures.

Local FDC

One of the major limitations of FDC for real problems is that its computation requires the knowledge of the global optimum (i.e. the solution of the problem at hand). Though FDC based on global optimum allowed a posteriori to understand previous results on well-known test problems of binary GA literature (for example, Gray vs binary coding for real numbers [JF95]), it is of poor use when facing a new problem whose solution is actually unknown.
However, it is possible to compute the correlation coefficient between a sample and the best individual *in that sample*. Such a coefficient, termed τ_{loc}, is computed as in Equation 124, with the difference that all distances are now distances to the "local" optimum (the optimum in the sample). Of course, this coefficient will probably depend on the choice of that local optimum, and its stability with respect to different samplings should be checked carefully.
Nevertheless, both this stability and, when possible, the comparison of the values (or the average value) of τ_{loc} with different samplings can give some hints about the fitness landscape: they indicate some homogeneity of the fitness landscape – at least in the area sampled by the initialization procedure. And this homogeneity in turn strengthens a priori the predictions that could be drawn from FDC results. On the other hand, nothing can be deduced for uniformly bad stability results for local FDC: it might be that the fitness landscape does not present any regularity in ruggedness.

FDC for comparing initialization procedures

One major limitation of FDC analyzes is that it does not take directly into account the Evolutionary Algorithm used thereafter. Moreover, section II.3 shows that a single correlation coefficient is completely unrelated to problem difficulty.

However, FDC plots (of random samples) might be a good tool for comparing different initialization procedures for the **same** problem. Initialization is the heart of FDC analyzes, and hence have a direct influence on FDC results. Hence, FDC-plots may give useful information for comparing initialization procedures. For instance, diversity is given by the shape and position of the samples in the fitness $\times$ distance-to-optimum space. Such scatter plots of (fitness,distance) points will be termed *FDC plots* in the following.

II.5.5 EXPERIMENTAL VALIDATION

This section experimentally demonstrates that FDC plots can indeed be used to choose between the three initialization procedures presented in section II.5.3: on different test functions, the FDC plots based on the three procedures will be discussed and compared. Moreover, results of actual runs of the same GA, starting from population drawn using these procedures, will validate the FDC-based approach.

Experimental conditions:
All problems below – except the Long Path problem – are set for bitstrings of length 900 (a more detailed description of the problems is given in appendix A).

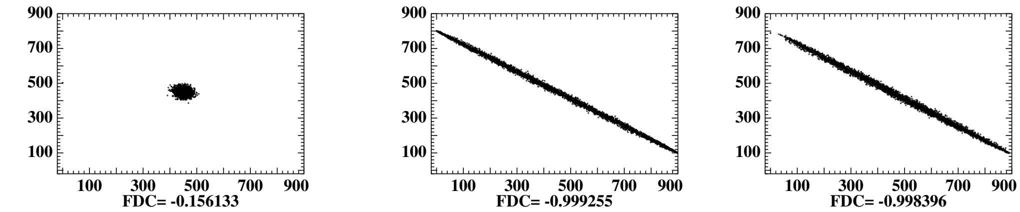
The FDC coefficients and plots are computed on samples of 6000 individuals (see section III.6 for an adaptive adjustment of the sample size in the case of continuous variables).

In all that follows, the Evolutionary Algorithm used to both validate the alternative initialization procedures and check the validity of FDC analysis is based on standards: population size 100, linear ranked-based selection, 1-point crossover with probability 0.6, mutation rate with probability of $1/l$ per bit, elitist generational replacement. The only exception is the Gray-coded Baluja function (section II.5.5) that uses a (10,100)-ES scheme (10 parents give birth to 100 offspring, among which the best 10 become the parents of next generation), 1-point crossover with probability 0.2, and mutation rate with probability of $4/l$ per bit. No tentative to tune these parameters was performed, as the objective was to compare the initialization procedures, not the absolute GA results *per se*.

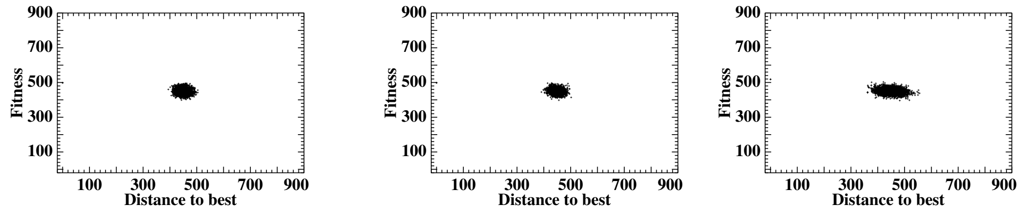
OneMax problems: In the well known *OneMax* problem, the fitness function is simply the number of 1's in the bitstring. Hence, the fitness is equal to l minus the distance to the global optimum: the FDC coefficient is -1, and the – global – FDC plot is a straight line [JF95]. As might be expected, both the Uniform Covering and the Homogeneous Block procedures find bitstrings close to the global optimum in the initial population, and hence achieve better convergence.

In order to avoid the bias favoring the two latter procedures, modified versions of the OneMax problem were designed: The fitness function is defined as the Hamming distance to a fixed bitstring. That bitstring is chosen to have a certain number O of 1's, randomly placed in the bitstring. Such problem is termed the *(O,n-O)-OneMax* problem. Fig. 37 presents the Local-FDC plots for the $(800, 100)$ and $(450, 450)$ problems, for BU, UC and HB initialization procedures. The standard deviation for 20 different local maxima is given below each plot, witnessing that the stability criterion discussed in section II.5.4 is fulfilled.

For the $(800, 100)$ problem, there are still large differences between the different procedures: Whereas the BU procedure (Fig. 37-a) shows a round-shaped cloud with FDC coefficient close to 0, both other procedures (Fig. 37-b and -c) resemble the ideal OneMax global FDC

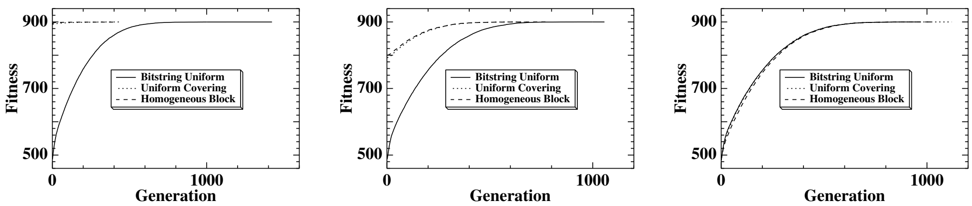


(800, 100) : (a) BU - $\sigma = 0.01$ (b) UC - $\sigma = 1\,10^{-5}$ (c) HB - $\sigma = 2\,10^{-5}$



(450, 450) : (d) BU - $\sigma = 0.01$ (e) UC - $\sigma = 0.04$ (f) HB - $\sigma = 0.09$

FIG 37 : *FDC plots for two modified OneMax problems and the three initialization procedures.*



(a) $(900-0)$-OneMax (b) $(800, 100)$-OneMax (c) $(450, 450)$-OneMax

FIG 38 : *Average on-line results of GA on three OneMax problems: influence of the initialization procedure.*

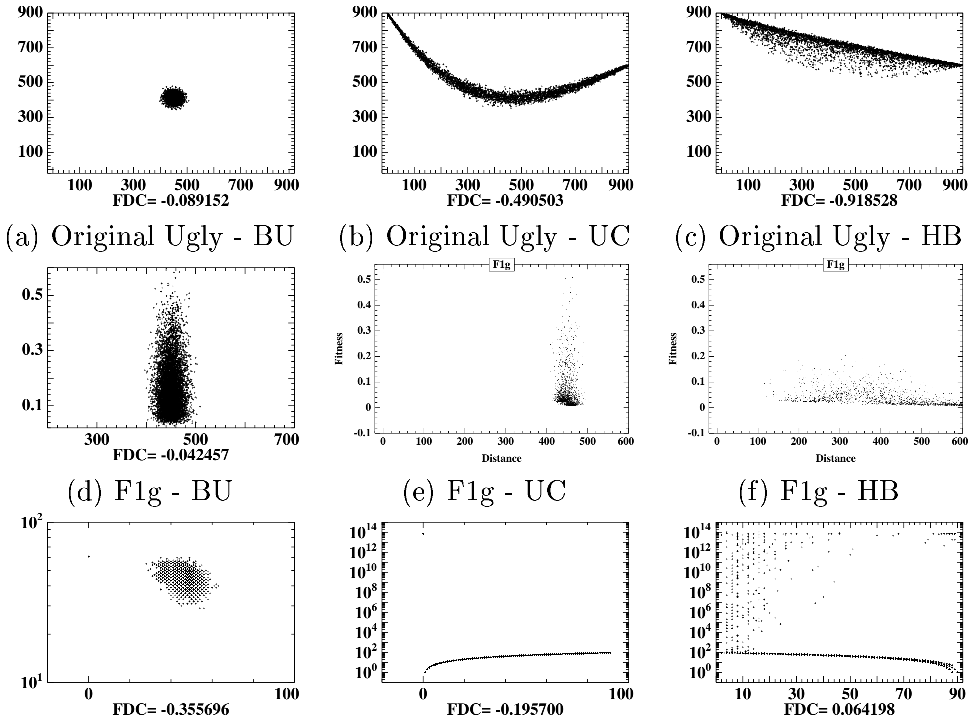
plots. Further, not only the FDC coefficient are close to optimal -1, but also the shapes of the clouds give clear indications. The diversity of the sample is larger for both criteria of fitness and genetic distance for the UC and HB procedures than for the BU procedure. In spite of the optimal bit-wise diversity for all three procedures (see Section II.5.3), the genetic diversity seems quite smaller for the standard BU procedure when the Hamming distance among genotypes is considered.
For the $(450, 450)$ problem, BU and UC (Fig. 37-d and -e) exhibit almost exactly the same plots and coefficients, while the genetic diversity is slightly larger in the HB plot (Fig. 37-f) (the "cloud" is wider on the x-axis), with a slightly better coefficient. However, in all cases, these results suggest that no direction will be preferred by the GA (coefficient close to 0, compact round-shaped clouds).

These a priori considerations are verified when it comes to actual runs of the GA: Fig. 38 shows the plots of the on-line performance of GA for the original and modified OneMax problems. These plots are the usual average best fitness along generations of 21 independent runs. On the $(800, 100)$ problem (Fig. 38-b), the T–test with 99 percent confidence states that the BU plot is different from both others up to generation 900. Lastly, in the extreme case of the $(450, 450)$ problem (Fig. 38-c), all three plots are almost indistinguishable, but at least using the biased procedures UC and HB does not degrade the results.

The Ugly problem: The situation here resembles that of the OneMax problem (the maximum is the all 1's bitstring). However, the FDC plots on the other hand are much more interesting: whereas the BU-plot has again a round regular shape (Fig. 39-a), a clear tendency toward the optimum is shown by the UC-plot (Fig. 39-b), and even more by the HB-plot (Fig. 39-c). Moreover, both latter plots show a higher range of both fitnesses and distances.
For the original Ugly problem, the average performance of GA runs are very similar to those of the $(900, 0)$ OneMax problem (see Fig. 38-a). Again, a modified version of the Ugly problem was designed: A reference bitstring is randomly generated using the BU procedure. The fitness of an individual is then the Ugly-fitness of its XOR with that reference bitstring. The correlation plots of the three procedures for that modified Ugly problem (not presented here) look very much alike: a round shape with a correlation coefficient close to zero. The average performance of GA runs (Fig. 40-a) reflects that situation.

Gray-coded Baluja function F1: The correlation plots show similar repartitions for the BU and UC procedures. Both of them present a better fitness range than the HB procedure. This means they are likely to start with fitter individuals than HB. Hence, the on-line results (Fig. 40-b) show that BU and UC achieve better performances than HB during approximately the first 100 generations. However, after a longer period of evolution, HB clearly takes the lead (as confirmed by the T–test with 99 percent confidence), and its off-line results are far better. This phenomenon may be due to the correlation difference. The HB procedure, starting from lower fitnesses than BU and UC, has a better correlation coefficient and better-shaped FDC plot (Fig. 39-f): as individuals improve their fitness, they are likely to get closer to the optimum.

The LongPath problem: Due to the wide range of fitness values, and to the small size of the path itself, the FDC plots for the LongPath problem are plotted in Log scale (note also the different scales of the BU plot of Fig. 39-g) For the same reason, the FDC coefficients are rather meaningless, as most points of the sample lie outside the path. A simple look at the

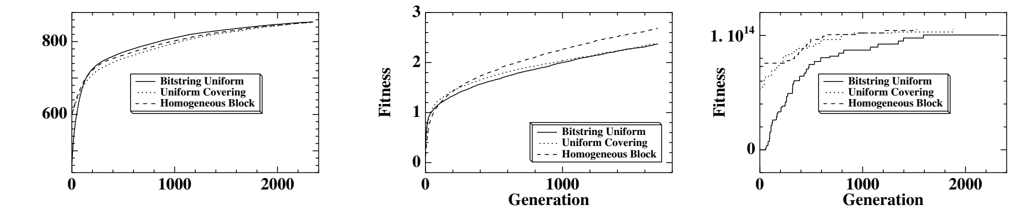


(a) Original Ugly - BU (b) Original Ugly - UC (c) Original Ugly - HB

(d) F1g - BU (e) F1g - UC (f) F1g - HB

(g) LongPath - BU (h) LongPath - UC (i) LongPath - HB

FIG 39 : *FDC plots for the Ugly, F1g and LongPath problems using the three initialization procedures.*



(a) Modified Ugly problem (b) F1g problem (c) LongPath problem

FIG 40 : *Average on-line performances of 21 GA runs on Modified Ugly, F1g and LongPath problems: influence of the initialization procedure.*

FDC plots for procedures UC (Fig. 39-h) and HB (Fig. 39-i) shows that the latter covers a much wider range of fitnesses - while both have the same range of distances. Moreover, on both plots, the initial OneMax-like part of the fitness landscape is clearly visible. Indeed, the on-line performances of both procedures are almost identical, as there seems to be enough diversity to allow to find shortcuts of the path (Fig. 40-c).

II.5.6 CONCLUSION

Two new binary initialization procedures have been introduced. The *Uniform Covering* procedure ensures that the obtained density of 0's and 1's per bitstring is uniform over $[0, 1]$. The *Homogeneous Block* procedure favors sequences of 0's or 1's in the bitstring, somehow betting on some regularity of the target optimal solution. Note that these procedures respect the bit-wise uniformity among the whole population.
Experiments on different problems of the GA literature have demonstrated the potential usefulness of these procedures: either better on-line results, or/and better off-line results can be obtained. Of course, the diversity of the panel of problems used during these tests is questionable, though these problems are usual benchmarks for GAs. However, the modified OneMax and Ugly problems have been used to address this issue: even in the worst case, both alternative procedures provide the same quality of results than the standard procedure.

Anyhow, we do not claim that the proposed procedures should be preferred to the standard procedure. Rather, we state that the alternative procedures should be considered together with the usual Bitstring Uniform procedure before starting the optimization of any binary problem.

Further, we have demonstrated that the measure of the – Local – Fitness–Distance Correlation has good predictive properties for the a priori choice of an initialization procedure, regardless of its accuracy as a general predictor of problem difficulty. This comparison of performances of different initialization procedures relies on the relative repartitions of the samples in the fitness $\times$ distance space, considered together with the correlation coefficients. Hence, at the cost of a single run of the evolutionary algorithm (a few thousand function evaluations), FDC coefficients and plots can be obtained, allowing us, in most cases, to choose among different available initialization procedures. Note that this conclusion holds for any search space where more than one initialization procedure can be designed.

If the choice of initialization is unanimously recognized as a crucial step in evolutionary algorithms, the choice of genetic is an even more critical step, as proven by theoretical studies for the mutation operator for example (cf. chapter I for more details). Such studies, however, don't allow us for choosing the adequate (efficient) operator for a given problem.

Hence, analogously to the study of this section, we propose to focus on comparing practical tools to choose a genetic operator. The following section is restricted to the case of crossover operators, yet, the same tools and conclusions can be extended to the mutation case (as illustrated in the examples of section III.6).

II.6 Statistical Indicators of Crossover Operators

Overview Having investigated the limitations (II.3) and the benefits (II.5) of the landscape based indicators, we turn to study some operator based indicators proposed in the literature: The choice of an operator in evolutionary computation is generally based on comparative runs of the algorithms. However, some statistical a priori indicators, based on samples of (parents-offspring) fitnesses, have been proposed to replace such brute comparisons. This section surveys some of these works in the context of binary crossover operators. We first extend these indicators to overcome some of their limitations. Unfortunately, experimental results on well-known binary problems suggest that any of the indicators used here can give wrong indications. All being defined as averages, they can miss the important parts: none of the tested indicators have been found to be a good indicator alone. However, detailed insights at the best offspring fitnesses, lead to designing a new tool based on offspring ranking which gives accurate predictions on the respective performances of binary crossover operators.

II.6.1 INTRODUCTION

Some previous studies have proposed heuristical approaches to estimate the efficiency of different operators a priori. These methods are usually based on statistical indicators computed on random samples of points in the search space. Two kinds of such methods can be distinguished.

The first set of methods does not take into account the specific characteristics of operators, but rather implicitly relying on conceptual notions that are common to all operators. For instance, the strong causality of mutations is assumed in [MdWS91], where a distance in the genotype space is used to assess correlation length. In the same line, the respectfulness of operators toward schemata [GR91, RS95a] or epistasis [Dav90] is used to characterize the suitability of the representation and its associated operators.

The second kind of method directly addresses the issue of operator performance relative to the fitness landscape. In the early days of Evolution Strategies, Rechenberg [Rec73] derived the famous 1/5 rule by the theoretical calculation of the expected progress rate toward the optimum for a zero mean Gaussian mutation, on the sphere and corridor models in $\mathbb{R}^l$, in relation to the probability of a successful mutation.

More recently, Weinberger [Wei90] introduced the notion of fitness operator correlation, for asexual operators, as the correlation between the parents' fitness and their iterated offspring. Manderick [MdWS91] adapted this correlation measure to crossover operators. He devoted attention to the correlation between mean parental fitness and mean offspring fitness, and indicated the possible usefulness of this indicator to assess the role of an operator on the NK-landscapes and the TSP problems.

However, as noted by Altenberg [Alt95], such a fitness operator correlation suffers from theoretical limitations: a maximal correlation is possible even in the absence of any improvement of fitness from parents to offspring. Altenberg further suggested that the performance of a genetic algorithm is related to the probability distribution of offspring fitness indexed by their parents' fitness that result from the application of the genetic operator during the GA. Unfortunately, Altenberg noted that empirical estimation of the fitness probability distribution could be problematic if the searched surface has multiple domains of attraction. Grefenstette [Gre95a] used the probability distribution of offspring fitnesses indexed by the mean parents fitnesses to yield a first-order linear approximation of the genetic operator's suitability. Another approach was suggested by Fogel and Ghozeil [FG96]: to compare different selection schemes and operators, they use the mean improvement of offspring, and conclude that the uniform intermediate recombinations yields better results (higher mean improvement) than the one-point crossover, for all three selection methods they tested, on some two-dimensional functions in the real-valued framework.

This section is concerned with statistical a priori indicators that can be used to assess crossover efficiency: All the above indicators (fitness operator correlation, probability distributions, and mean improvement) have been tested separately. As far as we know, no comparison has been made of their efficiency on the same problem. The goal here is to achieve such a comparison on different binary problems from the GA literature. The different selection schemes will not be discussed here, and for all experiments, linear ranked-based selection will be used. These experiments are limited to the use of a single crossover operator, although the same techniques could be used for any combination of operators. For instance, the experiments presented in [Spe95] suggest that an important factor can also be the interactions between two different operators .

The section is organized as follows. Section II.6.2 presents in detail and discusses the statistical indicators briefly introduced above. A modified version of the fitness operator correlation is introduced to address some of its limitations. The mean improvement method is modified to only account for strictly positive improvements. Section II.6.3 experimentally compares the estimates of those different indicators on various test cases in light of actual results of GA runs, demonstrating the inadequacy of the fitness operator correlation for most problems, and the potentialities of the statistical indicators based on best offspring fitnesses. These results yield a better understanding of the validity domain and the limitations of each statistical indicator. Moreover, a detailed look at some samples calls for on-line measures. Further research directions are sketched in Section II.6.4, in light of the obtained results. A new method, based on offspring ranking, shows reliable to compare operator efficiency at early GA generations.

II.6.2 COMPARING CROSSOVER OPERATORS

This section details some of the statistical indicators that have been proposed in previous work, and discusses some of their limitations. This leads us to propose alternative indicators.

Consider the problem of optimizing a real-valued fitness function f defined on a metric search space E and suppose that there exists one global optimum of f. The common basis of all statistical indicators described below is the use of a random sample of individuals in E. The operators to be assessed are then applied on all (or part of) the sample.

Mean Fitness Operator Correlation

Manderick & al. [MdWS91] define the Mean Fitness Operator Correlation (M-FOC) as follows:

Consider a sample of N couples of individuals $(x_i,\, y_i)$. Let (f_{pi}) be the mean fitness of couple $(x_i,\, y_i)$, and (f_{ci}) the mean fitness of their two offspring with crossover OP. The *M-FOC coefficient* of the operator OP is defined by:

$$\frac{\frac{1}{N}\sum_{i=1}^{N}(f_{pi}-\bar{f}_p)(f_{ci}-\bar{f}_c)}{\sigma_{F_p}\sigma_{F_d}}, \tag{163}$$

where $\bar{f}_p, \bar{f}_c, \sigma_{F_p}$ and σ_{F_d} are respectively the means and standard deviations of parents' and

offspring mean fitness.
This coefficient measures the extent to which the mean fitness of a couple of parents can give some information about the mean fitness of their offspring. A very high correlation between parents and offspring (M-FOC coefficient close to 1) indicates that offspring mean fitness would be almost proportional to parental mean fitness. No correlation (M-FOC coefficient close to 0) shows that no information about offspring mean fitness may be gained from the parents mean fitness. GAs using high-correlated operators are expected to achieve best performances as demonstrated in [MdWS91] on the NK-landscapes and the TSP problems.
Unfortunately, suppose there is a quasi-linear relationship between the mean fitnesses of both sets of parents and offspring. The M-FOC correlation will be maximum (1), whatever the slope of that linear relation. In particular, if that slope is less than 1, no offspring fitness can be better (in mean) than its parents': the correlation can be maximal when no improvement is to be expected! However unrealistic this particular case can be, it nevertheless shows the limits of M-FOC as a measure of operator's efficiency.
In that line, as noted by Fogel and Ghozeil [FG96], M-FOC cannot be used on problems that yield a zero mean difference between parents and offspring fitnesses, as in the case of linear fitness functions with real-valued representations. The correlation is always maximal, and cannot reflect the high dependency of their convergence rate on the the standard deviation of the Gaussian mutation [Rec73].

Best Fitness Operator Correlation
To address the above-mentioned limitations of M-FOC, we propose to study the correlation between the best (rather than mean) parent and offspring fitnesses. The definition of the B-FOC coefficient is the same as in equation 163, except that $\bar{f}_p, \bar{f}_c, \sigma_{F_p}$ and σ_{F_d} are replaced by the means and standard deviations of the best fitnesses of parents and offspring. This coefficient should avoid some of the ambiguities pointed in the preceding paragraph.

Mean Improvement
Fogel and Ghozeil [FG96] use 1000 couples of parents to compare different selection schemes and operators. To generate each couple, 100 random individuals are drawn. The couple is then chosen among them randomly, either as the two best individuals, or as one random and the best. The mean improvement of the best offspring (of each couple) over the best individual of the initial 100 individuals is then computed. Such mean improvements are finally averaged over the 1000 couples. As noted in [FG96], one limitation of this approach is its computational cost (102.000 fitness evaluations). The following definitions are a tentative to achieve the same goals at lower cost: Consider a sample of N random individuals as $N/100$ disjoint subsets of individuals. Among each of these subsets of size 100, 100 couples of parents are selected by ranking selection. Two cases are considered:

- **Mean Improvement over the best parent** In the same line as Fogel and Ghozeil's second procedure (computing the improvement over the best parent), we propose to compute the improvement of the best offspring of each couple over its best parent. The mean improvement of N couples of parents is computed as follows:
$$\text{MI-P} = \frac{1}{N}\sum_{i=1}^{N} \max(0, \mathcal{F}(\text{best offspring}) - \mathcal{F}(\text{best parent}))$$

- **Mean Improvement to target** Fogel and Ghozeil's first procedure amounts to computing improvement of the offspring over a target set to the best fitness of the current set of 100 individuals. Rather, we define a target $\mathcal{T}$ as the median of the set $\{f_i\}_{i=1,\frac{N}{100}}$ where f_i is the fitness of the best individual among those of the ith -100 size- subset generated. The mean improvement to target is then computed as follows:

$$\text{MI-T}= \frac{1}{N}\sum_{i=1}^{N}\begin{cases} \max(0, \mathcal{F}(\text{best offspring}) - \mathcal{T}) & \text{if } \mathcal{F}(\text{best offspring}) > \mathcal{F}(\text{best parent}) \\ 0 \text{ otherwise} \end{cases}$$

Probability of Improvement
The above definitions of MI-P and MI-T do not consider the number of offspring that are actually a source of improvement. In order to separate the mean improvement from the probability of improvement, the latter is defined as the fraction of offspring actually having a better fitness than their parents (PI-P) or the target (PI-T). By counting in the definitions of MI-P and MI-T, only those offspring, two new indicators are then defined, termed MI-P+ and MI-T+, .

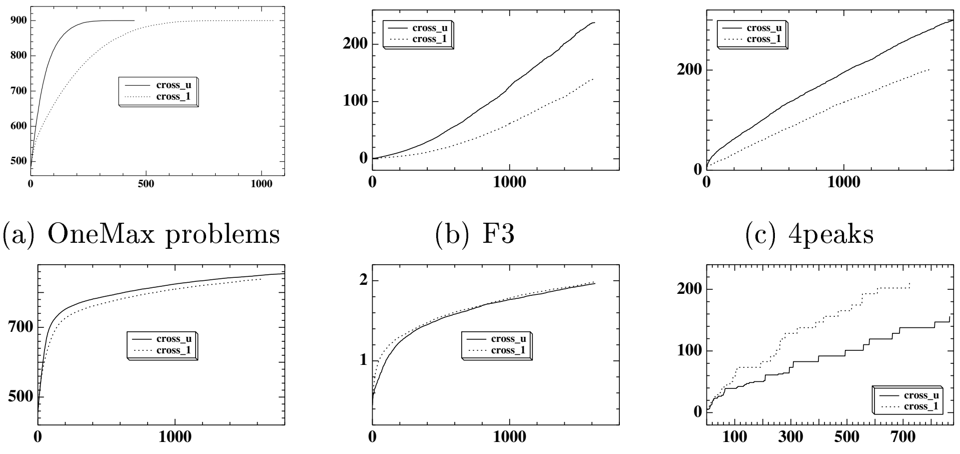
II.6.3 EXPERIMENTAL VALIDATION

From now on, the size of the binary search space will be $l = 900$ for all problems (except the Royal Road), and the underlying GA some standard binary GA: population size 100, linear ranked-based selection, crossover rate of 0.6, $1/l$ mutation, and elitist generational replacement.
Two operators are compared: The uniform crossover (UC) and the one-point crossover (1C). For each of them, the statistical indicators are computed either on samples of 10000 individuals, with one crossover per parent, or on samples of 200 individuals, each couple generating 100 offspring (by repeating 50 crossovers operations). The parents are selected with a linear ranking selection applied to each disjoint subset of 100 individuals. In both cases, the computational cost is 20000 fitness evaluations per operator.
The goal of these experiments is to compare operators under the same conditions of initialization and selection. Therefore neither the selection scheme (as in [FG96]) nor the initialization procedure (as in section II.5) will be modified.
The stability of all the following statistical indicators have been checked carefully. The relative difference between indicators values on UC and 1C, has been found stable over 10 trials in most cases. However, all unstability cases are marked with a "*" in the tables.

II.6.3.1 Results

For each problem considered below, the statistical indicators defined in subsection II.6.2 have been computed, and their predictions compared to actual GA runs.
Average on-line results of the runs (over 21 runs) are presented in Fig. 41, and the statistical indicators for both uniform (UC) and one-point (1C) crossovers are presented in Tables II.6.3.1 and II.6.3.1. Each table presents FOC and MI-P coefficients with one crossover per parent, as well as MI-T coefficients for both 1 and 50 crossovers per parent. Three (modified) OneMax problems (defined in II.5.5) have been tested. Only one of them is presented since they gave very similar results.

Stability Large unstability has been found for MI-T with the Royal Road problem. This is due to the highly discontinuous fitness values it yields (observed values are limited to 1, 5.2, 5.4, 21.6). Hence, when computing MI-T on few individuals, this can amount to contradictory results for different samples. More generally, MI-T has been found slightly unstable. This is not surprising when considering that very low probabilities are involved (lines PI-T in the tables). However, when considering the case of 50 crossovers per parent, the unstability no longer alters the significance of the differences between MI-T coefficients



(a) OneMax problems (b) F3 (c) 4peaks

(d) Ugly (e) F1g (f) Royal Road

FIG 41 : *Average on-line results of 21 GA runs on different problems (fitness ×generation): influence of the crossover operators.*

of operators. A last general remark is that all MI coefficients should be considered relative to the actual fitness range in the sample.

FOC Coefficients The predictions of M-FOC indicator were inaccurate for all test problems, with the exception of the F1g problem. On the 4peaks, Ugly and F3g problems, better on-line performances (Fig. 41) are obtained for the operator with lower M-FOC values (Tables II.6.3.1 and II.6.3.1). According to T-test with 99% reliability, runs for the Royal Road problem with either UC or 1C are similar until generation 500, though, their respective M-FOC coefficients are quite different (Table II.6.3.1).
On the same problems, B-FOC give similar indications than M-FOC, and hence suffer the same defects. On the OneMax problems, where the M-FOC always equals 1, the B-FOC coefficient reaches higher values for the 1C, while the runs (Fig. 41) using UC are far better than those using 1C.
Hence neither M-FOC nor B-FOC seems an accurate predictor of relative performance of operators.

Mean Improvement to Parents If we consider the F1g problem, the results of the runs (Fig. 41), show that the one-point crossover average run is better than the uniform one until generation 200 (confirmed with a 99 % confidence T-test). But mean improvement to parents values (Table II.6.3.1), suggest the opposite. A close look at the plot of best parents × offspring (Fig. 42), shows that UC yields better improvement than 1C only at low fitness values of (best) parents. The situation is reversed for higher fitnesses. This situation could have been predicted by the Best FOC results: a 0.7 correlation for the 1C means that the best offspring's fitness is almost proportional to its best parent's fitness, while the 0.2 value for the UC means on the other hand that well-fitted offspring can be obtained whatever the (best) parent's fitness. The misleading results of improvement to parents are so no longer surprising. It seems that MI-P values can be compared reliably when the B-FOC coefficients are similar, as is the case with the F3g and the three OneMax problems.
For the Ugly and Royal Road problems, the situation is similar to that of the F1g problem.

	OneMax problems		F3g		4peaks	
	UC	1C	UC	1C	UC	1C
M-FOC	1	1	0.88	0.99	0.7	0.93
B-FOC	0.7	0.8	0.6	0.8	0.6	0.99
MI-P	3.4(0.03)	2.7(0.03)	0.01	0.008	0.4	0.002
MI-P+	7(0.08)	6(0.06)	0.02	0.016	2	2
PI-P	0.23	0.23	0.26	0.25	0.1	0.003
MI-T	0.07(0.012)	0.06(0.016)	0.00026	0.00021	0.1	0.002
MI-T+	5.6(0.3)	5.6(0.5)	0.02	0.02	2.2	1.8
PI-T	0.006	0.005	0.006	0.005	0.005	6e-5
MI-T (50)	**3.2**(1)	**1.2**(0.7)	**0.01**	**0.004**	**0.6**	**0.02**
MI-T+ (50)	**8**(1)	**6**(1)	**0.028**	**0.025**	**2.7**	**1.7**
PI-T (50)	**0.02**	**0.018**	**0.018**	**0.016**	**0.01**	**0.0001**

Table 5: *Mean(standard deviation) of statistical a priori indicators for different problems, on a 10000 individuals sample, 2 offspring for each parent with the uniform and one-point crossover operators. Bold font correspond to a 200 individuals sample, and 50 offspring per parent. The + concerns strictly positive improvements only.*

	Ugly		F1g		royal	
	UC	1C	UC	1C	UC	1C
M-FOC	0.7	0.99	0.2	0.7	0.18	0.92
B-FOC	0.5	0.8	0.1(0.01)	0.6(0.005)	0.18	0.91
MI-P	6(0.07)	3.6(0.05)	0.05(7e-4)	0.03(4e-4)	0.19	0.019
MI-P+	12(0.1)	7.7(0.1)	0.09(9e-4)	0.05(8e-3)	4.2	4
PI-P	0.27	0.23	0.3	0.28	0.02	0.002
MI-T	0.08(0.01)	0.07(0.01)	7e-4(1e-4)	5e-4(1e-4)	e-3(8e-4)	2e-4(4e-4)
MI-T+	6.6(0.5)	6.8(0.5)	0.05(2e-3)	0.05(5e-3)	1.5(1.2)	0.9(2.3)
PI-T	0.006	0.005	0.007	0.006	4e-4	1e-4
MI-T(50)	**5**(2)	**1.3**(0.8)	**0.05**(0.02)	**0.02**(7e-3)	**0.07**(0.07)	**0.01**(0.02)
MI-T+(50)	**10**(2)	**8**(1.5)	**0.07**(0.01)	**0.06**(0.01)	**2.5**(2.8)	**0.6**(4)
PI-T(50)	**0.018**	**0.016**	**0.016**	**0.015**	**4e-4**(1e-4)	**4e-4**(8e-4)

Table 6: *Mean(standard deviation) of statistical a priori indicators for different problems, on a 10000 individuals sample, 2 offspring for each parent with the uniform and one-point crossover operators. Bold font correspond to samples of 200 individuals, and 50 offspring for each parent.*

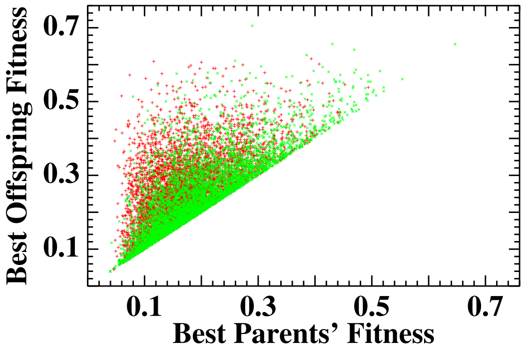
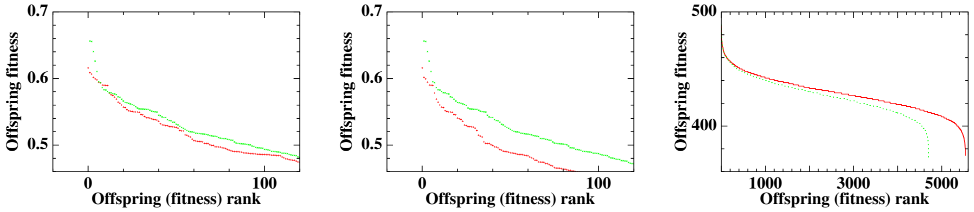


FIG 42 : *Fitness of best offspring × fitness of best parent, for the offspring that outperformed their parents, on the F1g problem.*

Average improvements to parents are very different in favor to UC, whereas the runs show identical performances: the T-test with 99 % confidence finds no difference between average runs until generation 30 (Ugly) and 500 (Royal Road). Again, at the contrary of 1C, the UC yields good fitness offspring from low fitness parents, as illustrated by the B-FOC coefficients. Note however that the reverse is not true: different B-FOC values do not necessarily imply false MI-P previsions, as illustrated by the 4peaks problem.

Considering the probabilities of improvement (PI-P and MI-P+) for the F1g and Ugly problems does not improve the results: the same bias as for MI-P is observed, for the same reasons (different B-FOC coefficients).

Mean Improvement to Target The case of 1 crossover per parent is found to be rather unstable and often involves very low improvement probabilities, which does not help comparison. Note for example that for the OneMax problems (table II.6.3.1), the observed differences of MI-T, with one crossover per parent, are very small (around 0.01) relative to the range of fitness values of the problem ([400,500]). Hence only the case of 50 crossovers per parents (bold font in tables) will be considered in the following.
When considering the improvement to target, the dependency on the parents' fitness vanishes. So no bias is induced by the asymmetric repartitions of improvement to parents. Nevertheless, for F1g and Ugly problems, MI-T values (Table II.6.3.1) are still indicating some superiority of UC, whereas the opposite actually happens during the GA runs for F1g, and Ugly runs are similar until generation 30 (Fig. 41). The main reason for that is probably that averaging the improvements hides both the evolution of their differences and the best fitted offspring obtained: For the F1g problem, when getting to higher parents' fitness, the 1C gets better improvement to target than the UC as can be seen for instance on the plot (best parent × best offspring) of Fig. 42.
Considered together with the probability of improvement, some additional information about the sample repartition can be obtained. In fact, PI-T takes into account the whole set of generated offspring, while MI-T only takes into account the set of best offspring of each parent. For the problematic cases of F1g and Ugly, MI-T+ values are still favoring the wrong operator, but with much smaller differences than MI-T, and PI-T values are close for both operators. MI-T+ still suffers from the bias of averaging, and does nevertheless hide the most important information about offspring, which is their maximal fitnesses, and how they evolve



(a) F1g :whole sample (b) F1g: Half sample (c) Ugly: whole sample

FIG 43 : *Fitnesses of best offspring × their rank, with one crossover per parent, for uniform and one-point crossovers: The relative position of curves changes when only the top half of the sample is considered for offspring generation.*

when higher parents are considered.

II.6.4 FURTHER DIRECTIONS: OFFSPRING RANKING

Other indicators, not presented here, have been computed for both UC and 1C operators; such as mean fitness values of all offspring, or of all offspring that outperformed their parents. None of these have been found of any help for predicting operators suitability. Rather, the most important factor is found to be the 'maximal potential' of an operator (i.e. how fit are the best offspring it can give birth to).
To illustrate this, consider the plots of the fitness repartition of offspring as a function of their rank in the sample. The case of the F1g problem presented in Fig. 43-a, shows the best offspring of the whole sample. One cannot say which curve is better: 1C yields higher fitness offspring than UC, but UC gives a higher number of well-fitted offspring. (Remember that all the considered indicators suggest the superiority of UC, and that the runs are ordered the other way round.) But consider now Fig. 43-b, which plots the same repartition of best offspring, but when considering only the best half of the parents in the sample: The curve for 1C gets largely better than UC.

Figure 43-c presents the same plot for the Ugly problem: curves of 1C and UC are very similar (in agreement with the runs), while all the previous indicators favor the UC. Such *offspring ranking* plot was found efficient in predicting the runs order for all the considered problems, for instance it is almost unchanged when considered with the top half of the sample.

A crucial factor for operator comparisons seems to be their ability to generate high values, even rarely, or even for the best of parents. And this can be quite different from any mean value on the whole sample. The mean as a measure does not give any information about the most likely direction of evolution. This calls for alternative ranking based indicators that put the emphasis on the upper tail of fitness distributions, as well as for on-line dynamic measurement.

II.6.5 SUMMARY AND CONCLUSION

Some statistical indicators of a genetic operator's suitability have been presented and experimented to compare 1-point crossover and uniform crossover in the binary framework.

These experiments have first confirmed the inadequacy of fitness operator correlation, which was either unable to detect differences, or even predicts the inverse of actual results of GA runs. FOC only indicates whether well-fitted offspring are obtained from well-fitted parents or not. However, such information can be useful in addition to other indicators.

Average improvement to parents (MI-P) can be very misleading for two reasons:
1- The dependency on the parents fitnesses can induce big differences (in MI-P) while the fitness distribution of offspring -independently of their parents- is the same for different operators.
2- It suffers from the loss of information induced by any measure based on averaging.
Those remarks address both cases of 1 and 50 crossovers per parents, which generally gave the same tendencies, the case of 50 crossovers, showing much higher differences.

When considering average improvement to target (MI-T), only the case of 50 crossovers per parents gave stable and significant differences relative to the fitness range of each problem.

MI-T no longer suffers from the first bias above, but still meets the second.
MI-T, even with 50 crossovers per parents, fails, on both F1g and Ugly problems, to predict the best crossover. A possible explanation is that it is due to averaging. In fact, on the F1g problem, the relative position of the offspring fitness distribution for UC and 1C operators changes when considering only fit parents. Moreover, the averaging does not capture any information about the fittest offspring of each operator. This information, is found to be crucial for comparing crossovers.

Hence, none of the proposed indicators was found satisfactory alone, though MI-T appears the most useful – especially when B-FOC values are similar. To design more efficient indicators, averaging should probably be avoided, and best fitness offspring should be emphasized. Whatever the indicator, the most important factor seems to be the way it evolves when getting to higher fitness parents. Further work will have to design online measures.

However, the *offspring ranking* method proposed above is only valid for predicting operator ability to rapidly increase the upper tail of population fitnesses. Indeed such strategy might not be efficient on the long term run: maximizing the cumulative on-line increase in fitness is not necessarily the best strategy to optimize a given problem. This calls for further investigation of the question: can we find any hints about the optimization strategy that fits a given problem ?

II.7 Conclusion of Chapter II

This chapter has investigated three types of landscape similarities, first in terms of dynamic EA behavior, second in terms of static landscape indicators (FDC, epistasis), and third in terms of operators indicators (as fitness improvement).

The first part of this chapter (section II.2) addresses the challenging issue of comparing the behaviors of two different algorithms when applied respectively to two different fitness functions. First, a number of *behavioral measures* are defined –independent from fitness distribution– are defined for characterizing EA's behavior (convergence time and geometrical quantities derived from the trajectory of the algorithm). Each *behavioral measure* obtained on a given fitness f, is scaled, using the behavioral measure obtained by running the same EA on the *distance fitness* to solution.

In the second part of the chapter, some statistical indicators of fitness landscapes and evolution operators, both based on random sampling of the landscape, are investigated and extended. The overall conclusion is that the indicators are blinded by a large proportion of completely irrelevant information present in the random sample.

1. We have designed a class of parametrized Site Wise Optimizable fitness functions, for which, landscape based indicators take arbitrary values. However, this situation is not that surprising. The reliability of an indicator is necessarily limited as it is a statistical summary, often defined as an average. Yet, the fact that the indicators examined here are even unable to deal with easy functions (SWO) calls for future design of alternative, more robust, indicators, eventually based on alternative sampling schemes.

2. We have experimentally demonstrated that the expected-fitness-improvement of offspring over their parents (resp. over a target fitness) does not reflect the actual on-line performance of crossover operators when used by EA instances. We have proposed a new indicator, based on the ability of an operator to generate –even rarely– high fit offspring. This indicator seems to better reflect what happens during EA run.

The key issues that emerge from the study of such indicators are:

- The long term EA behavior is not reflected by **typical** properties of landscapes and operators (such as expected fitness improvements), but rather depends on **atypical** properties (such as best fitness).

- The key problem in designing such summary statistics is to find a characterization of a broad class of fitness functions on which the EA behaves similarly. Clearly, the class of monotone fitness functions is too small to satisfy this requirement. Note that finding a good definition of "similar behavior" is also a major problem.

The method proposed in section II.2 for comparing EA's behaviors (section II.2) can be used with benefit to address the last issue, as it can be used to identify functions with similar EA behaviors.
However, this comparison method of EA behaviors relies on the definition of a genotypic distance on the search space. Hence, the choice of this distance may be crucial for the reliability of the results. The following chapter investigates different criteria for the choice of a genotypic distance for degenerate variable length representations. This is also developed in "Further Work" section.

Chapter III

Variable Length Representations

Plan du Chapitre III

L'étude expérimentale du problème inverse d'identification d'inclusions, en mécanique des structures, montre que les représentations par diagrammes de Voronoï ou listes de rectangles sont largement plus performantes que la représentation -plus classique- par matrice binaire dépendant du maillage.
Une approche générale pour construire des distances sur ces nouveaux espaces de representation est proposée, suivie d'une étude approfondie des propriétés de ces distances et de la distance de *Haussdorf*. La dernière partie propose des heuristiques pour choisir une representation et une initialisation *a priori*.

III.2 Un probleme inverse

Dans le cadre de l'élasticité linéaire, le problème consiste à identifier la répartition de deux matériaux donnés à l'intérieur d'une structure. Pour cela, on dispose de données experimentales (ou simulées par un calcul éléments finis): les déplacements en certains points de la surface de la structure, sous l'action d'un ensemble de chargements.

L'identification d'inclusions est un problème théoriquement mal posé, pour lequel les méthodes numériques déterministes se révèlent instables. Les algorithmes évolutionaires sont des méthodes d'optimisation stochastiques d'ordre 0 fournissant un cadre général à la résolution de problèmes inverses.
Cependant, l'application des algorithmes évolutionaires au problème de l'identification d'inclusions doit faire face au problème crucial du choix de la représentation des solutions, c'est-à-dire de l'espace de recherche. Trois représentations sont proposées.

III.3 Représentations pour partitions du plan

Une partition du plan en deux sous ensembles (notés 0 et 1) peut être representée par une liste de formes élémentaires (ex. rectangles) ou un diagramme de Voronoï (avec un label de 0 ou 1 pour chaque site). Dans les deux cas, un point de G, l'espace de recherche, est une liste non ordonnée et de longueur variable d'éléments (ou allèles) d'un ensemble A.

Contrairement au codage classique en matrice binaire, ces nouvelles représentations, permettent de séparer la complexité du codage de la précision des solutions possibles. En plus, elles ciblent un espace de recherche plus approprié. En effet, la solution du problème d'identification d'inclusions, est susceptible de contenir de gros blocs contigus d'un matiériau donné.

III.4 Résultats numériques

Les résultats numériques constituent une avancée importante dans le domaine de l'identification d'inclusions mécaniques: les données expérimentales permettent à l'algorithme génétique d'identifier la répartition des deux matériaux (connus) dans la structure de référence. La robustesse des résultats en présence de bruit est expérimentalement établie, et les mérites respectifs des différentes représentations sont discutés.

Cependant, en l'absence d'une distance génotypique, la visualisation des paysages de fitness ou la simple mesure de divesité d'un échantillon donné restent complètement inaccessibles.

III.5 Distances génotypiques

Etant donné une distance sur $\mathcal{A}$, une approche générale est proposée pour construire des distances dans $\mathcal{G}$, espace des listes (non ordonnées et de longueur variable) d'éléments de $\mathcal{A}$. Ces distances sont basées sur une procédure d'appariement entre les éléments de deux listes. Par ailleurs, la distance de *Hausdorff*, distance entre ensembles, peut être utilisée sur les représentations de Voronoï . Plusieurs outils d'analyse et de comparaison de ces distances sont proposés et mis en œuvre. Ces experiences montrent des limites importantes de la distance de *Hausdorff* relativement à l'optimization par AG.

III.6 Analyse des paysages de fitness des representations de formes

Des reponses heuristiques sont proposées à la question du choix d'une initialisation et d'une representation *a priori* pour les représentations de longueur variables:
Les distances génotypiques permettent d'utiliser le graphe de corrélation pour comparer la diversité des procedures d'initialisation (méthode proposée dans le deuxième chapitre). Cependant, la corrélation est incapable de différencier les représentations.
La methode de comparaison des opérateurs d'évolution par tri des enfants (chapitre II), permet de comparer les representations par l'intermediaire de leurs opérateurs de croisement et de mutation respectifs.

Overview of Chapter III

Previous experimental studies on the difficult Structural Mechanics problem of elastic inclusion identification has demonstrated that specific variable length representations (through Voronoï diagrams, or based on lists of elementary shapes), give better results than the "classical" binary representation based on a mesh of the structure.
After recalling the inverse problem, the representations, and the experimental results, this chapter proposes a general approach to design genotypic distances for variable length representations. The resulting distances are compared to the standard *Hausdorff* distance. Using such genotypic distances, it is then possible to apply the results of Chapter II to perform an *a priori* choice among the variable length representations at hand, and to tune their initialization procedures.

III.2 An inverse problem

In the framework of 2-dimensional linear elastic mechanical structures, the goal is to identify possible defects inside a given structure using the results of mechanical experiments – e.g. the displacement of some points of the structure when known forces are applied.
This problem is known to be theoretically ill-posed, and standard numerical methods are either impossible to use, or very unstable. Considering this inverse problem as an optimization problem allows us to use evolutionary algorithms – provided that suitable representations of defects inside a structure can be defined.

III.3 Representations for planar partitions

The well known binary representation for structural design can be used for the elastic inclusion identification problem when a single type of defect is considered: Each element of a mesh of the design domain is either sane (1) or defectuous (0). However, the complexity of this representation is that of the underlying mesh.
Using Voronoï diagram theory (adding a label 0 or 1 to each site) or describing the defects as a list of elementary shapes are two possible ways to address the complexity issue. Moreover, the resulting search spaces seem more suitable for the inclusion identification problem, whose solutions are expected to contain large homogeneous blocks. However, in both cases, the resulting search space is that of an unordered variable-length list of alleles.

III.4 Experimental results

Using the variable length representations described above, some artificial (the defects are a priori known) but difficult instances of the inclusion identification problem can be solved accurately and in reasonable computing time. Moreover, the identification is robust with respect to noise in the experimental data.

III.5 Genotypic distances

A general approach to design distances on spaces of unordered variable-length lists of alleles is presented, based on a similarity measure on the space of alleles, and on a stochastic matching of two such unordered lists. These distances are first compared to the *Hausdorff* distance, whose limits as a tool in the evolutionary optimization framework are enlightened.

Some general criteria are proposed to qualify different genotypic distances motivated from the analogy with the binary *hamming* distance. These criteria are based on the links between the distances and the degeneracy of the representation on the one hand, and between the distances and the evolutionary operators (mutation and crossover) on the other hand.

III.6 Fitness landscape analysis for inclusion representations

Using the distances defined in Section III.5 above, in order to compute the statistical indicators of performance described in Chapter II, it is possible to – partially – address the issue of the *a priori* choice of a representation for the inclusion identification problem.

The Fitness-Distance analysis allows us to compare and tune the initialization procedures (as proposed in section II.5), but does not reflect the off-line results of the different variable length representations.

The *offspring ranking* method (section II.6), on the other hand, gives more accurate predictive results for the comparison of the different representations, based on their respective crossover and mutation operators.

Chapter III
Variable Length Representations

III.1 Introduction

Evolutionary Algorithms (EAs) involve two spaces during their search for optimality: the phenotype space, where the fitness values is computed, and the genotype space, where the evolution operators are applied. The choice of the phenotype space is usually directed by the problem at hand. On the other hand, the choice of the *representation* of phenotypes, that is the choice of the genotype space and of the decoding function between a genotype and the corresponding phenotype, is left to the EA practitioner. Far up-stream from the usual choice of parameters (population size, operator probabilities, ...), the choice of representation is now unanimously recognized among the Evolutionary Computation community as being of vital importance for the success of the whole optimization process [Mic96, Fog95, B9̈5]. Furthermore, recent theoretical results [WM95, RS95b] suggest that black-box evolutionary optimization is bound to failure, hence emphasizing the need for a problem-dependent algorithm: the first possible problem-dependent choice is that of representation.
However, very few – if any – guidelines are available to help choose the best representation for a given problem. The final choice mostly depends on the programmer's experience and/or faith, on a few theoretical or heuristical studies, and on the large body of experimental comparisons performed by researchers on other problems. Two aspects of such studies make it even more difficult when facing a new problem. First, their conclusions are either contradictory – as witnessed by the endless debate "binary vs non-binary coding" [Gol91, Ant89] – or drawn in two different frameworks, as for the coding of real parameters [Sch81, CS88, Rad91a, JM91, ES93]. Second, it is very unlikely that any conclusion drawn from experimental results on a given problem applies to a brand new problem.
Among the few theoretical studies devoted to the study of representations, Radcliffe's theory of *formae* (which are generalized schemata) [Rad91a] offers a theoretical framework that gives design principles for operators on a given representation. One interesting consequence is the definition of representation-independent operators and evolutionary algorithms [SR96]. But it also allowed *a priori* comparisons of different representations for the same problem that were confirmed by numerical experiments on representations for the TSP problem [RS95a]. The basic notion for these comparisons is that of "related fitness" of formae, and the tool was statistical evaluations of the fitness variance among formae of prescribed order. However, the definition of formae (or, equivalently, of equivalence relations on the genotype space) for variable length representations is not straightforward. For instance, the notion of "sharing an allele" should be relaxed to "having some similar allele" – which is exactly what a distance tries to describe. Further, the concept of related fitness can then be partially captured by studying the relation between distance and fitness.

Some other approaches have been proposed, using statistical estimations over the search space to characterize problem difficulty [Gre95b, Kar95b, HR96a, REA96]. Their pros and cons will be discussed in Section III.7, in light of the results based presented in this chapter.

This chapter proposes a case study of some variable length representations (unordered lists of

real-valued *alleles*) that were designed for solving some difficult inverse problems of Structural Mechanics, the elastic inclusion identification [Sch95, Sch96, SJK97, SKJ96]. The phenotype space is a subspace of the space of two-dimensional "black-and-white" shapes (though easy to extend to any dimension) i.e. partitions of some given initial domain in $\mathbb{R}^2$ into two subsets with continuous boundaries.
More examples of variable-length representations in EAs have long been proposed, though: The seminal work in Evolutionary Programming [FOW66] evolved Finite State Automata with variable connections; The Messy GAs [GKD89, Kar95a] use a list of (position,gene-value) to represent bitstrings, emphasizing the flexibility of this representation toward linkage between allele positions; Tree structures representing S-expressions are the heart of Genetic Programming [Koz94]; Evolutionary approaches to Neural Networks topology optimization (see the survey in [Yao93], or [ASP93, Gru94]) are, by construction, variable-length representations; the Evolutionary planner [XMZT97] represents paths for a mobile robot as a list of intermediate goals. But none of the above mentioned works make use of a genotypic distance.

The aim of this work is to design and use distances on the genotype space, allowing a further analysis of the fitness landscape, in the perspective of evolutionary optimization. The ultimate goal is to derive useful hints for further tuning of the algorithm parameters, including the choice of the best representation and initialization, when facing a new instance of such problems or representations.

The chapter is organized the following way. Section III.2 introduces the mechanical problem of inclusion identification in Structural Mechanics. Section III.3 presents different representations for two-dimensional "black-and-white" shapes, i.e. partitions of some given initial domain in $\mathbb{R}^2$ into two subsets with continuous boundaries. Namely the *bitarray*, a fixed-length binary representation, then the *Holes* and *Voronoï* representations, both variable-length unordered representations, exhibiting fairly high degeneracy and epistasis; and some hints are given for the *a priori* choice of a representation.
These representations are compared in section III.4 on some instances of the inclusion problem, pointing out the limits of the bitarray representation, which is largely outperformed by the variable-length representations.

After motivating the need of a distance in the genotype space, section III.5 proposes a general method to design genotypic distances on search spaces of variable length unordered lists of alleles. The "only" prerequisite is some similarity measure on the allele space.
However, different choices of allele similarity can lead to different distances, and raise the issue of the choice among distances on the same search space. This leads to design some comparison tools for genotypic distances, related to their behavior with respect to either the evolution operators and GA dynamics, or the degeneracy of the representation. These are general fitness independent tools that are partially motivated from the analogy with the binary *Hamming* distance properties.

The last part of the section instantiates the methodology designed in section III.5.1 respectively on the *Holes* and *Voronoï* representations: different genotypic distances are built, and compared with respect to the proposed tools of section III.5.4. These adequacy measures show large differences between these distances and the *Hausdorff* distance, especially when GA optimization of *distance fitness* functions is involved (the fitness, to be minimized, is the genotypic distance to optimum).
The latter section allows to choose the distances that seems best suited to study the Holes

and Voronoï based genotypic spaces for evolutionary optimization. This choice ensures that there is a relation between the distance and the operators of the algorithm at hand, but is independent from the inclusion problem under study. A different algorithm, may yield a different distance choice.

Finally, section III.6 designs and illustrates tools for landscape analysis. Most methods dedicated to fitness landscape analysis, are based on random sampling techniques, and a uniform random sampling of the search space is critical to derive useful results. Procedures for deriving problem or case dependent sampling are proposed in subsection III.6.1.1. The next subsections further address issues regarding the statistical estimation of problem difficulty, pointing out FDC limits for this purpose, but confirming the usefulness of local FDC concept introduced and experimented in section II.5 to choose (and tune) initialization procedures. In addition, the notion of operator potential, introduced in section II.6, also demonstrates that it is reliable for an *a priori* choice of a representation (as representations here are directly related to their crossover and mutation operators).

The proposed methodology and its results are discussed in Section III.7, in the light of related works in the literature. Moreover, further directions are proposed to correct some weaknesses of the approach. the conclusion contains a summary of results.

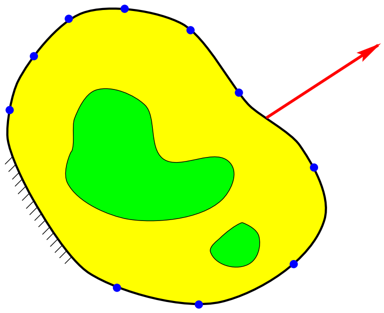


Fig 44 : *A sample inclusion problem*

III.2 An Inverse Problem

The first part of this section gives a detailed presentation of the mechanical problem of inclusion identification, and states a simplification hypotheses made throughout this study. The second part presents the inverse problem setting and general numerical framework and the Evolutionary Algorithm that will be used.

III.2.1 INTRODUCTION

In structural mechanics, the non-destructive identification of inclusions is a difficult problem, mathematically ill-posed, and resisting up-to-date numerical methods [LRN93, Con94]. In steel manufacturing plants, for instance, it is of vital importance to check if coal scoarcie are included in steel parts, and if their repartition does not dangerously weaken the whole part. In its simplest instance, a structure is known to be made of two different materials of given mechanical characteristics, but their repartition in the structure is unknown. The available data are the records of the mechanical behavior of the structure under known loadings. The goal is to find the geometrical repartition of both materials from these experimental data. For a given repartition of both materials, the computation of the simulated mechanical behavior of the structure is straightforward, using any Finite Element Analysis (FEM) package. The identification can then be viewed as an inverse problem.

The mechanical problem of inclusion identification is presented in detail in subsection III.2.2. Subsection III.2.3 introduces the framework that is used to address this inverse problem with EAs. A possible objective function for such inverse problems is the difference between the simulated mechanical behavior of a tentative repartition of both materials, and the actual (experimental) behavior of the real structure.

III.2.2 BACKGROUND OF THE DIRECT MECHANICAL PROBLEM

Consider an open bounded domain $\Omega \subset \mathbb{R}^N$ $(N = 2, 3)$, with a sufficiently smooth boundary $\partial\Omega$, filled with a linear elastic material (as in Figure 44). Under the hypothesis of small deformations (linear elasticity) the following equations hold:

$$\varepsilon(x) := \frac{1}{2}(\nabla u(x) \nabla u^T(x)) \qquad \text{the strain tensor,} \tag{164}$$

and

$$\sigma(x) := A(x)\varepsilon(x) \qquad \text{the stress tensor,} \tag{165}$$

where $u(x)$ is the displacement field at point x and $A(x)$ is the elasticity tensor (a fourth order tensor) involved in the *Hooke's law* (165): A is supposed to be *inhomogeneous*, which means that its value depends on the point x, and $A(x)$ is a positive definite tensor which satisfies some symmetry conditions.
When A is given, one can state two kinds of boundary value problems, respectively of *Dirichlet* and *Neumann* type:

$$\left\{\begin{array}{rcll} \mathrm{div}\sigma & = & 0 & \text{in } \Omega, \\ u & = & u_0 & \text{on } \partial\Omega, \end{array}\right. \tag{166}$$

and

$$\left\{\begin{array}{rcll} \mathrm{div}\ \sigma & = & 0 & \text{in } \Omega, \\ \sigma.n & = & g_0 & \text{on } \partial\Omega, \end{array}\right. \tag{167}$$

where u_0 and g_0 are, respectively, a given displacement field and a given external force field on the boundary $\partial\Omega$.
It is well-known (see *e.g.* [Cia78] that each of these problems has a unique solution (for the Neumann's problem, one has to impose an integral condition on g_0 to ensure existence, and an integral condition on u to eliminate rigid displacements).
In the following, the *inverse problem* will be considered:

$$\text{find } A \text{ such that },\forall i \in \{1,...,n\}, \exists u, \left\{\begin{array}{rcll} \mathrm{iv}\ \sigma & = & 0 & \text{in } \Omega, \\ u & = & u_i & \text{on } \partial\Omega, \\ \sigma.n & = & g_i & \text{on } \partial\Omega, \\ \sigma & = & A\varepsilon, & \end{array}\right. \tag{168}$$

where $(g_i)_{i=1,..,n}$ and $(u_i)_{i=1,..,n}$ are given.
Problem (168) is a discrete version of the "ideal" inverse problem:

$$\text{find } A, \text{ given the application: } \Lambda_A : u_{|\partial\Omega} \longrightarrow \sigma_{|\partial\Omega}. \tag{169}$$

The underlying physical problem is still lacking much more of known data than problem (168) since Λ_A is only known through a finite number of *experimental measurements* performed at a finite number of points. Hence, the "real" identification problem treated by mechanical engineers can be stated as:

$$\text{find } A \text{ such that },\forall i \in \{1,...,n\}, \exists u, \left\{\begin{array}{rcll} \mathrm{div}\ \sigma & = & 0 & \text{in } \Omega, \\ u(x^j) & = & u_i^j & \forall j \in \{1,...,p\}, \\ \sigma(x^j).n & = & g_i^j & \forall j \in \{1,...,p\}, \\ \sigma & = & A\varepsilon, & \end{array}\right. \tag{170}$$

where information on the boundary is only known at a finite number of experimental points $(x^j)_{j=1,...,p}$ for a finite number (n) of experiments. In addition, these data may be known with a certain amount of experimental error or noise.

State of the art If the aim is the numerical treatment of problem (169) (and *a fortiori* problem (168) or (170)) by "classical" (*i.e.* non-stochastic) methods, two theoretical points are crucial:

- existence and uniqueness of A as a function of Λ_A,
- continuity of the dependency of A with respect to Λ_A.

Existence is of course essential to the pertinence of the identification problem, but uniqueness and continuity are only needed to ensure the reliability and the stability of deterministic numerical algorithms. On the other hand, EAs can deal with non-continuous functionals and non-unique solutions.

Problem (169) is the elastic equivalent of the so-called *tomography* problem where the elliptic operator is the conductivity operator ($\mathrm{div}(A\nabla u)$, u scalar field) instead of the elasticity one ($\mathrm{div}(A\varepsilon(u))$, u vector field).

The tomography problem has been widely studied. Under some hypothesis, existence and uniqueness have been proved. However, the continuity of the functional is only known in a weak sense, which cannot help numerical simulations.

The elasticity problem (169) is more difficult. Existence and uniqueness have been proved for isotropic Hooke's laws, but there is no continuity result (see [Bui94, Bui95], or [Con94] for a comprehensive bibliographical discussion on this subject).

Numerical simulations by classical methods have shown that both tomography [KM90] and elastic identification problems [Con94] are ill-posed, and thus EAs are good tentative choice for a robust numerical method.

The simplified problem To highlight specific problems involved in these algorithms, the mechanical problem (170) was restricted to a two-dimensional simpler class of problems:

Let A_1 and A_2 be two isotropic elasticity tensors, fully defined by Young's moduli E_1 and E_2 and Poisson ratios ν_1 and ν_2. The aim is to solve Problems (168) and (170), restricting allowable values of $A(x)$ to

$$A(x) = \begin{cases} A_1 & \text{if } \chi(x) = 0 \\ A_2 & \text{if } \chi(x) = 1 \end{cases} \tag{171}$$

where χ is a characteristic function defined on Ω.

These problems, although less general than (168) and (170) are still beyond the capabilities of deterministic algorithms (see [Con94]).

However, Problem (170) can be treated in the same way, as well as identification in non-linear elasticity (this is part of on-going work, see Section III.4.4).

The direct elasticity problem is solved by a classical finite element method. All simulations (see Section III.4) have been performed on a square domain, held fixed on its left side, discretized by a regular mesh of size 24×24 elements. All details and mechanical constants are specified in Section III.4.

III.2.3 THE INVERSE PROBLEM OF INCLUSION IDENTIFICATION

The problem to be solved is the inverse problem corresponding to the direct problem described by Equation 171 above. Having a good simulation of a direct problem through Finite Elements Analysis, Evolutionary Computation can be used to address the inverse problem in a straightforward manner, as described schematically in Figure 45:

A structure (the *design domain*) is made of two different known materials, but their respective repartition in the structure is unknown. The available data consists in the experimental records of the mechanical behavior of the structure under known loadings (displacements of some points on the structure when given point-wise forces are applied). The goal is to find the geometrical repartition of both materials from these experimental data.

The identification is an inverse problem: For a given repartition of both materials, the computation of the simulated mechanical behavior of the structure is straightforward using any Finite Element package, and allows the comparison of the simulated behavior to the experimental reference data. The goal is then to find the optimal repartition (i.e. whose

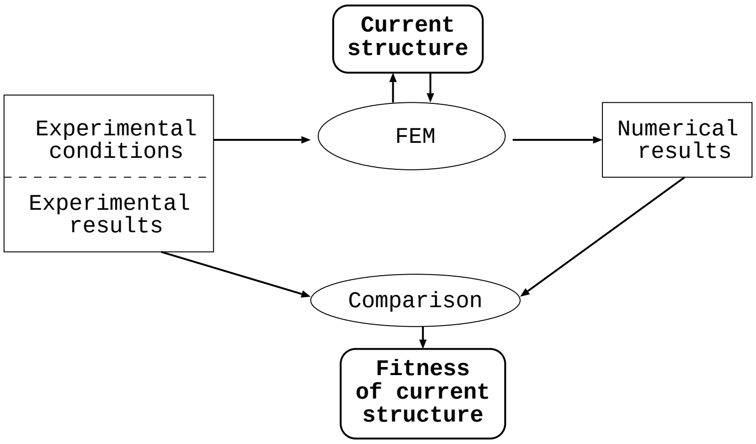


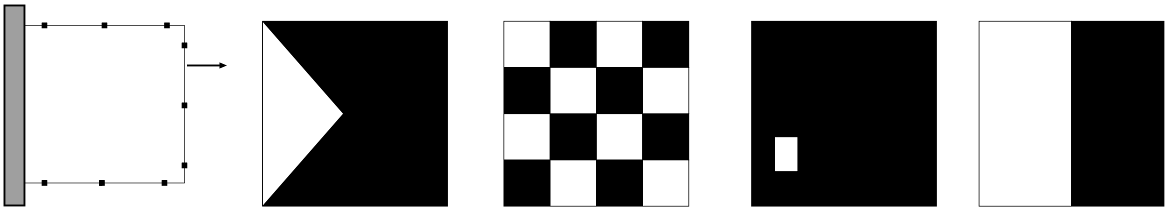
FIG 45 : *Evolutionary approach for the inverse problem of mechanical inclusion identification: The fitness of the individual at hand is achieved by comparing the actual experimental results with the numerical results obtained by Finite Element Analysis on the structure that the individual represents.*

simulated mechanical behavior is the most similar to the reference one). For instance, when reference data are displacements, the least-square error on the displacements is the fitness function to minimize.

Experimental settings All numerical results for the problem at hand (Section III.2.2) will be obtained on the two-dimensional square structure presented in Figure 46-a: the structure is held fixed on its left-side, and normal point-wise forces are applied at given points of the three other boundaries.
The aim is to identify the repartition of two materials into the square: a hard material ($E = 1.5$ and $\nu = 0.3$) and a soft material ($E = 1$ and $\nu = 0.3$). A fixed mesh of size 24×24 will be used used throughout the experiments, unless otherwise mentioned.
For the purpose of this study, some simulated simple instances of mechanical inclusion identification are used as the "experimental" data: At least one global solution is known, allowing more precise insight in the results. Figure 46 shows the geometrical layout of the mechanical problem, and the four reference structures used throughout this study. The fairly easy *corner* problem and the *checkerboard* problem, difficult in the eyes of mechanical engineers, are taken from [Con94], while the *asymmetric* and *symmetric* problems are purposely designed to test different behaviors of the representations introduced further in this section.

The fitness functions In real world situations, the design of the fitness function should take into account all available loading cases experimented on the real structure, using as much information as possible about the behavior of the structure to be identified. However, the choice made in these preliminary experiments of using "simulated experiments" makes it possible to use as many loading cases as needed. In an attempt to have a sampling of the structure's mechanical behavior as uniform as possible over the domain, 37 different loading cases were used, each loading case consisting of applying a given normal force at one point of the boundary of the structure (as shown on Figure 46-a).

A very important issue in inverse problem solving is the *generalization capability* of the solution: How good is the result of the optimization process when used under experimental



(a) The problem (b) Corner (c) Checker-board (d) Asymmetric (e) Symmetric

FIG 46 : *The mechanical framework: (a) is the experimental layout of the structure, together with one loading case (a normal point-wise force). (b)-(e) are the four test cases, used to generate the "experimental" results – thus also representing one solution to the corresponding inverse problem.*

conditions that are different from the ones used during the optimization itself? The usual answer is to use, during the identification, many different experimental conditions, also termed *fitness cases*. The fitness is then the average of the error over all fitness cases. Needless to say, the total computational cost increases with the number of fitness cases N:

$$f(x) = \frac{1}{N}\sum_{i}^{N} |d_i(x) - d_i(x^*)|,$$

where x is a two dimensional partition of Ω, and $d_i(x)$ is the displacement at the i^{th} measurement point of structure x.
However, using simulated "experimental" values allows us to test the overall algorithm with different points of view: In real world situations, some gauges are placed on the boundary of the structure, and only the displacements at those points are available. Figure 46-a is an example of such a situation, where 9 gauges are represented. However, it seems clear that the more measure points, the easier the identification task. Hence, three different fitness functions have been used throughout the numerical experiments presented in Section III.4, using the 37 loading cases described above, with different numbers of measure points:

- The most informative fitness function, referred hereafter to as the *total fitness*, takes into account the displacements at *all nodes* of the mesh.
- An intermediate fitness function uses only the displacements of all nodes lying at the boundary of the structure, and is termed the *boundary fitness*.
- The *real-world fitness* uses only the 9 measure points represented in Figure 46-a, incorporating much less information to the fitness function.

The phenotype space The phenotype space is a subspace of the space of two-dimensional (though easy to extend to any dimension) "black-and-white" shapes (i.e. partitions of some given initial domain in $\mathbb{R}^2$ into two subsets with continuous boundaries). Such spaces of shapes appear naturally in image processing applications [LN95], as well as for problems of shape optimization [CJ95, GMP96, Sch96]; topological Optimum Design, where the goal is to find the shape of a mechanical structure with minimum weight and sufficient strength

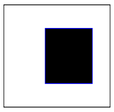
[Jen92, CSJ94, Kan96]; the inverse problem of geological soil properties [SEB98]; and the identification of mechanical inclusions [SJK97]. The latter will be the test problem in the present chapter, and will be detailed in next section.

However, different representations are available for the phenotype space of partitions of a design domain of $\mathbb{R}^2$, and the choice of the best representation for a given problem instance is still an open problem, whatever the target application.

The search space The most critical step in evolutionary algorithms is the choice of the representation, which defines the search space. In any case, the algorithm will find a good – if not the best – solution in the search space. Hence it seems that larger search spaces allow better solutions to be found. However, the larger the search space, the more difficult the optimization task, and a trade-off must be found.

Considering past works on the Optimum Design problem, (a closely related problem from the Evolutionary Computation point of view, where the goal is to find a partition of a design domain into material and void), a straightforward representation is defined from a fixed mesh of the structure, leading to a fixed-length bitstring representation well-suited to standard Genetic Algorithms. This "bitarray" approach allowed significant breakthroughs in Topological Optimum Design [KJS95, KS96]. However, this approach suffers from strong limitations due to the dependency of its complexity on the underlying mesh: the optimization problem rapidly becomes intractable when this mesh is refined.

To cope with that difficulty, two non-standard representations (termed the *Voronoï representation* and the *H-representation*) were introduced [Sch95, Sch96], independent of any *a priori* mesh size, but leading to variable-length "individuals" for which specific operators were designed and implemented. These representations are presented in next section.



$\longrightarrow$

```
000000
001110
001110
000000
```

$\longrightarrow$ [000000011100011100000000]

Phenotype – – – – Genotype – – – – pseudo-equivalent Genotype

FIG 47 : *A sample bitarray representation for a* 6×4 *mesh.*

III.3 Representations for 2D Partitions

This section introduces actual instances of variable length representations. All ideas presented in further sections will be experimented using these representations.

III.3.1 BIT ARRAY REPRESENTATION

A straightforward representation from the GA point of view, and the most commonly used representation for the phenotype space $\mathcal{P}$ is an array of bits (termed *bitarray*), based on a regular discretization (*mesh*) of the design domain into $n \times n$ *elements* (see also [Jen92, CSJ94]). Each bit corresponds to one element, and indicates which of the two subsets this element belongs to. Figure III.3.1 shows an example of a bitarray representation for a simple 6×4 mesh: The genotype space can thus be viewed as the standard space of bitstrings $\{0,1\}^{n \times n}$. Note however that such a point of view is misleading when designing the crossover operator: the standard multiple-point crossover operators are geometrically biased, and specific two-dimension operators must be designed, as demonstrated in [KS95].

Limits of the bitarray representation When designing a representation for that phenotype space of the partitions of two materials, some considerations about Finite Element Analysis must be taken into account: Since the fitness values are computed using a Finite Element Analysis of the phenotype, the structure must first be discretized. It is well-known that the discretization step introduces numerical noise in the resulting simulated mechanical results. But the Evolutionary Algorithm heavily relies on the comparisons of such results. Hence, a crucial point is that all Finite Element Analysis of structures whose mechanical behavior are to be compared (i.e. in the same population) should be performed using the same mesh. A detailed discussion of this issue can be found in [Kan96, Sch96].
Due to this mandatory use of the same discretization for all structures in the population, or perhaps simply because it is a straightforward representation from the GA point of view, the most commonly used representation is the binary representation presented in section III.3.1.

Nevertheless, the bitarray representation encounters very severe restrictions from the mechanical and numerical points of view. The accuracy of the Finite Element Analysis is dictated by the size of the mesh. Increasing the size of the mesh would not only increase the cost of the fitness computation (whose complexity is quadratic in the size of the mesh), but also the size of the bitarrays. This in turn would require increasing not only the size of the population, as heuristically derived in [GDC92] and theoretically proved in [Cer96], but also the number of generations needed to reach the same level of convergence. Hence, the bit-array representation poorly scales up when refining the mesh, not to speak of handling 3-D problems.

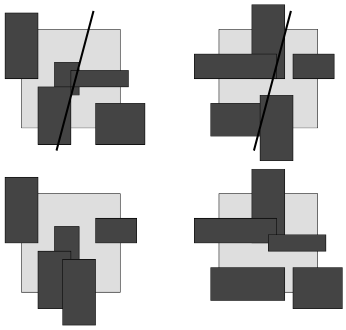


FIG 48 : *Example two genotypes for the rectangle representation (above). The phenotypes are the partitions of the design domain into the black and the grey subsets. The crossover operator (below) exchanges the holes whose center is on one side of a random line drawn across the design domain.*

Therefore other representations (presented in III.3) have been designed in order to overcome this limitation, and separate the complexity of the representation from the accuracy of the fitness evaluation [Sch95, Sch96].

III.3.2 HOLES REPRESENTATION

A first variable length representation for partitions is based on an old-time heuristic method in Topological Optimum Design: from the initial design domain, remove material where the mechanical stress is minimal, until the constraints are violated. However, the lack of backtracking makes this method useless in practical applications. Nevertheless, this idea gave birth to the *Holes* representation (another approach in the domain of dynamical image processing led to a similar representation in [LN95]).

Genotype space: Consider a list of holes $(H_1, H_2, \dots, H_l)$, $l \leq L_{max}$, each having an elementary shape and taken from a library of possible simple shapes, and variable dimensions. The design domain receives a default value (e.g. subset 0), and the list of "holes" describes the topology of the other subset (e.g. subset 1). An important point is that the holes must be allowed to extend out of the initial domain Ω. Otherwise, the boundary regions of Ω would have a very high probability to stay at the default value, imposing a strong bias on the resulting phenotypes.
Note also that the choice of the default value is arbitrary. Hence, two different representations can be defined, the *1-Holes* and the *0-Holes*, specifying the value represented by the holes.

Rectangles Representation: The first trials used only rectangles as elementary shapes [Sch95, Sch96]. Figure 48 gives an example of two genotypes for the *Rectangle* representation. In that case, a genotype can be written as $(R_1, R_2, \dots, R_l)$, $l \leq L_{max}$, where each rectangle R_i is defined by the coordinates of its center, and both dimensions, four real numbers with prescribed bounds.

Triangles Representation: The triangle representation is another instance of a hole representation in which the only possible elementary shape for the holes is the triangle. An individual is thus represented in the *Triangle* representation as a list of triangles $(T_1, T_2, \dots, T_l)$, $l \leq L_{max}$, each triangle being defined by the coordinates of its three summits.

Decoding procedure: However, the mechanical considerations that lead to the design and use the bitarray representation still hold (section III.3.1), and structures in the same population must be compared to the same mesh. The first step of the fitness computation therefore is a projection on the fixed mesh at hand: This is done lazily by all elements getting the value of their center of gravity (i.e. the default value if that center is not in any hole, the hole value otherwise). But the important point is that the complexity of the representation (the number of holes) is not dependent on that of the mesh (the number of elements, determined from accuracy requirement on the mechanical model). The effect of mesh refinements will not be studied here (see e.g. [SKJ96]), and the same 24×24 mesh will be used throughout the chapter.

Degeneracy and epistasis: The *Holes* representation is clearly degenerate, in that many different genotypes actually represent the same phenotype. First, any hole that is covered by a sublist of the holes list can be removed without modifying the phenotype. Second, the projection on a fixed mesh introduces another degree of degeneracy, as the holes can be modified slightly (by less than a mesh-size, in average) without modifying the phenotype. This aspect of the *Holes* representation will have to be taken into account in the following.
In the *Holes* representation, the expression of a hole in the phenotype cannot by any mean be modified by another hole: the epistasis of that representation is minimal. However, when considering the parts of the phenotype that have the default value, the addition of a single hole will change that value to the hole value: from the crossover operator point of view, it means that the transmission of “genetic material” is highly asymmetric regarding both possible values for the phenotype parts.

Crossover operator: From the experience of the bitarray representation [KS95], the exchange of holes by the crossover operator is a geometrically-based choice: a random line is drawn across the design domain, and the holes, whose center are on one side of that line, are swapped between the parents (Figure 48 is an example of such crossover for the rectangle representation).

Mutation operators: Different mutations, addressing the different components of the representation, have been designed. User-supplied weights govern the choice of the operator to be used after an individual has been selected for mutation.

- A first mutation operator performs a Gaussian mutation on the – real-valued – parameters of a randomly chosen hole.
- "Standard" mutation operators for variable-length representations must include the addition and destruction of some elements of the list.
- The notion of *minimal mutation* will be needed further on: The definition for discrete allele space [SR96] (or [HR96a] where it is termed *neighbor search* operator) has been here abusively extended, and a minimal modification is the smallest mutation that propagates to phenotypes, and hence is visible on the fitness. Due to the decoding procedure (section III.3.2 above), the minimal mutation is defined as the mutation that modifies the label of at least one element of the mesh that is used to compute the fitness.

III.3.3 VORONOI REPRESENTATION

Another possible way of representing polygonal shapes comes from computational geometry, more precisely from the Voronoï diagram theory (see [PS85, BY95] for a detailed introduction

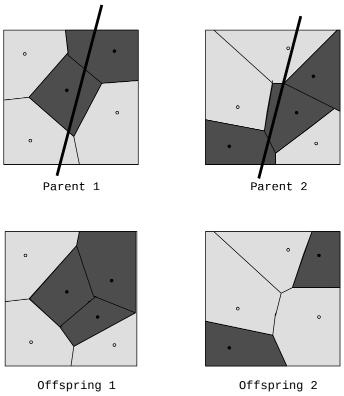


FIG 49 : *Samples of Voronoï representation: the genotypes are the lists of labeled sites, the phenotypes the resulting partitions of the design domain (above). Sites on the same side of a random line drawn across the design domain are exchanged by the crossover operator.*

to Voronoï diagrams, and a general presentation of algorithmic geometry).

Genotype space: Consider a finite number of points $(V_0, \ldots, V_l)$,, $l \leq L_{max}$ (the *Voronoï sites*) of the design domain Ω. To each site V_i is associated the set of all points of the design domain for which the closest Voronoï site is V_i, termed *Voronoï cell*. The *Voronoï diagram* is the partition of the design domain defined by the Voronoï cells. Each cell is a polygonal subset of Ω, and any partition of Ω into polygonal subsets is the Voronoï diagram of at least one set of Voronoï sites.
Consider now a list of Voronoï sites, in which each site is labeled 0 or 1. The corresponding Voronoï diagram represents a partition of the design domain into two subsets if each Voronoï cell is labeled as the associated site (here the Voronoï diagram is supposed regular (i.e. each cell corresponds to exactly one site). Example of Voronoï representations can be seen in Figure 49. The Voronoï sites are the dots in the center of the cells.

Decoding procedure: As for the *Holes* representation, the decoding procedure of the Voronoï representation first projects the labeled Voronoï diagram on the fixed mesh used for the Finite Element Analysis by considering the cell the centers of the elements belong to.
Degeneracy and epistasis: As in the case of the *Holes* representation, the degeneracy of the Voronoï representation is clearly very high: many sites can be added to the genotype without modifying the phenotype (e.g. in the corners of the design domain).
On the other hand, the epistasis of the *Voronoï* representation is high, symmetrically with respect to both labels: the size of the Voronoï cells depends on all the neighboring sites. For instance, a 0 site closely surrounded by 1-sites might not appear at all in the phenotype.

Evolution operators: The evolution operators are quite similar to those of the *Hole* representation:

- Crossover by the geometrical exchange of sites (see Figure 49 for an example);
- Mutation by the modification of existing sites (Gaussian mutation of coordinates, or flip of label), or by addition or destruction of some sites.
- The minimal mutation is the displacement of a site by the size of one element.

Some results, using the Voronoï representation, are presented in next section, demonstrating outstanding performance on artificial instances of the inclusion identification problem.

III.4 Numerical Results on Inclusion Identification

This section presents some numerical experiments on the inclusion identification problem using Evolutionary Algorithms. The first results address the feasibility issue. In subsection III.4.2, the EAs demonstrate their robustness in the presence of noise in the experimental data. Finally, comparative experiments of subsection III.4.3 bring a partial answer to the question of the a priori choice of a representation for a given instance of problem. In particular, the bitarray representation definitely demonstrates its poor performances when compared to the mesh-independent representations (Voronoï , 1- and 0-rectangles).

III.4.1 The Evolutionary Algorithm

The Evolutionary Algorithm used for all experiments presented in the following is a standard GA scheme: rank-proportional selection (only the rank in the population of all individuals is taken into account in the selection process, regardless of the actual fitness values), crossover rate of 0.5, mutation rate per individual of 0.5, all offspring replace all parents. The population size is set to 100, and at most 300 generations of the algorithms are allowed – but it stops whenever 50 generations are run without any improvement of the overall best fitness. Hence, between 10000 and 25000 Finite Element Analyses (each involving 37 loading cases, but with the same global matrix) were performed for each run, requiring around 2 hours of CPU time on a high range HP workstation (HP-PA 8000) for the 24×24 mesh used in most cases.

However, the stochastic nature of the algorithm forces all experiments to be run many times with different independent initial populations: as clearly stated in [Jr94], "You should never draw any conclusion of a single run of any Evolutionary Algorithm".
Hence, for each setting of experimental conditions (problem and representation), 21 independent runs have been performed, then the online performances (fitness along generations) or the off-line performances (fitness reached in the end of evolution) of these runs are compared: At each generation, the best individuals of all runs are sorted according to fitness, and the median value (here, the 11^{th} out of 21) is shown. Note that EA results are generally presented through their averages. But, as discussed in [FF96], whereas using the average values (together with T-test results or confidence range of average) makes the implicit assumption of normal repartitions for the sampled values, choosing the median values seems more adapted to non-parametric statistics. Moreover, when dealing with fitnesses that can take very small values (as the mean square errors of the displacements), averages hardly reflect the actual behavior of the algorithm: Error values for different runs can be of fairly different orders of magnitude. Hence, a few large values might hide a majority of very small errors. Nevertheless, the results of comparisons involved in this study, are the same when the median or confidence range of average run is used.

III.4.1.1 Feasibility Results

The very first experiments were performed using the Voronoï representation, described in Section III.4. They were obtained on the reference partitions represented in Figure 46 of section III.2.2.
The first results on the corner problem were astonishingly good: the exact solution was found

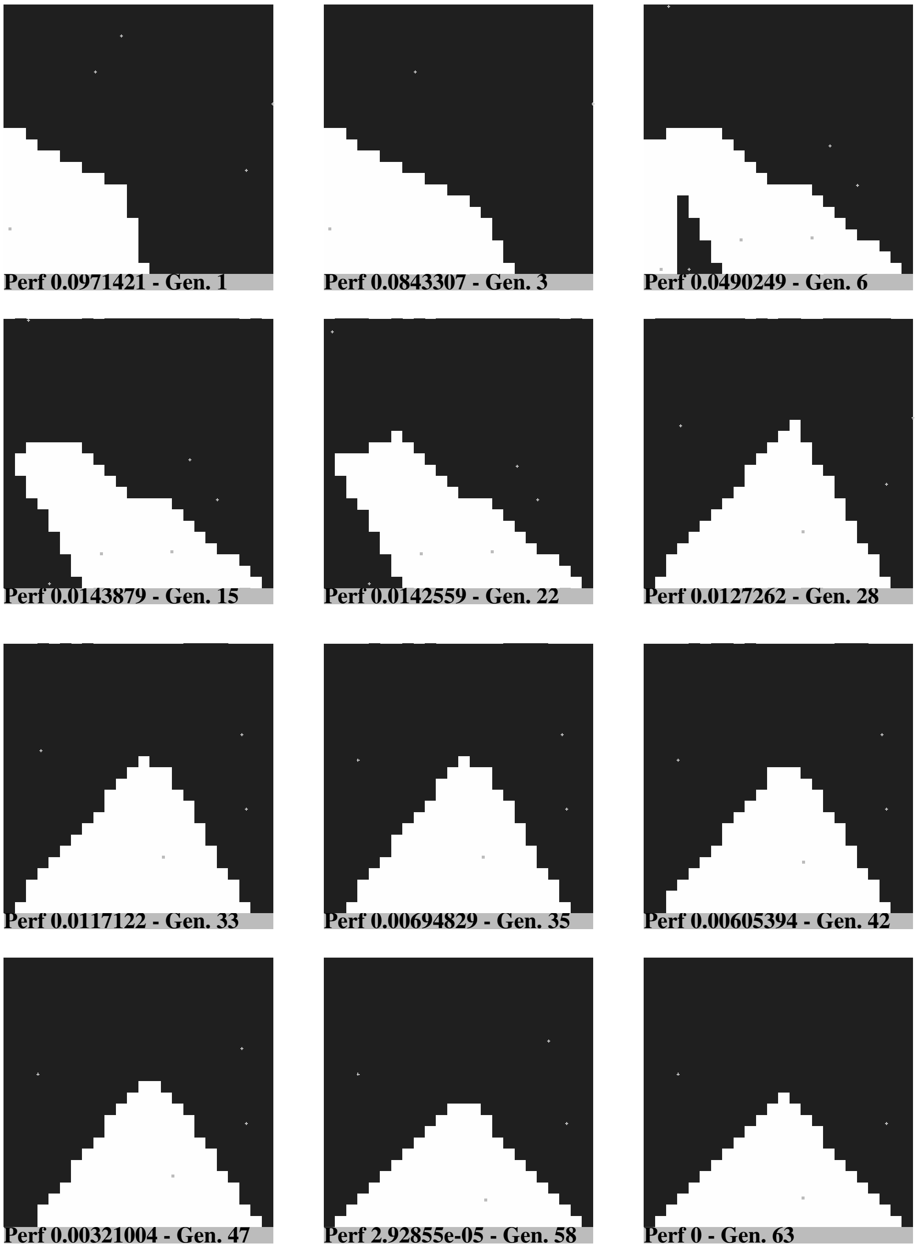
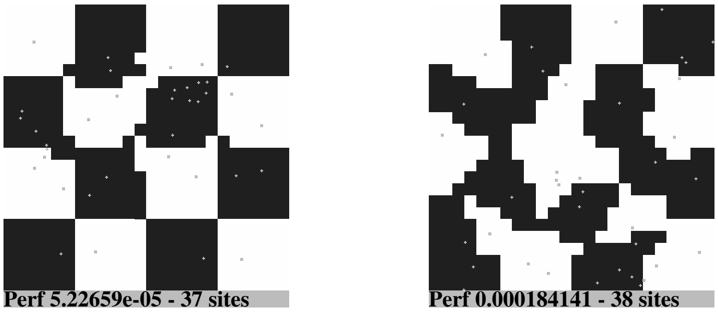
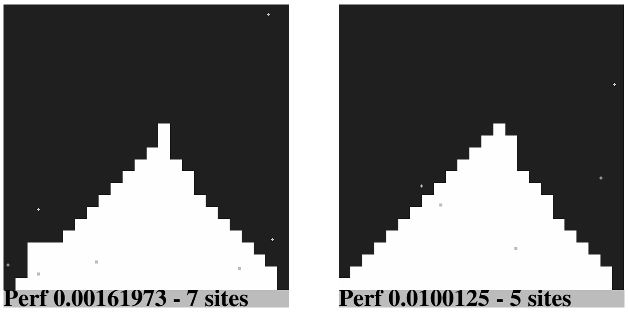


FIG 50 : *A successful run of the Voronoï representation on the corner problem using boundary fitness. Plots of the best individual at different generations of the evolution*



(a) (b)

FIG 51 : *The – difficult – checker problem: (a): with total fitness. (b): with real fitness.*



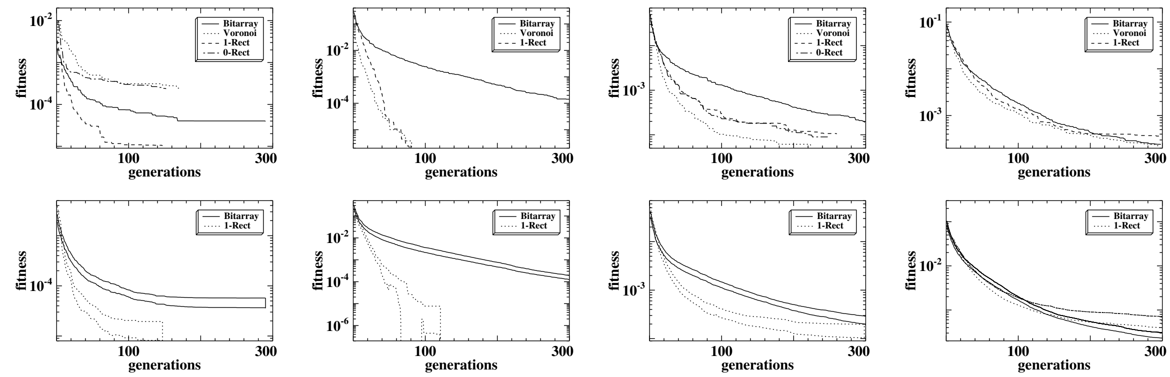
(a) (b)

FIG 52 : *Robustness to noise. (a): with 2% noise. (b): with 5% noise.*

3 times out of 10 independent runs, in less than 100 generations when using the "total" fitness, and even twice (in about 250 generations) using the boundary fitness. Figure 50 shows an example of a successful run, where the best individual at different stages of evolution is plotted together with the error. The Voronoï sites are represented on the figure by grey little dots.
On the symmetrical and asymmetric problems of Figure 46, some difference begins to appear between the different fitnesses, but these examples are still easy problems. And when it came to the checker problem, the total fitness gave much better results than the real-world fitness, as can be seen in Figure 51. However, the real-world fitness gave some interesting results, as can be seen on Figure 51-b: the actual values are clearly identified along the free boundary, and badly along the fixed part of the boundary, where very little information is available.

III.4.2 Results on noisy data

After these first satisfactory results on exact data, further validation of the proposed approach had to take into account possible errors and noise in the data, before using actual – noisy – experimental data. In order to test the algorithm's sensitivity to noise, artificial noise was purposely introduced in the reference "experimental" displacements.
Figure 52 shows the results obtained on the (easy) "corner" example, when the relative amount of noise is set to 2% and 5%: all reference displacements were multiplied by a term $(1+\varepsilon)$, where ε was a random variable uniformly distributed in $[-0.02, 0.02]$ and $[-0.05, 0.05]$ respectively. The results are (of course !) degraded, but they demonstrate a fairly good robustness, at least on this example.



(a) asymmetric (b) symmetric (c) corner (d) checker

FIG 53 : *Comparative results of 21 independent runs on the four test problems, for different representations, the bitarray representation starts at the same initial populations as the 1-rectangles representation. Median convergence curves are sketched at first line, and confidence range of some average results at second line: bitarray performances are lousy, except for the checker-board problem.*

	Asymmetric	**Symmetric**	**Corner**	**Checker**
bitarray uniform initialization	0.0013	0.0013	0.0014	4e-4
bitarray rectangles initialization	8e-6	4e-5	1e-4	2e-4
Voronoï	5e-6	0	0	7e-5
1-Rect.	0	0	4e-5	1e-4
0-Rect.	5e-5		7e-5	
Triangle	8e-6	0	3e-06	0.001

Table 7: *Off-line performances of GAs for all four* 24×24 *test problems and different representations. The best result from 21 runs of 300 generations each are presented. The figures are the* L^2 *error on the displacement at the measure points: A null best error means that the actual global solution was found exactly at least once.*

III.4.3 Comparative results

This section compares the four representations defined in section III.3, namely the bitarray, the Voronoï , the 1- and 0-rectangles and the triangle representations. As discussed in Section III.4.1, all results are median over 21 independent runs.

Figure 53 presents the median on-line results (i.e. the plots of error on the displacements along generations), while Table 7 presents the (best) off-line results after 300 generations. Note that the 0-rectangle was not tested on the two symmetrical problems (the symmetric and checker-board problems): though the problems are not mechanically symmetric (the hard and soft materials are exchanged), the asymmetry is hardly distinguishable from the EA point of view, and the results of the 0-rectangle representation are broadly of the same nature than those of the 1-rectangle on these problems.

Some clear tendencies emerge from these results:

- The bitarray representation (with uniform initialization) is clearly outperformed by all other representations, except on the checker-board problem with the triangle initialization. Note also that bitarray representation is still outperformed by the other

	Asymmetric	Checker
Voronoi	2e-5	5e-4
Rectangle	1.5 e-5	8e-4
Bitarray	0.082	0.03

Table 8: *Best results of off-line performances for two test problems on different representations using a 48×48 mesh. From one or three runs are aggregated here.*

representations when a biased initialization is used. Using the same initial populations as that of the 1-rectangle case, the best run of the bitarray trials (evolved up to 800 generations), gives errors of respectively $8e-6$, $1e-5$, $9e-5$ and $8e-5$ for the 4 problems. These results are still worst than the results obtained with variable length representations in 300 generations. In particular they were never better than the Voronoï representation.

- The Voronoï representation performs globally better than all Holes representations (apart on the asymmetric problem), with a significant difference for the corner problem.
- The results on the asymmetric problem demonstrates the asymmetry of the holes representations: the 0-rectangle is widely outperformed by the 1-rectangle on that – purposely defined – problem.
- The shape of the target subset that is to be matched by the holes does influence the relative results of the Holes-representations: the triangle representation outperforms the rectangle representation on the corner problem, while the reverse is clearly true on the checker-board problem.

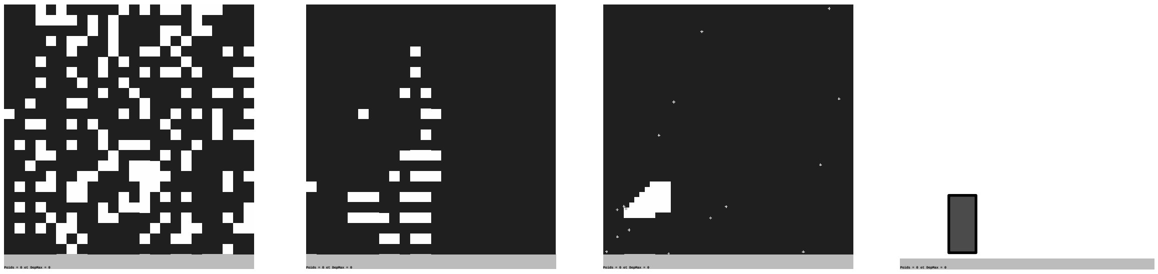
III.4.3.1 Influence of the mesh

This section presents preliminary investigations to a posteriori justify the a priori arguments against the bitarray representation related to its dependency on the underlying mesh (see Section III.3.1). Two different meshes are considered: the 24×24 mesh used in the preceding sections, together with a refined 48×48 mesh are used. The same 37 loading cases are considered in the fitness (see section III.2.3), and the reference displacements are computed on the same mesh than the following runs.

Table 8 gives the respective off-line results of the bitarray, the Voronoï and the 1-rectangle representations, for the asymmetric and the checker-board problems of Figure 46. Further, Figures 54 and 55 show the result obtained by the bitarray-, the Voronoï - and the rectangle-representation on the asymmetric and the checker-board problems for one run of 300 generations on the 48×48 mesh: whereas both variable-length representations obtain a solution resembling the actual structure, the bitarray representation ends up in a totally irrelevant result.

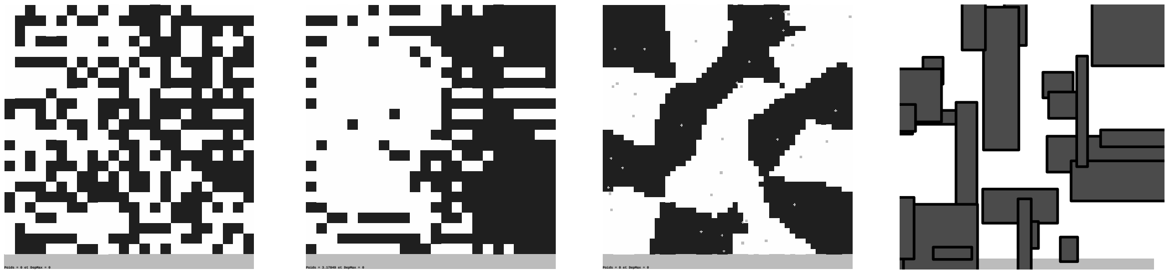
Note that a single run for the 48×48 mesh take about 10 hours of CPU time on a powerful HP-PA8000 processor; hence few runs were performed with the 48×48 mesh (less than 21). However, it is already obvious that the performance of the bitarray representation does indeed decrease rapidly when the complexity of the mesh increases.

On-going work addresses the more general problem (169) (i.e. the case where the mechanical characteristics of the materials are unknown). The proposed representations can be extended easily to that framework: the label of Voronoï sites can easily be replaced by a set of real



(a) Bitarray Uniform initialization (b) Bitarray 1-Rect. initialization (c) Voronoï (d) 1-Rectangles

FIG 54 : *Best structure after 300 generations on the asymmetric problem, on the* 48×48 *mesh (colors are inverted on the rectangle plot).*



(a) Bitarray Uniform initialization (b) Bitarray 1-Rect. initialization (c) Voronoï (d) 1-Rectangles

FIG 55 : *Best structure after 300 generations on the checker-board problem, on the* 48×48 *mesh (colors are inverted on the rectangle plot).*

values (e.g. the E and ν coefficient of the material); in the same line, real-valued coefficients can be attached to each hole in the Hole representation. The decoded values for a point in the structure will then be a function (ex. the average) of all values that cover it. First experiments on the problem of the identification of spatial geological models in petroleum prospection [SEB98] gave encouraging results.

III.4.4 DISCUSSION

The evolutionary method demonstrates to be powerful on the simplified problem of linear elasticity involving two materials of known characteristics, widely outperforming the to-date results of specialized numerical methods.

However, the – almost – standard representation of structures by arrays of bits (based on a given mesh) proved to be totally inaccurate for that type of problem, and the breakthrough results were obtained using non-standard representations together with specifically designed genetic operators. The Voronoï representation uses a variable number of labeled Voronoï sites to define a partition, while the Holes representations uses a list of elementary shapes over the structure: these elementary shapes indicate which parts of the structure are not made of the default value.

First, the complexity of these representations is independent of any discretization of the structure. Hence the complexity of the algorithm itself, in terms of number of fitness evaluations, only depends on the problem at hand.

Second, the Voronoï representation seems more robust with respect to a priori knowledge about the quality of the solution: unless the solution is known to be made of very small parts of one material spread over in the structure, in which case a hole representation with default value to the majority material is the best choice, the Voronoï representation performs as good or even better than the rectangle and triangle representations tested here. Moreover, the Voronoï representation is independent of any guess regarding the shape of the inclusion itself, whereas the holes representation perform poorly if the wrong guess is made.

Regarding the a priori choice of a representation for a given problem, the specific characteristics of the problem at hand should be taken into account: if the general shape of the small inclusions can be guessed, choosing elementary shapes for holes that easily fill up the supposed inclusion shapes can help the identification process. Though this is generally beyond what can be asked of mechanical experts, and *a priori* investigation of the fitness landscape can be of some help here, as will be shown in section III.6.

However, before investigating the fitness landscape properties, some tools have to be designed, the first of which are some genotypic distances on the space of variable length unordered lists of alleles.

III.5 Genotypic Distances

Evolutionary Computation (EC) generally distinguishes two spaces involved in the evolution process: the phenotype space, on which the fitness is computed, and the genotype space, where the evolution operators apply [Fog95]. Consequently, different approaches are possible to the notion of convergence of an Evolutionary Algorithm (EA). The fitness point of view only considers fitness values: all individuals (points of the genotype space) with same fitness are impossible to distinguish, and poor insight into the behavior of the algorithm can so be obtained. The phenotypic point of view looks at the points in the phenotype space: this gives a clearer picture of the algorithm's behavior than the fitness values alone. However, the accuracy of this approach can also give a strongly biased view of the fitness landscape, depending on the characteristics of the *representation* (i.e. the mappings between the phenotype space and the genotype space). If the representation is degenerate (i.e. different genotypes correspond to the same phenotype), again it will be impossible to separate different points of the genotype space. Moreover, if the representation presents a strong epistasis, the trajectory of a population in the phenotype space will look very different than in the genotype space: As the expression of a *gene* (a component) in the phenotype of an individual depends on the other genes, two individuals "closed together" in the genotype space can be quite far away from each other in the phenotype space. In order to precisely study these phenomena, a distance on the genotype space is mandatory. The epistasis, for instance, can be viewed as a defect of the isomorphism of the decoding mapping (from the genotype space into the phenotype space).
Many attempts to study the behavior of evolutionary algorithms on the fitness landscape rely on a distance in the genotype space: the computation of the correlation length of mutation operators studied in [MdWS91] uses a distance in the genotype space; the statistical regression in [Gre95b] is based on an implicit metric in the genotype space; the Fitness Distance Correlation [JF95] explicitly uses the distance to the optimum to derive a measure of problem difficulty; the density of states in [REA96] uses an estimate of the number of optima in regions of given measure; and the long path analysis in [HR96a] needs the notion of neighborhood, which is considered as built from an operator. Whatever the method or the goal of these studies (get a better understanding of the dynamics of the evolutionary algorithm, or/and be able to make good choices when tuning the parameters of the algorithm), they rely on a metric structure of the genotype space.
Note that there are yet other important features of Evolutionary Computation that require a distance on the genotype space, namely the different niching mechanisms. The crowding [DeJ75, Mah92] and the sharing [GR87] schemes (see [Mah95] for a survey) aim at controlling the spread-out of the population in the search space, and hence rely on the notion of proximity (though there exist phenotypic variant of these niching methods).

But the definition a genotypic distance ranges from obvious to very difficult, depending on the representation. Most EAs use fixed length representations, from the bitstrings of historical GAs [Hol75, Gol89] to the real-valued vectors of ES [Sch81, BS93] and EP [Fog92]. The definition of a distance on these spaces is straightforward (e.g. use the standard Hamming or Euclidean distances). Note however that the definition of a distance can be rather tricky even on well studied – fixed length – spaces such as the permutation space in combinatorial optimization (see [Ron97] for an example for the TSP problem). Anyhow, when it comes to non standard representations such as variable length lists, there is no canonical way to define a distance.

Nevertheless, previous work suggest different ways for constructing distances related to the evolution operators. For example, given a representation and a set of evolution operators, a possible distance is the minimal number of genetic operator applications to get from one individual to the other [HR96a, Jon95] (see Section III.7 for more details).
However, in practice, this latter definition cannot be used if genetic operators link any two points of the space with non-null probability, in one transition (this is the case of the binary mutation that flips each bit with some fixed probability). Next, the actual computation can be time consuming.
Further, in the case study of this chapter, the mutation operators involve real valued Gaussian perturbations, which involve very different probabilities of transition from one point to operator-neighboring points. Hence, the minimal number of operator applications is rather meaningless. This motivates the construction of alternative distances, indirectly related to the evolution operators and easier to compute.

This section proposes a case study of variable length representations (unordered lists of real-valued *alleles*), namely the Voronoï and Holes representations introduced in section III.3. The phenotype space is a partition of some given initial domain in $\mathbb{R}^2$ into two subsets with continuous piece-wise linear boundaries. Whereas the definition of a distance is obvious in the phenotype space, (from pixel-wise comparison of two black-and-white shapes, for instance), the aim of this work is to design and use distances in the genotype space, allowing a further analysis of the fitness landscape, as well as algorithm parameters tuning.

Subsection III.5.1 proposes a general method for designing genotypic distances on search spaces of variable length unordered lists in non-discrete spaces. The "only" prerequisite is some similarity measure on the allele space. However, different choices of allele similarity can lead to different distances, and raise the issue of choice among distances on the same search space.

In subsection III.5.4, the potential connections between the distance and the operators (e.g. crossover and elementary mutations) are investigated, mainly motivated by the analogy with the Hamming distance in the binary framework. Further, the distortion induced by the degeneracy of the representation is studied. We then consider that a distance is suitable to landscape analysis (relevant to the optimization process), if this distance is connected to the algorithm and its parameters.

Different genotypic distances are compared with respect to the proposed measures. These adequacy measures show large differences between these distances and the *Hausdorff* distance, especially when GA optimization of *distance fitness* functions is involved.

III.5.1 CONSTRUCTION OF GENOTYPIC DISTANCES

This subsection will now address the general case of the representations, where an individual is an unordered list of alleles. The basic problem when dealing with such representations is the definition of the (genotypic) distance. Three issues can raise some difficulty: The number of alleles can be variable; No canonical ordering of alleles might exist. When alleles are continuous parameters, there is no point in looking for common alleles between two individuals (as opposed to the corner-representation for the TSP [RS95a]). Some guidelines will be proposed to account for any combination of these three difficulties.

III.5.1.1 Prerequisites for a distance on $\mathcal{G}$

Suppose the genotype space $\mathcal{G}$ is that of unordered lists of *alleles*, all alleles taken from a space $\mathcal{A}$. Moreover, there exists δ a distance over the space $\mathcal{A}$.
Consider two individuals $x = (a_i)_{i\in[1,n]}$ and $x' = (a'_i)_{i\in[1,n']}$ of $\mathcal{G}$ (with $n \geq n'$). A *matching* between x and x' is defined as an injective application from $[1,n]$ in $[1,n']$. The set of all matchings between x and x' is denoted $M(x,x')$.

Proposition III.5.1
If δ is a bounded distance on $\mathcal{A}$, then

$$\Delta(x,x') = \min_{\mu\in M(x,x')} \big(\sum_{1}^{n'} \delta(a_i, a'_{\mu(i)})\big) + (n'-n)diam_\delta(\mathcal{A}) \tag{172}$$

is a distance on $\mathcal{G}$.

Proof.
Function Δ obviously satisfies
$\forall x \in \mathcal{G},\ \Delta(x,x) = 0$, and
$\forall (x,x') \in \mathcal{G}^2,\ \Delta(x,x') = \Delta(x',x)$.

Let us prove the triangular inequality:
Let $(x_1,x_2,x_3) \in \mathcal{G}^3$

- First case: $n_1 = n_2 = n_3 = n$

$\Delta(x_1,x_2) = \sum_{i=1}^n \delta(a_i^1, a^2_{\mu(x_1,x_2)(i)})$
$\Delta(x_2,x_3) = \sum_{i=1}^n \delta(a_i^2, a^3_{\mu(x_2,x_3)(i)})$
$\Delta(x_1,x_3) = \sum_{i=1}^n \delta(a_i^1, a^3_{\mu(x_1,x_3)(i)})$

Consider the quantity
$\Delta_{aux}(x_1,x_3) = \sum_{i=1}^n \delta(a_i^1, a^3_{\mu(x_2,x_3)(\mu(x_1,x_2)(i))})$

Since δ is a distance on $\mathcal{A}$, obviously
$\Delta_{aux}(x_1,x_3) \leq \Delta(x_1,x_2) + \Delta(x_2,x_3)$

Besides since $\Delta(x_1,x_3)$ minimizes the sum of distances between pairs of matched alleles from x_1 and x_3, we get the desired inequality

$$\Delta(x_1,x_3) \leq \Delta_{aux}(x_1,x_3) \leq \Delta(x_1,x_2) + \Delta(x_2,x_3)$$

- General case :

The space $\mathcal{A}$ is completed with an element (noted 'w') such that
$\delta(w,w) = 0$, and
$\delta(a_i,w) = diam_\delta$
It easy to check that δ is still a distance on the completed space. Suppose each individual of $\mathcal{G}$ is completed with the element w up to $\max(n_1,n_2,n_3)$ elements. Then, using the first case result, we prove the triangular inequality on the completed space. This also proves the desired result. □

III.5.1.2 Distance on $\mathcal{A}$

Suppose the genotype space $\mathcal{G}$ is that of unordered lists of *alleles*, all alleles taken from a space $\mathcal{A}$. Moreover, suppose there exists a *similarity measure* Sim from $\mathcal{A}^2$ into $\mathbb{R}_+$ (i.e. satisfying $Sim(a,a) \geq Sim(a,b); \forall (a,b) \in \mathcal{A}^2$) and defining for all alleles (a,b) in $\mathcal{A}^2$ the quantities $|a-b|$ and δ by

$$\begin{array}{l} |a-b| = Sim(a,a) - Sim(a,b), \\ 2\delta(a,b) = |a-b| + |b-a| \end{array} \tag{173}$$

The following proposition gives some sufficient conditions on Sim to ensure that δ is a distance, i.e. that δ satisfies the triangular inequality.

Proposition III.5.2
Let $Sim : \mathcal{A}^2 \to \mathbb{R}_+$ a similarity measure on $\mathcal{A}$, and $|a-b| = Sim(a,a) - Sim(a,b)$.
If there exists g and h such that

$$Sim(a,b) = g(A_a, A_b)\, h(d(a,b)),$$

where d is a distance on $\mathcal{A}$, A_a (we should note $A(a)$) is a strictly positive function on $\mathcal{A}$, $h : [0,+\infty] \to [0,1]$ is a monotone decreasing function such that $h(0) = 1$ and

$$1 + h(d+d') - h(d) - h(d') \geq 0 \;\; \forall (d,d') \in \mathbb{R}_+^2$$

Finally $g : [0,\infty] \to \mathbb{R}_+$ satisfies the inequality

$$1 + g(\alpha,\beta) \geq g(\alpha,1) + g(1,\beta) \;\; \forall (\alpha,\beta) \in \mathbb{R}_+^2$$

as well as one of the two set of conditions below, $foreach$ $(s,s') \in \mathbb{R}_+^2$

$$\left\{ \begin{array}{l} g(s,s) = s \\ g(ks,ks') = k\, g(s,s')\, \forall k \in \mathbb{R}_+ \\ 0 \leq g(s,s') \leq min(s,s') \end{array} \right. \quad or \left\{ \begin{array}{l} g(s,s) = 1 \\ 0 \leq g(s,s') \leq 1 \end{array} \right.$$

then,

$$\delta(a,b) = (|a-b| + |b-a|)/2$$

is a distance on $\mathcal{A}$.

Remark: Note that this proposition already supposes that there exists a distance d on $\mathcal{A}$. But the proposed construction yields a number of different distances of which d is a particular case. See sections III.5.5 and III.5.6 for a comparison of distance performance with respect to evolution (ex. the area based (and *Hausdorff*) distances give very bad results).

Proof.
The proof is proposed in the particular case of $g(s,s) = s$. The arguments used in the other cases are similar.
1. δ is positive from the hypothesis of the proposition
2. $\delta(a_1,a_2) = 0$ iff $h(d(a_1,a_2)) = 1$ iff $d(a_1,a_2) = 0 \ldots$
3. To prove the triangular inequality, consider 3 alleles a_1, a_2, a_3. Then:
$\delta(a_1,a_2) = (A_{a_1} + A_{a_2})/2 - g(A_{a_1}, A_{a_2})h(d(a_1,a_2))$
$\delta(a_2,a_3) = (A_{a_2} + A_{a_3})/2 - g(A_{a_2}, A_{a_3})h(d(a_2,a_3))$
$\delta(a_1,a_3) = (A_{a_1} + A_{a_3})/2 - g(A_{a_1}, A_{a_3})h(d(a_1,a_3))$

Let's note $A_{a_1} = \alpha\, A_{a_2}$, $\quad A_{a_3} = \beta\, A_{a_2}$ and $d(a_1, a_2) \; = \; d_{12}$

$$\begin{aligned} T \quad &= \delta(a_1,a_2) + \delta(a_2,a_3) - \delta(a_1,a_3) \\ &= A_{a_2}\big(1 - g(\alpha,1)h(d_{12}) - g(1,\beta)h(d_{23}) + g(\alpha,\beta)h(d_{13})\big) \end{aligned}$$

Knowing that d is a metric distance, and that h is a decreasing function, we have $h(d(a_1,a_3)) \geq h(d(a_1,a_2) + d(a_2,a_3))$.
Combining this with the fact that $1 + g(\alpha,\beta) \geq g(\alpha,1)) + g(1,\beta)$, we obtain

$$TA_{a_2}^{-1} \geq 1 - g(\alpha,1)\big(h(d_{12}) - h(d_{12}+d_{23})\big) - g(1,\beta)\big(h(d_{23}) - h(d_{12}+d_{23})\big) - h(d_{12}+d_{23})$$

Since $g(\alpha,1) \leq 1$ and $g(1,\beta) \leq 1$, we get

$$TA_{a_2}^{-1} \geq 1 + h(d_{12}+d_{23}) - h(d_{12}) - h(d_{23}) \geq 0$$

$\square$

A direct consequence of this proposition is that

Lemma III.5.3
Let (a,b) be a couple of points of $\mathcal{A}$, then

$$\delta(a,b) = \frac{g(A_a,A_a) + g(A_b,A_b)}{2} - g(A_a,A_b)\, h(d(a,b))$$

is a distance on $\mathcal{A}$ if g and h are one of the following:

$$g(a,b) = \begin{cases} 1, \; or \\ min(A_a,A_b), \; or \\ 2\, min(A_a,A_b)/(A_a+A_b) \end{cases} \qquad and \qquad h(d) = \begin{cases} 1 - d/C, \; or \\ \exp(ß - \frac{d}{C}) \end{cases}$$

where $C \in \mathbb{R}_+$ is such that $h(d) \leq 1$.

This leads to six possible expressions for δ, in linear and exponential scales:

$$\delta(a,b) = \begin{cases} \dfrac{d(a,b)}{ßC} \\ \dfrac{A_a+A_b}{2} - min(A_a,A_b)(1 - \dfrac{ßd(a,b)}{ßC}) \\ 1 - \dfrac{2\, min(A_a,A_b)}{A_a+A_b}(1 - \dfrac{ßd(a,b)}{ßC}) \end{cases}$$

$$\delta(a,b) = \begin{cases} 1 - e^{\frac{ß - d(a,b)}{ßC}} \\ \dfrac{A_a+A_b}{2} - min(A_a,A_b)e^{\frac{ßd(a,b)}{ßC}} \\ 1 - \dfrac{2\, min(A_a,A_b)}{A_a+A_b}e^{\frac{ßd(a,b)}{ßC}} \end{cases}$$

In the following, A_a denotes the area of hole (resp. a Voronoï cell) a. If $g(a,a') = 1$, δ defines a distance on the space of alleles. Otherwise, δ defines a distance on the subspace of non degenerate alleles only (non void holes or Voronoï cells).

Analogy with Hamming distance Note that the analogy with Hamming distance (on ordered lists) is straightforward. A similarity Sim(a,b), that takes value 1 if both alleles are equal and 0 if they differ, yields a trivial allelic distance d_a taking value 0 if both alleles are equal, 1 if they differ. The Hamming distance between strings $x = (x_1, \ldots x_i \ldots)$ and $x' = (x'_1, \ldots x'_i \ldots)$ can be viewed as $\sum d_a(x_i, x'_i)$

One can also view δ as the generalization of d_a to the case of continuous parameters. Moreover, "allele-matching" allows us to overcome the lack of canonical ordering on these variable length representations. Finally, unmatched alleles are taken into account and increase the distance. These distances (defined by Equation 172) are called matching based distances.

III.5.1.3 Other distances on $\mathcal{G}$

Some guidelines have been given in propositions III.5.1 and III.5.2 for the construction of genotypic matching based distances on the space of variable length unordered lists of alleles: Given a genotypic distance on $\mathcal{A}$, a distance on $\mathcal{G}$ is simply the sum of the distances between matched alleles. However, different choices are available from the literature:

• The first one, the *Hausdorff* distance is also based on the choice of a distance on A, but no allele-matching is required:

***Hausdorff* distance:** *Given d a distance on E, let $d(a, B) = min\{d(a, b), b \in B\}$ where $a \in E$, $B \subset E^n$. The* Hausdorff *distance between two subsets of E, $A \subset E^{n'}$, $B \subset E^n$ is:*

$$\Theta(A, B) = max\{d(a, B) : a \in A\} + max\{d(b, A) : b \in B\}$$

Of course this is a distance between sets, and not lists. Voronoï diagrams naturally fit in this context, but not the Holes representations. From a practical point of view, it's rather unlikely that the same –real valued– hole appears twice in the same (bounded size) list.

• The second possible distance, the *Levenstein* distance, is usually used to compare variable length ordered lists (as strings). Hence, it requires a definition of some pre-ordering criterion on each list before it can be applied:

***Levenstein* distance:** *The Levenstein distance between two lists of characters is the minimal cost calculated as the sum of the costs of the elementary operations of substitution $f(a, b)$, of insertion $f(\emptyset, a)$ and destruction $f(a, \emptyset)$ of the characters.*
For each choice of the function f, the algorithm of Wagner-Fisher's determines this cost, as well as the correspond suite of operations.
Consider two strings $x = a1a2...an$ et $y = b1b2...bm$, we note xi the prefix $a1a2...ai$ and $d(i, j)$ the distance between xi and yj.

$d(0, 0) := 0$

For $i = 1 \ldots n$ *do*
  $d(i, 0) := d(i-1, 0) + f(ai, \emptyset)$

For $j = 1 \ldots m$ *do*
  $d(0, j) := d(0, j-1) + f(\emptyset, bj)$

For $i = 1 \dots n$ *and* $j = 1 \dots m$ *do*
$m1 := d(i-1, j-1) + f(ai, bj)$
$m2 := d(i-1, j) + f(ai, \emptyset)$
$m3 := d(i, j-1) + f(\emptyset, bj)$
$d(i, j) := min(m1, m2, m3)$

The distance between the strings x *and* y *is then* $d(n, m)$.

To use this distance on the Holes or Voronoï representations, an arbitrary ordering has to be defined (coordinate sorting for example ...). This defines as many distances as there are choices for such an ordering. Also, different choices for the function f, and the elementary operations are available.

A mostly interesting direction would be to study the connection between the *Levenstein* distance and the evolution process (see. comparison tools designed in next subsection), especially when GA operators are used as the elementary operations of the *Levenstein* distance. But this is beyond the scope of this work.

In the following, we stick to comparing the matching and *Hausdorff* distances, for different choices of a distance on $\mathcal{A}$.

III.5.2 GENOTYPIC DISTANCES FOR HOLES REPRESENTATIONS

This subsection instantiates the general ideas introduced in subsection III.5.1 on a variable length holes representation, in the case of the exponential scale (cf. lemma III.5.3).

Distances for the *Holes* representations can be designed regardless of the actual possible elementary shapes for the holes. The only prerequisite for designing such a similarity measure following proposition III.5.2 is the choice of a distance d on A. In the following, we consider d to be $L1$ distance between holes: $\delta_{L1}(R, R') = |x - x'| + |y - y'| + |l - l'| + |L - L'|$.

III.5.2.1 Hole based distance

Following the general guidelines of Subsection III.5.1, a distance of the genotype space can be defined from any similarity measure between holes of proposition III.5.2.
A first kind of such similarity measures, is to consider $Sim(R, R) = 1$ for each hole R regardless its size, and

$$Sim_H(R, S) = e^{\frac{ß - d(R,S)}{ßC}}, \tag{174}$$

$$Sim_{HA}(R, S) = \frac{2\min(A(R), A(S))}{A(R) + A(S)} e^{\frac{ß - d(R,S)}{ßC}}, \tag{175}$$

where $d(R, S)$ is the *Hausdorff* distance between the holes R and S, and C is a scaling parameter computed such that the quantity $e^{-d(R,S)/C}$ ranges from 1 (when $R = S$) to almost 0 (when d(R,S) is maximal in the design domain). Practically, a value of $C = d_{max}/6$ has been used, so that $e^{-d(R,S)/C}$ ranges approximately in $[0.0025, 1]$. However, experiments show that the value of C has no major influence on the numerical results presented in the following, and no influence concerning the comparative conclusions between distances.

Following now the general ideas of Section III.5.1, a distance can be designed, using Equations 173 to define a distance δ_A over the space of holes from the similarity measure given above and Equation 172 to define the distance Δ_A over the genotype space.

$$\Delta_H(x,x') = n - \sum_{i=1}^{n'} e^{\frac{ß - d(R_i, R'_{\mu(i)})}{ßC}} \tag{176}$$

$$\Delta_{HA}(x,x') = \quad n - \textstyle\sum_{i=1}^{n'} \frac{2\min(A(R_i),A(R'_{\mu(i)}))}{A(R_i)+A(R'_{\mu(i)})} e^{\frac{ß - d(R_i, R'_{\mu(i)})}{ßC}} \tag{177}$$

where $\mu = \mu(x,x')$ is the *heuristical best matching* between x and x' computed as in Section III.5.4.1, A_{max} is the maximal area of a hole in the design domain.

III.5.2.2 Area based distance

Another approach to defining a similarity measure over the space of holes is to define $Sim(R,R)$ as $A(R)$, the area of hole R, and

$$Sim_A(R,S) = \min(A(R),A(S)) e^{\frac{ß - d(R,S)}{ßC}}, \tag{178}$$

where $d(R,S)$ is again the *Hausdorff* distance between the holes R and S, and C is a scaling parameter computed as explained in the preceding section for the holes based distance. Following the same process than in the section III.5.1, equation (172) leads to defining the genotypic distance between $x = (R_i)_{i\in[1,n]}$ and $x' = (R'_i)_{i\in[1,n']}$, with $n > n'$ in the genotypic space as:

$$\begin{aligned} 2\Delta_A(x,x') = \quad & \sum_1^n A(R_i) + \sum_1^{n'} A(R'_i) + \sum_{n'+1}^{n} A_{max} \\ & -2\left(\textstyle\sum_{i=1}^{n'} \min(A(R_i), A(R'_{\mu(i)})) e^{\frac{ß - d(R_i, R'_{\mu(i)})}{ßC}} \right) \end{aligned} \tag{179}$$

where $\mu = \mu(x,x')$ is the *heuristical best matching* between x and x' computed as in Section III.5.4.1, A_{max} is the maximal area of a hole in the design domain.

Remark: The first idea in designing a similarity measure was to define $Sim(R,S)$ for two holes R and S as $A(R\cap S)$, $A(X)$ denoting the area of subset X of $\mathbb{R}^2$.
However, such a definition leads to undesirable results whenever the two holes have an empty intersection: the similarity is the same, whatever the distance between the two disjoint holes. That is why the early definition (not used) was:

$$Sim(R,S) = \min(A(R),A(S)) e^{\frac{ß - d(R,S)}{ßC(R,S)}}, \tag{180}$$

where $C(R,S)$ is a scaling parameter computed such that the quantity $e^{-d(R,S)/C(R,S)}$ for two identical holes that are exactly half way overlapping is equal to 0.5. This is achieved by the formula $C(R,S) = (l_R + l_S)/(4 * ln(2))$, where l_R is a characteristic dimension of hole R (alternatively its length or width for a rectangle, or a more complex formula for triangles) chosen according to its shape and position relatively to S.

With a variable C, δ is no longer a distance, which is why it was abandoned. Nevertheless, the metric violations are rather negligible: using the measures defined in subsection III.5.4.1 over a sample of 1000 random rectangles, the percentage ρ of violating examples for δ is 0.04% while the average of violations $\bar{\omega}$ is 4(4)%.

III.5.2.3 A compromise between the genotypic and phenotypic distances

Suppose the number of alleles of an individual x' is higher than that of x. Once each allele of x have been matched with an allele of x', there remains a number of unmatched alleles in x'. According to the definition of Eq. 172, the distance $\Delta(x,x')$ is increased by a fixed amount of δ_{max} for each unmatched allele. Which results in a genotypic distance directly dependent on the number of alleles.

A different and intuitive approach to measure genotypes similarities is to focus on a compromise between genotypic and phenotypic resemblance: the difference between two individuals that have similar phenotypes, shouldn't depend on their respective alleles number, but on the differences between these alleles.

This is achieved by the following definition, where an unmatched allele of x' increases the distance according to its difference from alleles of x. Hence, if an unmatched allele of x' is equal to some –previously matched– allele of x, it doesn't increase the genotypic distance. More precisely, we define a candidate 'dissimilarity' measure as:

$$\Delta^{\varphi}(x,x') = \sum_{1}^{n} \delta(a_i, a'_{\mu(x,x')(i)}) + \sum_{\mu(x,x')^{-1}(i)=\emptyset} \min_{j\in[1,n]} |a'_i - a_j|, \tag{181}$$

where $\mu(x,x')$ is the matching that minimizes the expression of $\Delta^{\varphi}(x,x')$.

Note that Δ^{φ} is only a distance over the space of fixed length lists of holes. However, numerical results show that it differs from a distance by very small amounts: The numerical test regarding the triangular inequality for Δ^{φ} gave the following results: over a sample of 1000 random individuals, the percentage ρ of violating examples is 0.06% while the average of violations $\bar{\omega}$ is 4(5)%.

This intuitive and non-metric 'dissimilarity' measure (eq. 181) demonstrates that it is as good as the effective corresponding (same choice of Sim) distance (eq. 172) with respect to most criteria cited above III.5.4. Three major differences have been found:

- Δ^{φ} is much more robust to degeneracy than the corresponding Δ distance. This is coherent with the fact that unmatched holes (in Δ^{φ}) increase the distance according to their position.
- To compare initialization procedures, Δ^{φ} gives more useful information, as it focuses on genotypic diversity within a given phenotype, whereas the Δ distances directly depend on the genotype length.

- Optimizing the *distance fitness* functions, convergence time and solution quality results are largely below those obtained with the corresponding Δ distances.

This dissimilarity measure will be used (in addition to distance Δ) for some experiments in later sections. Further hints for the usefulness of this distance can be found in the discussion III.7, where we show a strong correlation between the Holes based Δ^{φ} distance and the minimal number of discrete elementary mutation applications that links two points of the space.

III.5.3 GENOTYPIC DISTANCES FOR VORONOI REPRESENTATIONS

This subsection instantiates the general ideas introduced in subsection III.5.1 on the Voronoï representations, in the case of an exponential function h in lemma III.5.3.

Here again, the definition of a distance on the Voronoï genotypes follows the ideas in section III.5.1. But different points of view are available: From the purely genotypic point of view, an individual is completely defined by the list of Voronoï sites. However, considering only the list of sites does not bring much information about the resulting partition of the domain, and it might be useful to also consider the genotype as a list of Voronoï cells, hence involving cells in the distance. Along these lines, four different distances can be designed.

In all that follows, the Euclidean distance of $\mathbb{R}^2$ is denoted d, V denotes a Voronoï site, $l(V)$ is its label (0 or 1), and $A(V)$ the area of the associated cell (in the individual at hand). Note that the associated cell depends on the other sites' positions, hence we should note $A(V,(V_i)_i)$.

III.5.3.1 Site based distances

A first approach is to defining similarity measure is to consider the genotype as a list of sites, and define:

$$\begin{aligned} Sim_S(V_1,V_2) &= e^{\frac{ß-d(V_1,V_2)}{ßC}} && \text{if } l(V_1)=l(V_2), \\ &= 0 && \text{otherwise,} \end{aligned} \tag{182}$$

$$\begin{aligned} Sim_{SC}(V_1,V_2) &= \frac{2\min(A(V_1),A(V_2))}{A(V_1)+A(V_2)} e^{\frac{ß-d(V_1,V_2)}{ßC}} \text{ if } l(V_1)=l(V_2), \\ &= 0 \text{ otherwise.} \end{aligned} \tag{183}$$

where C is a scaling parameter set as in III.5.2.1, and d is the Euclidean distance between sites. Note that $Sim_S(V,V) = Sim_{SC}(V,V) = 1$ for any site V.

Following the guidelines of Section III.5.1, namely Equations 173 and 172, the Site-distance Δ_S between $x=(V_i)_{i\in[1,n]}$ and $x'=(V'_i)_{i\in[1,n']}$ is constructed as:

$$\Delta_S(x,x') = max(n,n') - \sum_{1}^{min(n,n')} e^{\frac{ß-d(V_i,V'_{\mu(i)})}{ßC}} \tag{184}$$

This variant of the distance is named the Site based distance and noted as Δ_S.
The second similarity case (Sim_{SC}), takes into account more than the site itself, as the area of the cells is introduced in the similarity measure itself. The resulting distance on the genotype space is called the Site-Cell distance: for that distance.

$$
\begin{array}{cccc}
\begin{array}{c} 0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0 \\ 0\,1\,1\,1\,0\,0 \\ 0\,1\,1\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0 \end{array} &
\begin{array}{c} 2\,2\,2\,2\,2\,2 \\ 1\,1\,1\,1\,1\,2 \\ 1\,0\,0\,0\,1\,2 \\ 1\,0\,0\,0\,1\,2 \\ 1\,1\,1\,1\,1\,2 \\ 2\,2\,2\,2\,2\,2 \\ 3\,3\,3\,3\,3\,3 \end{array} &
\begin{array}{c} 0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,1\,1\,1 \\ 0\,0\,0\,0\,1\,1 \\ 0\,0\,0\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0 \end{array} &
\begin{array}{c} 0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,2 \\ 0\,0\,0\,0\,1\,2 \\ 0\,0\,0\,0\,1\,2 \\ 0\,0\,0\,0\,0\,2 \\ 0\,0\,0\,0\,0\,0 \end{array} \\
I_1 & DI_1 & I_2 & |I_2 - I_1| = 11/10
\end{array}
$$

FIG 56 : *The $DT1$-distance, based on the 1/1 Distance Transformation*

III.5.3.2 Cell based distances

Going further from pure genotypic distance toward phenotypic distance, a second approach is to consider the genotype as a list of cells.
Then, the matching between "alleles" requires the definition of a distance between cells. The DT distance has been preferred to the Hausdorff distance in many image processing application [Bor84], and its definition is recalled here.

Symmetric DT-distance:

Distance Transformation : The distance transformation (DT) has been defined to measure the distance between features in black and white images. It converts a binary picture consisting of feature and non feature elements (e.g. black and white parts), into an image where each element has a value that approximates the distance to the nearest feature element. Good integer approximations of Euclidean distance can be computed by the process known as Shamfer distance. All details about distance transformation can be found in [Bor84].

DT-Distance: Given two binary images I_1 and I_2, their feature dissimilarity is defined the following way: first, I_1 (Figure 56-a) is converted by the Shamfer 1/1 DT described above into distance image DI_1 (Figure 56-b). The dissimilarity $|I_1 - I_2|$ is then the average of the distances to features of I_1 of all feature-points of I_2, given by the corresponding value of the distance image DI_1 (Figure 56-d). The dissimilarity $|I_1 - I_2|$ is zero as soon as the features of I_2 are contained in the features of I_1.
Finally, the $DT1$-distance between images is defined by:
$d_{DT1}(I_1, I_2) = 0.5(|I_1 - I_2| + |I_2 - I_1|)$.

Symmetric DT-distance: The distance between bitarrays can be computed using the DT concept if both the ones and the zeros are considered in turn as "features" for the DT. The sum of dissimilarities between the zeros and the ones of both bitarrays is a distance over the genotype space of bitarrays:

$$\Delta_{DT}(B_1, B_2) = d_{DT1}(B_1, B_2) + d_{DT0}(B_1, B_2). \tag{185}$$

The cell based distances:

The similarity measure between two cells can now be defined in terms of the DT-distances between the corresponding cells, resulting in:

$$
\begin{aligned}
Sim_C(V_1, V_2) &= e^{\frac{\text{ß} - d_{DT_l}(C_1, C_2)}{\text{ß}C}} && \text{if } l(V_1) = l(V_2), \\
&= 0 && \text{otherwise,}
\end{aligned} \tag{186}
$$

where C_i is the cell corresponding to V_i. The similarity $Sim_{CS}(V,V)$ is 1, whatever is the area of the corresponding cell. The resulting genotypic distance is termed Cell based distance and denoted Δ_C.

$$\begin{aligned} Sim_A(V_1,V_2) &= \min(A(V_1),A(V_2))e^{\frac{ß-d_{DT_l}(C_1,C_2)}{ßC}} && \text{if } l(V_1)=l(V_2), \\ &= 0 && \text{otherwise,} \end{aligned} \tag{187}$$

The similarity $Sim_A(V,V)$ is now the area of the cell.
Using Equations 173 and 172, with the respective similarity measures, lead to the Area and Cell-distances Δ_A and Δ_C:

$$\Delta_A(x,x') = A(\Omega) - \sum_{1}^{min(n,n')} \min(A(C_i),A(C'_{\mu(i)}))e^{\frac{ß-d_{DT}(C_i,C_\mu(i)')}{ßC}} \tag{188}$$

where $A(\Omega)$ is the area of the whole domain, sum of areas of all cells of an individual.

$$\Delta_C(x,x') = max(n,n') - \sum_{1}^{min(n,n')} e^{\frac{ß-d_{DT}(C_i,C_\mu(i)')}{ßC}} \tag{189}$$

However, a major drawback of the Cell-distances is the computational cost of all DT-distances. Note that the *Hausdorff* distance between cells would be just as expensive to compute.

III.5.4 DISTANCE VALIDATION AND COMPARISON TOOLS

An important issue after a distance has been designed, and before using it to analyze the behavior of an Evolutionary Algorithm, is to ensure, first, that numerical implementation is possible (time consideration), and second, that there is indeed some link between the distance and the operators at hand (crossover, mutation). Different measures are proposed to characterize the adequation between the genotypic distances and the genetic algorithm. These measures are used in the next sections (III.5.5 and III.5.6) to select the distances that are most suitable for studying the fitness landscapes on the variable length representations at hand.
Note that this choice ensures that there is a relation between the distance and the operators of the algorithm at hand, but is independent from the inclusion problem under study. On the other hand, a different algorithm, may yield a different distance choice.

III.5.4.1 Heuristic Matching

Finding the best matching to compute Δ from Eq. (172) is itself a NP problem, and cannot be solved exactly. To cope with this difficulty, some heuristic matching has been designed. To approximate the best matching between two lists of alleles $(a_i)_{i\in[1,n]}$ and $(a'_i)_{i\in[1,n']}$, with $n \leq n'$, the following deterministic algorithm was used:

1. Set a high matching level $L \in [0,1]$ (typically 0.90).
2. For all alleles a_{i1} in x yet unmatched
   find the closest allele a'_{i2} in x' (for distance δ),
   match the pair if $Sim(a_{i1},a'_{i2}) > LSim(a_{i1},a_{i1})$;
   otherwise, match the pair if $\delta(a_{i1},a'_{i2}) = \min_{a_k unmatched}\delta(a_k,a'_{i2})$;
   otherwise a_{i1} stays unmatched.

3. If all alleles a_i are paired, Stop
   otherwise, decrease L (typically by subtracting 10%) and goto 2.

The complexity of this algorithm is in $n \times n'$, as it requires the computation of all similarities between pairs of alleles. Experimental tests introduced below enforce the presumption that replacing the exact best matching by the above heuristic matching does not significantly degrade the utility of the distance at hand (although this heuristic matching does not ensure that $d(x, x') = d(x', x)$). Hence, all experiments throughout this work will use heuristic matching to approximate the best matching (Proposition III.5.1) for computing the distance between individuals in variable length genotype spaces.

Violations of the triangular inequality: In the actual implementations of the distance Δ defined in Equation 172, the best matching is approximated (III.5.4.1 above). Hence the distances actually used for the representations defined in Section III.3 may violate the triangular inequality.
In order to nevertheless justify their use, and as a first attempt to compare their respective quality, the amounts of violation were experimentally computed the following way:

1. Generate a random sample of individuals
2. Take all t possible triples of individuals;
3. For each triple, and their three possible orderings A_1, A_2, A_3, compute the "violation of triangular inequality"
   $$\omega(A_1, A_2, A_3) = \delta(A_1, A_2) + \delta(A_2, A_3) - \delta(A_1, A_3);$$
4. Report the percentage ρ of strictly positive violating examples of (unordered) triples over all possible unordered ones, and the average $\bar{\omega}$ (and standard deviation $\sigma_{\bar{\omega}}$) over those examples of the relative amounts of violation $\frac{\omega(A_1,A_2,A_3)}{\delta(A_1,A_3)}$.

Numerical results show no violation over a sample of 1000 random individuals with some instances of the Δ distance.

III.5.4.2 Distances and evolution

For any optimization algorithm, the adequacy of the distance used to perform any landscape analysis is not formulated: This subsection tries to make these assumptions explicit, and derives some measures (detailed below) of adequacy of a distance toward evolution operators. These adequacy tools are mainly motivated from the analogy with the *Hamming* distance properties, in the binary framework.
These measures will be used in the next sections ((III.5.5 and III.5.6)) to compare the genotypic distances, in order to select the most suitable ones for landscape analysis. For instance, if the distance is unconnected to the algorithm at hand, it cannot capture useful [55] information from the fitness landscape.

A mostly interesting method for studying the adequacy of a genotypic distance with some algorithm, is to consider algorithm's behavior on *distance fitness* [56] functions:

[55] to improve or understand the optimization process
[56] The fitness of each point is proportional to its distance to solution (see section II.2, where it is used as a basis to *a posteriori* compare GA behavior in the bitstring framework)

1. Distance landscapes: In order to choose an adequate genotypic distance on these new spaces, we propose to consider GA performance on each *genotypic distance fitness* as a measure of the distance's adequacy to describe the similarity in the genotypic space (note that in the binary space, the *Hamming fitness* is unimodal, and straightforward to optimize for a simple 1-bit-flip hill climber).

This criterion has been found very instructive: experiments of section III.5.5 show that a population based GA was unable to optimize some *Hausdorff fitness* functions, whereas it succeeded in finding the optimum genotype when a matching based distance is used for the fitness (in this latter case, even a simple (1+1)-ES scheme was successful).

2. Phenotypic-Genotypic correlation: Given a point $s \in \mathcal{G}$ in the search space, we compute the correlation between the genotypic distance to s and the phenotypic distance to s, on a random sample of points of $\mathcal{G}$.
Forthcoming results exhibit important differences in genotypic diversity (on the x axis) according to the distance at hand.

3. Distances and mutation: In order to check for the strong causality [57] between the mutation operator μ (displacement or modification of an allele) and some genotypic distance, a random sample of N individuals is generated, and each genotype undergoes one minimal mutation μ (defined in sections III.3.2 and III.3.3).
δ_{Mut} (σ_{Mut}) is the mean distance (standard deviation) between parents in the sample and their offspring (in the bitstring case, for the bit-flip minimal mutation, $\delta_{Mut} = 0$.

$$\delta_{Mut} = \frac{1}{N} \sum_{i=1}^{N} d(x, \mu(x)) \tag{190}$$

4. Distance and Crossover: In the bitstring framework, and whatever the crossover operator used, an offspring always lies on the "Hamming line"' between his parents: If o is an offspring of parents p and p', then $d_H(p,p') = d_H(p,o) + d_H(p',o)$. Hence with the notations below, we have $\tau_{PO} = 1$ and $\delta_{PO} = 0(0)$.

In order to study the connection of a distance to some crossover operator, a random sample of N individuals is generated, and N couples in that sample are randomly mated. The following coefficients are then computed:

1. $\delta_{PO}(\sigma_{PO})$, the expected difference (standard deviation) of parents and offspring distances.

$$\delta_{PO} = p_i, o_i)d + (p'_i, o_i) - d(p_i, p'_i)|, \tag{191}$$

   where p_i and p'_i are two parents randomly chosen in the sample, and mated once to generate o_i.

2. τ_{PO}, the correlation coefficient between the set of parents distances ($dp_i = d(p_i, p'_i)$) and the set of offspring distances ($do_i = d(p_i, o_i)d + (p'_i, o_i)$).

[57] In Real space, Evolution Strategies have long emphasized the need for *strong causality* for the mutation operator [Rec73]: A small modification of the genotype yield a small change in fitness

$$\tau_{PO} = \frac{\frac{1}{N}\sum_{i=1}^{N}(dp_i - \bar{dp})(do_i - \bar{do})}{\sigma_{dp}\sigma_{do}}, \tag{192}$$

where $\bar{dp}, \bar{do}, \sigma_{dp}$ and σ_{do} are respectively the means and standard deviations of parent distances and the offspring distances over the sample.

5. Distances and degeneracy: Another point of view, that does not appear at all in the bitstring framework, is the relation between the genotypic distance and degeneracy. As quoted in sections III.3.2 and III.3.3, the variable length representation considered in this chapter are highly degenerate. Hence, different genotypes are available for the same phenotype, and the choice of the genotypes highly influences the results of any landscape analysis on these representations. Therefore, before using genotypic distances on such spaces, it is important to study the influence of degeneracy on the distance values.

To address this problem, consider two different genotypes g_1 and g_2 for the same phenotype, and a sample of N individuals. Compute the two sets of distances between all sample individuals and g_1 and g_2 respectively. The correlation coefficient between these two sets is termed Distance-Distance Correlation coefficient (τ_{DDC}).

$$\tau_{DDC} = \frac{\frac{1}{N}\sum_{i=1}^{N}(d1_i - \bar{d}1)(d2_i - \bar{d}2)}{\sigma_{d1}\sigma_{d2}}, \tag{193}$$

where $\bar{d}1, \bar{d}2, \sigma_{d1}$ and σ_{d2} are respectively the means and standard deviations of distances to genotype g_1 and and distances to genotype g_2 over the sample.
This coefficient, together with the scatter plot of all points of the two sets defined above, gives an idea of how much the degeneracy distorts the search space.

In the following, different genotypic distances are compared with each of the criteria presented above, and Table 11 gives a summary of the results. This should allow us to choose the distances that will be used in section III.6 to perform fitness landscape analysis for the inclusion identification problem of Section III.2.

III.5.5 DISTANCE COMPARISON FOR RECTANGLE REPRESENTATION

In the following, each of the criteria in subsection III.5.4 are used to compare the matching based (denoted Δ) and *Hausdorff* genotypic distances (denoted Θ).

Recall that the distance definition for both the matching based distances and *Hausdorff* distances, relies on the choice of the distance δ on the space of alleles, which directly depends on the choice of the similarity measure Sim between two alleles a and a' (two rectangles here), as in lemma III.5.3: $Sim(a,a') = g(a,a')\,h(d(a,a'))$.

In the following, we focus on comparing two of the three similarity measures: the hole based ($Sim_H(a,a) = g_H(a,a) = 1$), and the area based ($Sim_A(a,a) = A(a)$), typically the third case of $Sim_{HA}(a,a') = 2\,min(A_a, A_{a'})/(A_a + A_{a'})$ gives similar (often slightly worse) results than the Sim_H case.

Moreover, few differences have been observed between the linear and exponential scaling of function h: typically, linear scaled distances give similar or slightly worse results than the respective corresponding exponentially scaled distances (with an exception in the case of the *Hausdorff* Θ_A distance, where the linear distance landscapes are easier to optimize than the exponentially scaled ones). Hence, we mostly stick to comparing distances in the exponential scaling case $h(d) = e^{-d/C}$ (with an exception for distance landscapes criterion in Fig. 58).

A summary table that puts together the results obtained with all the criteria is presented at the end of this subsection (Tab. 11). For each of the 5 measures detailed below, it gives the percentages of *error* (the difference from the results one would obtain in the binary space) relative to the range of the measure.

1. Distance landscapes As discussed in section III.5.4, we consider GA performance on *distance fitness* functions as a measure of the 'quality' of the distance at hand.

Four *distance fitness* functions are considered, where the solution phenotype is respectively that of an asymmetric, symmetric, corner and checker (Fig. 46). Note that the asymmetric genotype contains one rectangle only, and cannot be used to compare the distances [58]. It is used here to illustrate (and tune) GA results on a straightforward function case with these variable length representations.
All experiments use a 100 population size GA at 0.5 mutation and crossover rates, and a stopping criterion of 30 generations with no improvement.

GA results are summarized in tables 9 in the case of exponentially scaled distances. Note that all GA runs stopped after the stopping criterion was satisfied. But increasing the number of generations of the stopping criterion yields the same conclusions: The *Hausdorff* landscape contains some strong traps, from which the GA couldn't escape within a reasonable time (30 generations).
Clearly, the quality of the GA results is heavily dependent on the choice of the distance, as shown in Figure 57, where the best individual found by the GA is plotted. Results quality can be ordered from best to worse: Δ_H, Δ_A, then the *Hausdorff* distances Θ_H and Θ_A.

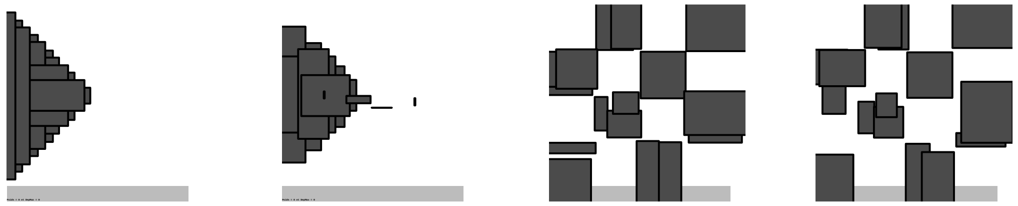
A noticeable feature is that *Hausdorff* distance between the best individual found by the GA and the solution is sometimes very low (Table 9). Yet the phenotypes and number of rectangles –of GA best individual– are quite different from those of the solution.
Hence, there is a number of 'traps' in the *Hausdorff* landscape. Note that, in the corner case (Fig. 57), the best individual found by the GA contains 'middle' rectangles that replace a number of overlapping rectangles in the solution genotype. On the other hand, the inverse happens for the linear Θ^l_H *distance fitness* to a checker (Fig. 58): The best individual found by the GA contains more (overlapping) rectangles than the actual solution.
One can make a classification of the distances as follows :
Δ_H ¿ Δ_{HA} ¿ Δ_A ¿ Θ_H ¿ Θ_{HA} ¿¿ Θ_A

[58] In fact with a genotype of one allele only, the *Hausdorff* distance sums up to: $\Theta(a,x) = min_i(\delta(a,a'_i)) + max_i(\delta(a,a'_i))$, which is almost the sum of two δ distance functions on the space of alleles (rather than lists of alleles).

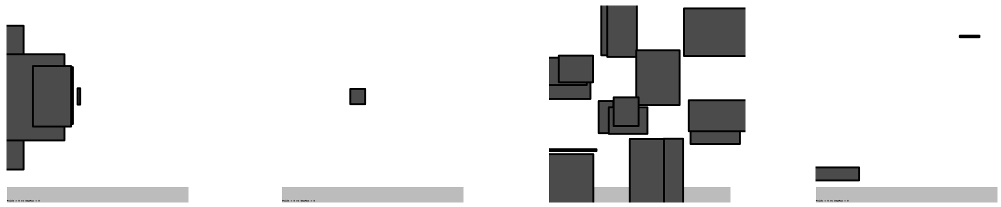
	dissy^{1}	**sym**^{6}	**coin**^{11}	**damier**^{16}
$\Delta_{\mathbf{H}}$	0.001 (0.001)	0.04 (0.01)	0.13(0.02)	0.2(0.02)
Ham	0(0)	1.4(2.2)	0(0)	8.7(4.5)
N_c	181(51)	550(116)	818(65)	671(13)
$\Delta_{\mathbf{A}}$	0.08(0.04)	9.2(3.9)	70(28)	52(21)
Ham	0.9(1.6)	4.5(5)	12(5)	62(20)
N_c	173(83)	638(123)	638(124)	682(9)
$\Theta_{\mathbf{H}}$	0.003(0.001)	0.1(0.1)	0.4(0.1)	0.2(0.03)
Ham	0.2(0.8)	15(41)	**33(17)**	**72(14)**
N_c	183(48)	360(111)	**291(83)**$^{(5,11)}$	**610(82)**$^{(16,15)}$
$\Theta_{\mathbf{A}}$	0.2(0.1)	28(24)	62(11)	35(6)
Ham	1.2(1.8)	**41(58)**	**103(31)**	**258(31)**
N_c	194(60)	**404(175)**$^{(7,6)}$	**194(103)**$^{(1,2)}$	**233(180)**$^{(2,18)}$

Table 9: *Average (standard deviation) GA results on different distance fitness functions, over 21 runs. First line gives the fitness (genotypic distance to optimum) of the best individual found by the GA, the second line (Ham), its phenotypic distance to optimum , and the third line (N_c) gives the number of generations to convergence. The number of rectangles found by the (median,best) runs is given as an exponent, when different from the optimum. Bold fonts indicate* **failure cases**, *where both the number of rectangles and the phenotype found by the GA are false.*

corner optimum : 11 rectangles   checker optimum :16 rectangles

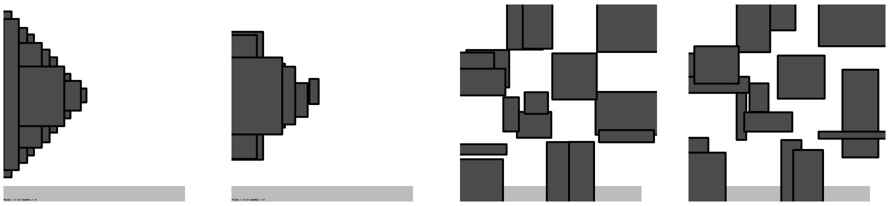


$\Delta_{\mathbf{H}} = \mathbf{0.13}$, 11 rect. $\Delta_{\mathbf{A}} = \mathbf{78}$, 11 rect. $\Delta_{\mathbf{H}} = \mathbf{0.2}$, 16 rect. $\Delta_{\mathbf{A}} = \mathbf{48}$, 16 rect.

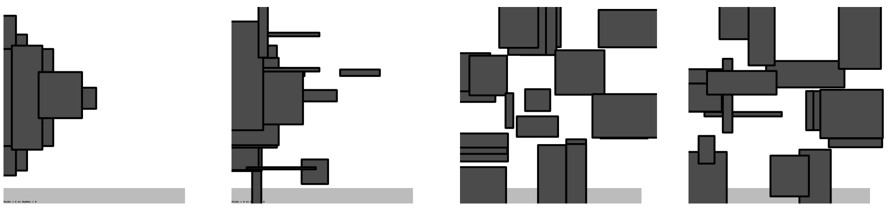


$\Theta_{\mathbf{H}} = \mathbf{0.3}$, 5 rect. $\Theta_{\mathbf{A}} = \mathbf{71}$, 5 rect. $\Theta_{\mathbf{H}} = \mathbf{0.2}$, 16 rect. $\Theta_{\mathbf{A}} = \mathbf{36}$, 2 rect.

FIG 57 : *Best individual (of median run out of 21) found by the GA on different* distance functions, *at* **exponential scales***: GA results with the Hausdorff distance functions are lousy and unstable (the best checker runs find genotypes at 15 (Θ_H) and 18 (Θ_A) rectangles).*



$\Delta^{\mathbf{l}}_{\mathbf{H}}$ $\Delta^{\mathbf{l}}_{\mathbf{A}}$ $\Delta^{\mathbf{l}}_{\mathbf{H}}$ $\Delta^{\mathbf{l}}_{\mathbf{A}}$



$\Theta^{\mathbf{l}}_{\mathbf{H}}$ $\Theta^{\mathbf{l}}_{\mathbf{A}}$ $\Theta^{\mathbf{l}}_{\mathbf{H}}$ $\Theta^{\mathbf{l}}_{\mathbf{A}}$

FIG 58 : *Best individual (of median run out of 21) found by the GA on different* distance functions, *at* **linear scales***: The GA don't find the optimum of the Hausdorff distance functions.*

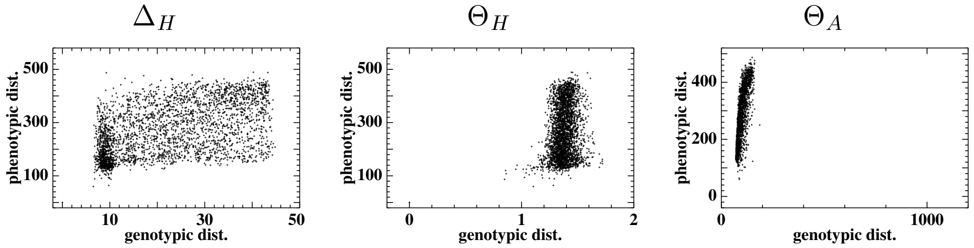


FIG 59 : *Correlation between genotypic and phenotypic distances to the corner. Area based distance Δ_A is omitted because it gives similar plots to those of Δ_H. Note the difference in diversity on the x-axis between the distances.*

Mutation	$\mathbf{\Delta_H}$	$\mathbf{\Delta_A}$	$\mathbf{\Theta_H}$	$\mathbf{\Theta_A}$
add/delete	1(0.09)	571(53)	0.4(0.2)	39(55)
resize/move	0.09(0.08)	5(9)	0.2(0.1)	7(11)
Dist. range	50	50*576	2	2*576

Table 10: **Average (standard deviation) distance between an individual and its mutated offspring** *(as defined in III.5.4), over a 3000 random sample. The* Hausdorff *distances are much more sensitive to mutation than the matching distances are, especially when considered relatively to their distance range.*

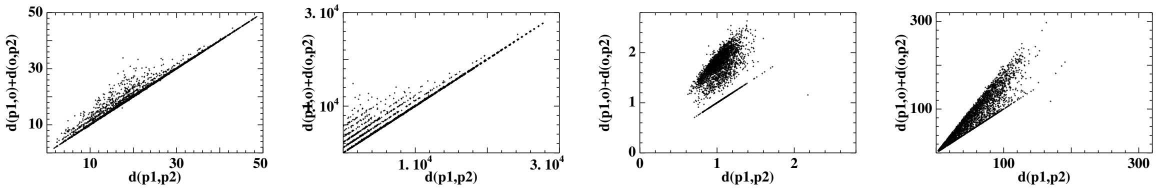
2. Phenotypic-Genotypic correlation The main motivation here is to compare the diversity of random samples with each genotypic distance. Figure 59 shows the correlation between the genotypic and the phenotypic distances to a genotype of a corner (that of Fig. 57).
The matching distances provide a much wider plot on the x-axis (relative to distance range) than the *Hausdorff* distances. Note also that GA results with the Θ_A *Hausdorff* fitness are particularly bad (Fig. 57).

3. Distance and mutation Table 10 shows the average distance between an individual and its mutated offspring over a 3000 random sample, as defined in Sect. III.5.4.2 Eq. (190). The *Hausdorff* distances are much more sensitive to mutation than the matching distances, especially when considered relative to their distance range. Note also the high standard deviations (relative to the averages) obtained with the *Hausdorff* distances.

4. Distance and crossover We compute the correlation τ_{PO} and difference δ_{PO} between parents and offspring as defined in III.5.4.2 (Eqs 191 and 192). Figure 60 shows that :

1- The best results are obtained with the Δ_H distance (τ_{PO} close to 1 and δ_{PO} close to 0). However the Δ_A results stay at a reasonable order relative to distance range.

2- The Θ_H *Hausdorff* distance gives particularly low τ_{PO} values (ex. 0.47), and high δ_{PO} values relative to distance range. On the other hand, the Θ_A distance is a lot better than the



	Δ_H	Δ_A	Θ_H	Θ_A
$\tau_{PO}=$	0.99	0.98	0.47	0.92
$\delta_{PO}=$	0.7(1.5)	476(1192)	0.7(0.3)	28(23)

FIG 60 : *Parent Offspring Correlation τ_{PO}, and difference δ_{PO} (cf. subsect. III.5.4), over a 3000 random sample. Note the particularly bad results of the Θ_H distance.*

Θ_H with respect to these measures.

Recall the particularly bad GA results on the Θ_A *Hausdorff distance fitness* (Fig. 57). This implies that the property of good parents offspring correlation (τ_{PO}) for the crossover operator is not sufficient to ensure GA convergence on the associated distance fitness.

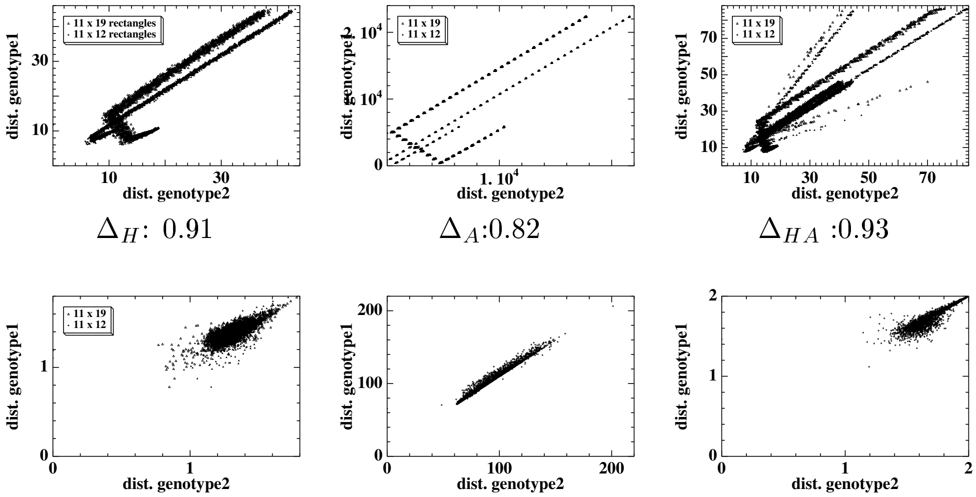
5. Distance and degeneracy The consistency of distance-based correlations with respect to genotype change are studied in the following: Figure 61 shows typical correlation plots between the distance to g_1 and the distance to g_2, where g_1 and g_2 are two different genotypes for the same phenotype. Some plots show superposed results for two different choices of genotype g_2.
The matching based distances, globally show the same tendancies: They are robust to degeneracy when the number of rectangles N_1 and N_2 respectively of the genotypes g_1 and g_2 are close to each other. Otherwise, if $N_1 < N_2$ (ex. 11 vs. 19 at Fig 61-a and -b) the correlation plot is far from linear. Let's describe what happens in the case of Δ_H for example.

Consider an individual x with N rectangles:

1. If $N = 1$, then the dots corresponding to x lie around the point ($d(x, g_2) = N_2, d(x, g_1) = N_1$). For example, the point (19,11) at Figure 61-a.
2. If N increases up to N_1, x gets closer to both g_1 and g_2 at the same time.
3. If N increases from N_1 to N_2, x gets closer to g_2, but further from g_1. For instance the distance $d(x, g_1)$ is bounded below by $N - N_1$. This explains the first angle observed in the figures.
4. When N increases over N_2, x gets further from both g_1 and g_2, which yields a second angle in the plots: the distance $d(x, g_i)$ increases up to N_{max} (50 here).

On the other hand, if the *Hausdorff* distances are more robust w.r.t. to degeneracy than the matching distances (with respect to the number of rectangles), their correlation plots show a smaller diversity, relative to distance range.

6. Conclusion Table 11 puts together the results obtained with the compared distances: For each of the 5 measures detailed above, it gives the percentage of *error* (the difference from the results one would obtain in the binary space; the most striking errors are indicated



Δ_H: 0.91 Δ_A:0.82 Δ_{HA} :0.93

Θ_H: 0.78 Θ_A:0.99 Θ_{HA} :0.92

FIG 61 : *Correlation plots and rates (τ_{DDC}) between the distance to a genotype g_1 of the corner phenotype, and the distance to a genotype g_2 of the same phenotype, on a random sample of 3000 individuals. Some figures show two superposed plots: the correlation between distances to g_1 and distances to g_2, and the correlation between distances to g_1 and distances to g_2' (another corner genotype).*

relative 'error'(%)	$\mathbf{\Delta_H}$	$\mathbf{\Delta_A}$	$\mathbf{\Theta_H}$	$\mathbf{\Theta_A}$
Mutation δ_{Mut} (*100/D)	2	2	**20**	4
Crossover $(1-\tau_{PO})$ (*100)	1	2	**53**	8
Degenerescence $(1-\tau_{PO})$ (*100)	1-**11**	1-**19**	6-**22**	1-5
Distance fitness $d(X_{N_c}, X^*)$ (*100/576)	0-1.5	1-**11**	2-**13***	**7-45***
distance range $D = diam(\mathcal{G})$	50	50×576	2	2×576

Table 11: *Summary results of different comparative criteria of the genotypic distances on the* **rectangles representation**. *For each measure, the difference to the results one would obtain with the binary representation are given, relatively to the measure range. The * sign at last line, indicates cases where the number of rectangles of the best individual found by the GA (optimizing the distance fitness functions), is different from that of the optimum. Bold font indicate big errors.*

in bold). It follows from these comparisons that the Δ_H distance is probably the most suited distance for the landscape analysis for Rectangles representation (in relation to the GA and operators at hand).

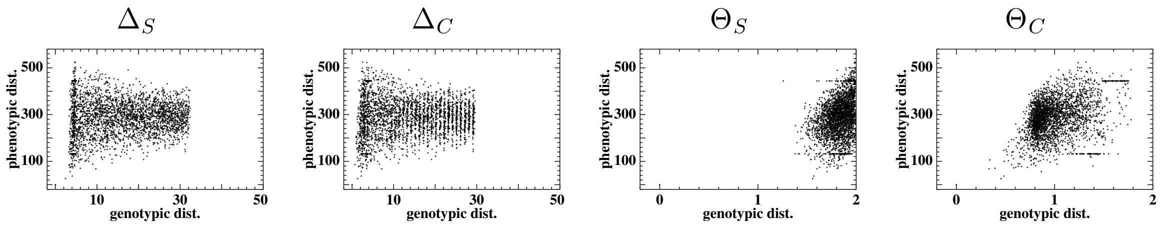


FIG 62 : *Correlation between genotypic and phenotypic distances to corner genotype.*

III.5.6 DISTANCE COMPARISON FOR Voronoï REPRESENTATION

We now perform the same analysis for Voronoï distances:
Having seen that the holes based distances (ex. Δ_H) are much better than the area based ones (ex. Δ_A), we focus on comparing the Voronoï genotypic distances, in the case where each site (resp. cell) contributes to the distance with a maximal amount of 1 (Eq. 184 and 189).

We will concentrate on the results regarding the distance landscapes criterion, as the results obtained with the other measures globally bring the same conclusions than with the Δ_H and Θ_H distances on the rectangles representation.
However, a noticeable feature is that the diversity of the Θ_C distance is much better than that of Θ_S, as shown in Figure 62. Yet, GA results on the Θ_C landscapes are largely below those obtained on the Θ_S landscapes. For instance, bold fonts in Table 64 show failure cases on the distance fitness functions.

We now turn to comparing **distance landscapes** results. Table 64 summarises the results of the runs from different points of view: fitness value (genotypic distance to solution), hamming distance to solution, and convergence time. Figure 63 shows median results out of 21 runs with different distance functions.

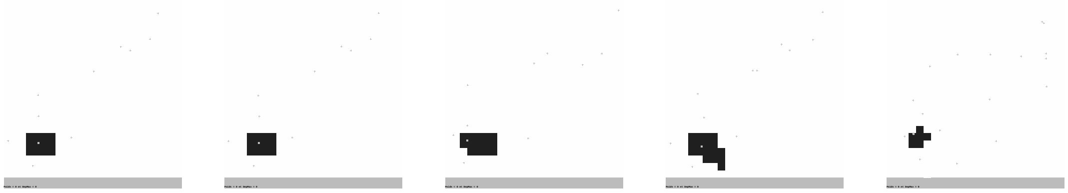
First, let's focus on comparing cell and site-based matching distances. If we consider (median, average or best) GA convergence speed on *distance functions*, the cell-based distances largely outperform the site-based distances. However, a closer look at results quality largely enfavors the site-based distances. For example, median results with the checker and asymmetric distance functions are sketched in figures 63, in both cases, the best results are found by Δ_S distance.

Second, we compare the matching and the *Hausdorff* distances:

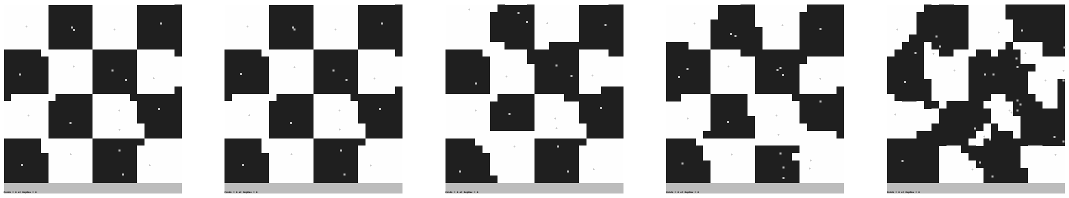
• The difference in solution quality between cell and site-based distances is a lot wider for the *Hausdorff* than for the matching distances.

• GA convergence speed is almost proportional with respect to the number of sites with the matching distances, whereas it's more or less stable with the *Hausdorff* distances. Note the analogy with the binary Hamming distance functions (ex. OneMax), where a convergence speed of (1+1)-ES starting at some distance k from the solution is $(l \ln k)$ for a string length of l.

• However, the picture changes if we consider results quality: Obviously, when the number of sites increases, the *Hausdorff* distance function is largely outperformed by the other matching



optimum Δ_S Δ_C Θ_S Θ_C



optimum Δ_S Δ_C Θ_S Θ_C

FIG 63 : *Distance functions: results of median run over 21, with different genotypic distance fitness functions. The* Hausdorff *distance is significantly outperformed by all others.*

	dissy^{11}	**sym**^{7}	**coin**^{5}	**damier**^{20}
$\Delta_\mathbf{S}$	0.08(0.03)	0.05(0.02)	0.04(0.01)	0.19(0.02)
Ham	0.3(0.5)	0.3(0.4)	0(0)	0.7(0.8)
N_c	785(205)	438(120)	307(72)	1090(37)
$\Delta_\mathbf{C}$	0.09(0.03)	0.02(0.01)	0.01(0.007)	0.1(0.06)
Ham	1(1.5)	4.6(5)	2.7(3.7)	12(8)
N_c	503(334)	267(71)	197(59)	706(290)
$\Theta_\mathbf{S}$	0.5(0.4)	0.13(0.05)	0.1(0.05)	0.6(0.3)
Ham	7(4)	1.7(0.05)	15(16)	**43(80)**
N_c	**315(177)**$^{(\mathbf{12,11})}$	352(86)	312(234)	**208(174)**$^{(\mathbf{24,20})}$
$\Theta_\mathbf{C}$	0.08(0.01)	0.04(0.01)	0.02(0.01)	0.1(0.04)
Ham	5(2)	18(7)	15(10)	**111(19)**
N_c	**182(57)**$^{(\mathbf{19,26})}$	**147(54)**$^{(\mathbf{23,19})}$	**210(99)**$^{(\mathbf{9,5})}$	**272(100)**$^{(\mathbf{34,29})}$

FIG 64 : *Average (standard deviation) GA results on different* distance fitness *functions, over 21 runs. The first line gives the fitness (genotypic distance to optimum) of the best individual found by the GA, the second line (Ham), its phenotypic distance to optimum , and the third line (N_c) gives the number of generations to convergence. The number of sites found by the (median,best) runs is given as an exponent, when different from the optimum. Bold fonts indicate* **failure cases**, *where either the number of sites or the phenotype found by the GA is false.*

based distances. For example, in the case of a *Hausdorff* fitness to a checker, the number of sites (and the phenotype) of the best individual found by the GA is completely different from that of the solution (bold fonts in table 64).

• Further investigation of the Θ_S distance function to a checker, shows that the big variance in convergence speed and Hamming distance to solution is mainly due to the fact that two runs get stuck in a local optimum, within 100 generations. These local optima are respectively all white and all black structures, with only one Voronoï site.

Hence, the most robust distance with respect to results quality is certainly the Δ_S distance. Though, a good compromise between speed and quality is given by the Δ_C distance.

While the *Hausdorff* distances are completely unreliable for holes representations, the Θ_S distance here for Voronoï representation gives rather good results. Still the *Hausdorff* landscape contain strong traps that make the results from few runs unreliable.

III.6 Landscape-Based Analysis

The experiments conducted in sections III.5.5 and III.5.6, allow us to choose, for each representation defined in section III.3, the genotypic distance that is best suited for landscape analysis (according to the adequacy measures defined in section III.5.4): The matching holes based distance Δ_H (and alternatively Δ_H^{φ}) for the rectangles representation, and the matching site based distance Δ_S for the Voronoï representation. Further hints for choosing the holes (resp. site) based distance, follow from the strong correlation between these distances and the number of minimal mutation application that link any two points of the search space. This is widely investigated in section III.7.

This section is dedicated to landscape analysis using the above distances (Δ_H, Δ_H^{φ} and Δ_S). First some tools for landscape analysis are designed. Second, using these tools, different hints are given to allow an *a priori* choice of a problem-dependent representation and an initialization procedure for a given representation.

III.6.1 EXTENDED FRAMEWORK FOR LANDSCAPE ANALYSIS

This subsection addresses some important issues that either were not discussed in the previous works about landscape analysis, or are irrelevant in the bitstring framework (chapter II).
Most methods dedicated to fitness landscape analysis, are based on random sampling techniques, as fitness distance correlation [JF95], epistasis [Dav90], operators improvement or correlation, [FG96, Wei90, MdWS91].
Special attention is given, in the following sections, to the case of fitness distance correlation. In addition to the size of the random sample, some extensions of FDC concepts are proposed to relax the prerequisite that the global optimum is known, and to overcome problems due to the degeneracy of the representation.
All results are discussed, in turn, in light of related works in section III.7, and some heuristic procedure for the choice of a representation for 2D shape is given, completing the case study. Finally, all results are summarized in the concluding Section III.8.

III.6.1.1 Size of the random sampling

The importance of the random sampling procedure in any statistical study, as well as for the initialization of any EA, does not need to be emphasized. Most of the problems presented in [JF95] involve small bitstrings, and the FDC can be computed by enumerating the whole search space. For all problems involving more than 12 bits, though, a sample of 4000 individuals is randomly chosen, uniformly in the search space. Unfortunately, this arbitrary choice of the sample size might not be suited to much larger spaces, not to mention continuous spaces.
Moreover, even for the same search space, different problems might require very different sample sizes. Caricature examples are given by the OneMax and the LongPath problems in $\{0,1\}^N$.
Yet another important factor influencing the “quality” of the sample is the random initialization procedure. It is usually acknowledged to be uniform over the search space – as the choice of bits with equal probability for 0s and 1s over $\{0,1\}^N$. However, it might be difficult to design an unbiased initialization procedure on search spaces for the variable length shape representations: Section III.6.2 will address this issue in detail. Nevertheless, assuming a given initialization procedure, the rest of this section proposes some hints for automatically

adapting the size of the random sample to both the size of the search space and the ruggedness of the fitness landscape.

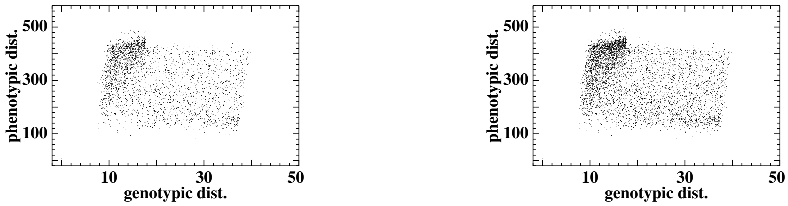
Search for Stability With little hope for any *a priori* estimation of a sufficient size for the random sample, *a posteriori* tests for stability were the only choice: If the addition of a significant number of examples to an existing sample does not significantly modify some statistical measurement on that sample, it can be considered stable for this measure.
The most obvious measure that can be used to estimate that the stability is reached is the target value itself (here the FDC coefficients), though it is always dangerous to test the statistical stability of a sample using the target measure that is estimated on that sample, due to possible over-fitting.
Other possible measurements rely on estimations of the diversity in the sample. Diversity is difficult to assess, but can be measured considering the repartition over the sample of known quantities, such as the fitness values, or some phenotypic or genotypic distance to some known optimum.

The Sizing Algorithm For a given measure to test the stability of the sample, the following algorithm is proposed.

1. Generate N (typically 1000) random individuals.
2. Sort the initial sample according to the chosen measure (ex. fitness values or distance to optimum).
3. Cut fitness range into intervals, each one containing the same proportion ρ of individuals of the sample (typically 20 intervals with $\rho = 5\%$). Note $f_0^1, \dots f_{\rho^{-1}}^1$ the fitness-intervals boundaries.
4. Repeat steps (a-b-c) until the average difference of the fitness-interval boundaries (Δf) becomes smaller than some proportion ε (typically 0.1) of the average interval length.
   (a) Add N random individuals to the sample.
   (b) Compute the new fitness-intervals boundaries as in step 3 ($f_0^n, \dots f_{\rho^{-1}}^n$).
   (c) Compare to the previous fitness-intervals boundaries: $\Delta f = \overline{|f_i^n - f_i^{n+1}|}$

New individuals are added to the sample until the repartition of the fitness values becomes stable with respect to some criterion. Note that many experiments were conducted after the empirical stability test succeeded, without ever experiencing any loss of stability. Of course, the three parameters N_{epoch}, ρ and ε have to be adjusted. But – and this is confirmed by experiments – the resulting algorithm is robust with respect to these parameters, and the values given above were used for all following experiments.

The sizing algorithm at work Practically, in the specific case of the variable-length representations for shapes, three criteria have been tried, concurrently, to test the stability: the fitness, the genotypic distance to some global optimum genotype, and the phenotypic distance to a known global optimum. Typical results regarding the minimal sizes of the stable sample in both cases, with epoch size of 1000, are given in Table 12, together with the subsequent FDC coefficients.
For the (1- and 0-) rectangle representations, the samples generated using the genotypic distance Δ_H are generally smaller than those generated using the phenotypic (*Hamming*)



(a) 3000 sample (genotypic stability) (b) 5000 sample (fitness stability)

FIG 65 : *FDC plots (with Δ_H distance) using the two different stability criterion of the distance to optimum (a) and fitness value (b), on the corner problem for the 0-Rectangle representation. The FDC coefficients and the global outlook of the plots look very much alike, though two more samples were necessary to reach stability in the fitness space.*

Representation	**Genotypic stability**	**Phenotype stability**	τ
1-Rectangle	3000	3000	0.5
0-Rectangle	3000	5000	-0.3
Voronoï	4000	4000	0.02

Table 12: *Minimal stability sizes and corresponding FDC coefficients on the inclusion problem for both the distance-to-optimum (genotypic) and the phenotypic criteria.*

distance. The situation is reversed for the Voronoï representation. Yet, in all cases, the correlation phenotypic versus genotypic coefficients and plots are very similar. A detailed comparison is presented for the checker problem with 1-Rectangle representation, for which the largest difference in sample size was observed. On the correlation plots in Figure 65, the general outlook of the samples, after stability was reached, are identical.
Further experiments have been conducted using the Θ_H distance. Genotypic Stability sample sizes with respect to the Θ_H distance, are still around 3000 or 4000 individuals.

III.6.1.2 Extended FDC

Some important difficulties appear when trying to blindly apply the basic definition of FDC, as recalled in section II.3, in complex, non-discrete, and degenerate spaces: in addition to the unrealistic requirement that the global optimum is known, the effects of epistasis and degeneracy effects are unknown.
These problems are addressed in the following: an extended definition of FDC is proposed and the possible bad effects due to the epistasis and the degeneracy of the representation are discussed.
The extended FDC is then used on the problem of mechanical inclusions identification. Whereas the accuracy of FDC predictions is clear for tuning the initialization procedure (Section III.6.2), the use of FDC as a tool for a priori choice of a representations seem a little hazardous in general: the comparison of absolute values of FDC coefficients for two different representations is completely unreliable. However, useful hints can be obtained for the inclusion problem, to limit the number runs of "good old" trial-and-error method.

Local FDC One of the major limitations of FDC for real problems is that its computation requires the knowledge of the global optimum (i.e. the solution of the problem at hand).

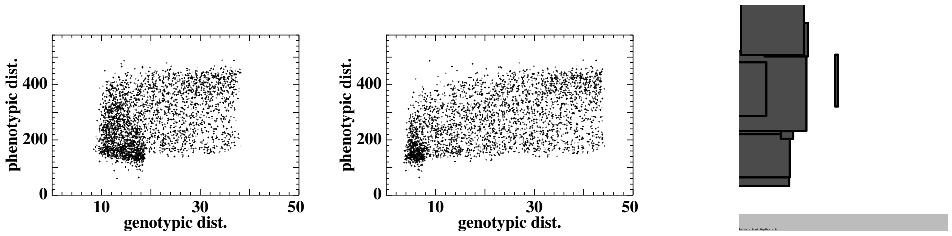
A "Local Fitness Distance Correlation" has been introduced and advocated in section III.6.2, relaxing this unrealistic prerequisite. The Local FDC computes the correlation coefficient between a sample and the best individual *in that sample*. Such a coefficient, termed τ_{loc}, is computed as in Equation (124), with the difference that all distances are now distances to the "local" optimum (the optimum in the sample). Of course, this coefficient will likely depend on the choice of that local optimum, and its stability with respect to different samplings should be checked carefully.

Local FDC at work The above ideas have been applied to the variable length representations studied in this chapter. The first remark is that the optimum in the sample generally resembles the actual solution of the problem (Figure 66-c). Hence it is not surprising that the FDC plots and coefficients (Figures 66-a and -b) do not differ much. Though, recall that the matching distances are very sensitive to degeneracy when the number of rectangles from one genotype to the other is very different (as illustrated by correlation plots (DDC) of figure 61). This explains why local and global correlation results are different with the caricatural asymmetric case of figure 67. For instance the number of rectangles of the global optimum used for these experiments (Fig. 67) is 4, versus 45 for the "local" optimum (the optimum in the sample).
But of course these results could have been expected from the experiments of section III.5.4, figure 61 , that estimated the correlation of distances to different genotypes through the DDC coefficients: the distance to some genotype is a quasi linear transformation of the distance to another genotype, if they have close numbers of alleles only.

The fact that the distance is directly related to the number of rectangles, weakens the reliability of correlation results: this distance does not allow to distinguish whether diversity is due to degeneracy (superposed rectangles for example) or due to a different coverage of the design domain.
This motivates the use of a modified version (denoted Δ_H^{φ}) of the Δ_H distance, as defined in section III.5.2.3 and equation 181: With Δ_H^{φ}, unmatched rectangles contribute to increase the distance according to the amount of degeneracy they involve. In this way, superposed rectangles don't increase the distance. This measure (Δ_H^{φ}) is very robust to degeneracy, and behaves fairly with respect to all the criteria of distance comparison of section III.5.4 (not presented here). Note that Δ_H^{φ} satisfies the triangular inequality on the space of fixed length rectangles representation, and that on that subspace it coincides with the Δ_H distance.
Further experiments confirm the usefulness of the Δ_H^{φ}, at least as a diversity measure of random samples generated with different initialization procedures.

III.6.2 OPTIMIZING THE INITIALIZATION PROCEDURE

One of the important issues which is generally omitted when designing an Evolutionary Algorithm is that of the initialization procedure: most of the time, a uniform sampling procedure is assumed, and easily implemented, be it based on bitstrings or real-valued parameters with prescribed bounds. However, when dealing with complex representations (e.g. variable length), there might exist no obvious uniform initialization procedure.
Moreover, all test procedures for distances defined in section III.5.4, as well as the analysis tools based on FDC (sections II.5.4 and III.6.1), make intensive use of the random sampling procedure. It is hence of vital importance to study how the modification of that procedure influences on the resulting snapshot of the fitness landscape. Further, it might allow to design optimized random initialization procedures: whereas the use of FDC is more than questionable for long-term predictions of actual evolutionary runs (see sections II.3 and

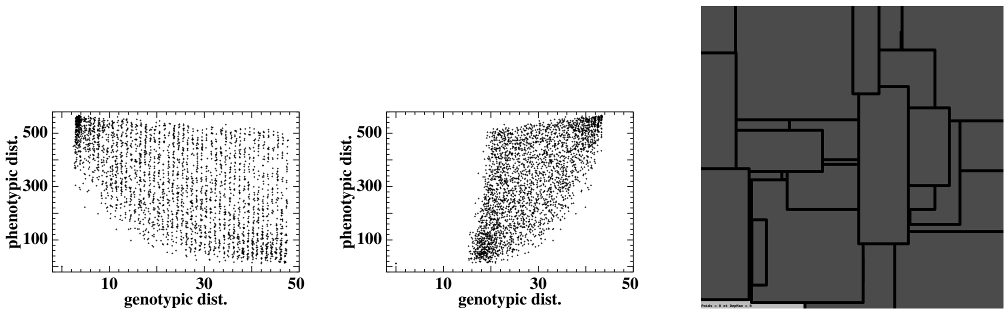


(a) Global FDC
19 rectangles genotype

(b) Local FDC
8 rectangles genotype

(c) Local optimum genotype

FIG 66 : *Comparison of the global (a) and one local (b) FDC plots and coefficients (with Δ_H distance) for the 1-Rectangle representation on the corner problem. The actual optima of the sample is shown in (c).*



(a) Global FDC
4 rectangles genotype

(b) Local FDC
45 rectangles genotype

(c) Local optimum genotype

FIG 67 : *Comparison of the global (a) and one local (b) FDC plots and coefficients (with Δ_H distance) for the 0-Rectangle representation on the asymmetric problem. The actual optima of the sample is shown in (c).*

III.6.3.1), using different initialization procedures certainly has direct consequences on FDC results, in turn useful for tuning the initialization procedure itself. Unexpected results have been obtained that way in the bitstring framework in section II.5.

An initialization procedure can be viewed as optimal considering the analogy with uniform coverage of the search space: the aim can be to get the largest range of both fitness values and intra-sample distances.

III.6.2.1 Rectangle representation

Consider the rectangle representation, with initial domain $\Omega = [0, L_x] \times [0, L_y]$.
Three initialization procedures have been defined. In each case, we assume that a maximal number of rectangle R_{Max} (50 in all experiments of this chapter) and maximum values for the dimensions of the rectangles x_{Max} and y_{Max}, are defined.
Then, for each individual, we have to choose the number of rectangles, and the position and shape of each rectangle.

Uniform number This procedure was the very first idea that came out with the holes representation. Then, for each individual,

1. Draw a number of rectangles uniformly in $[1, R_{Max}]$.
2. Repeat until the chosen number of rectangles is reached
   (a) Choose the coordinates of the center of each rectangle, uniformly in Ω.
   (b) Choose the dimensions of the rectangle uniformly in $[0, x_{Max}]$ and $[0, y_{Max}]$.

Of course, different choices of x_{Max} and y_{Max} do lead to different sampling of the genotype space, hence resulting in different behaviors of the evolutionary algorithm. This phenomenon can be captured by the FDC analysis, as witnessed by the FDC coefficients and plots of Figure 68-a and -b, for the values $x_{Max} = L_x, y_{Max} = L_y$ and $x_{Max} = 0.2L_x, y_{Max} = 0.2L_y$. Note that results with the Δ^{φ} function are much more spectacular. Moreover, actual runs of the Evolutionary Algorithm demonstrate similar influence of the initialization procedure in the early generations (not shown).

Uniform ratio Another procedure can be defined, whose aim is to obtain a uniform sampling of the **ratio** of both materials (zeros and ones) in the initial population. For each individual,

1. Choose a ratio ρ of the rectangle material uniformly in $[0, 1]$.
2. Add rectangles (as in the uniform number procedure) until the ratio of rectangle material reaches ρ.

When compared to the first definition (with size parameters half of Ω dimensions), this procedure mainly changes the distance-to-optimum range of the sample up to five times more, but no significant difference was ever observed in the FDC results.

Uniform bounds Both above procedures strongly depend on the choice of x_{Max} and y_{Max}. To overcome this dependency, the uniform number procedure above has been modified. For each individual,

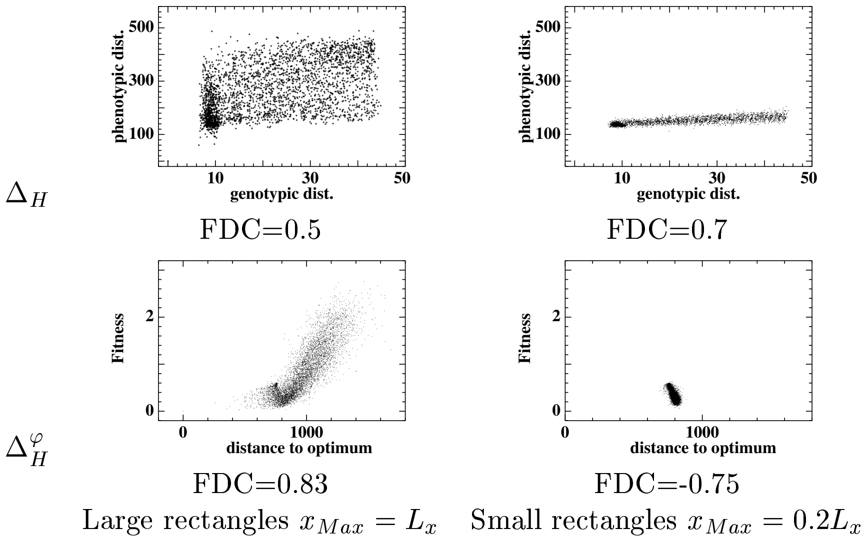


FIG 68 : *FDC plots for different size of rectangles for the uniform number procedure of the 1-rectangle representation on the corner problem: large rectangles ($x_{Max} = L_x$) give large ranges of phenotypic distances while small rectangles ($x_{Max} = 0.2L_x$) only sample a very limited region of the search space. Though, no useful information can be derived from the diversity on the x-axis: it reflects the fact that the initialization procedure yields a uniform number of rectangles in the sample. On the other hand, the Δ_H^φ distance allows to go beyond degeneracy effects (linked to superposed rectangles), and gives a compromise between phenotypic and genotypic diversity*

1. Draw a proportion π uniformly in $[0.2, 0.6]$.
2. Define x_{Max} and y_{Max} as $\pi \times L_x$ and $\pi \times L_y$ respectively.
3. Generate the rectangles as in the Uniform number procedure above.

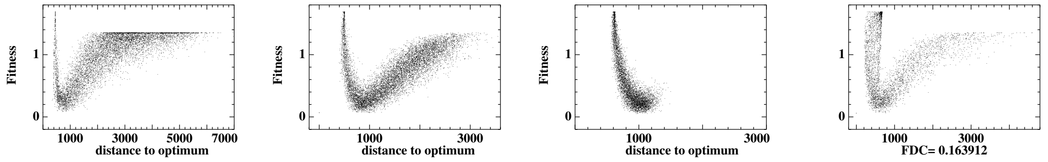
Using the same bounds for all individuals induces strong biases in the sample: small values implies that only small holes can be created, while large values will hardly generate individuals with only small holes. Hence this latter procedure have been used in all experiments presented in this chapter.

Note that the second procedure could be modified the same way (with random x_max and y_max values for each individual). Experiments (not presented here) show that it gives similar FDC plot shapes and coefficients as the latter described, but with higher distance to optimum range. No major difference in runs convergence have been observed between them.

III.6.2.2 Triangle representation

As for the Triangle representation, three initialization procedures have been designed following the guidelines of the uniform number procedure of rectangles representation presented in III.6.2.1, with differences on how triangles are generated:

1. The **3 points procedure** generates the three summits uniformly in a domain larger than Ω.



FDC=0.72	FDC=49	FDC=-0.78	FDC=0.16
(a) 3 points	(b) 0.5-one point	(c) 0.25-one point	(d) uniform bounds

FIG 69 : *FDC plots with the Δ_H^φ distance, for four different initialization procedures and the Triangle representation on the checker-board problem. The upper bound in plot (a) corresponds to the genotypes made of a few degenerate triangles, having homogeneous default-value phenotypes. Note the differences in shapes (and FDC coefficients) from (a) to (c), while (d) seems to cover the search space much more properly.*

2. The **γ-one point procedure** generate one summit S uniformly in Ω; the coordinates of the two other summits are then drawn following a Gaussian probability distribution around S of deviation γL_x, γL_y.

3. The **uniform bounds procedure** is the same as the **γ-one point procedure**, except that, for each individual, γ is first chosen uniformly in $[0, 0.8]$.

Figure 69 shows typical results of the 3 points procedure, and the one point procedure for two values of γ (0.5 and 0.25), on the checker-board problem. The 3 points and the 0.5-one point procedures (Figure 69-a and -b) do not generate enough examples on the left lower side of the plot (genotype made out of small triangles and possibly high fitness), though the phenomenon appears stronger for the 3 points procedure. On the other hand, the 0.25-one point procedure (Figure 69-c) doesn't generate enough chromosomes on the right side of the plot. The most complete coverage of the search space seems to be given by the **uniform bounds procedure** as can be seen in Figure 70. Hence it has been used in all experiments.

III.6.2.3 Voronoï representation

For the Voronoï representation, procedures similar to the **uniform number** and the **uniform ratio** procedures for the rectangle representation have been tested – with the difference that, after a site has been added, its label is drawn in $\{0, 1\}$ either with probabilities $(0.5, 0.5)$ (for the uniform number procedure) or with probabilities $(\rho, 1 - \rho)$ (for the uniform ratio procedure).
However, only very slight differences have been ever observed in correlation results and distance ranges, and the uniform number procedure have been used for all results presented in this chapter.

III.6.3 CHOICE OF A REPRESENTATION

In this subsection, FDC and operator ranking tools are studied from the point of view of *a priori* choice of a representation on a given problem. The outcomes of each criterion will be systematically compared to actual runs of an Evolutionary Algorithm:
Performance of actual runs of the evolutionary algorithm have already been studied in section III.4.3, and are presented in figure 53.

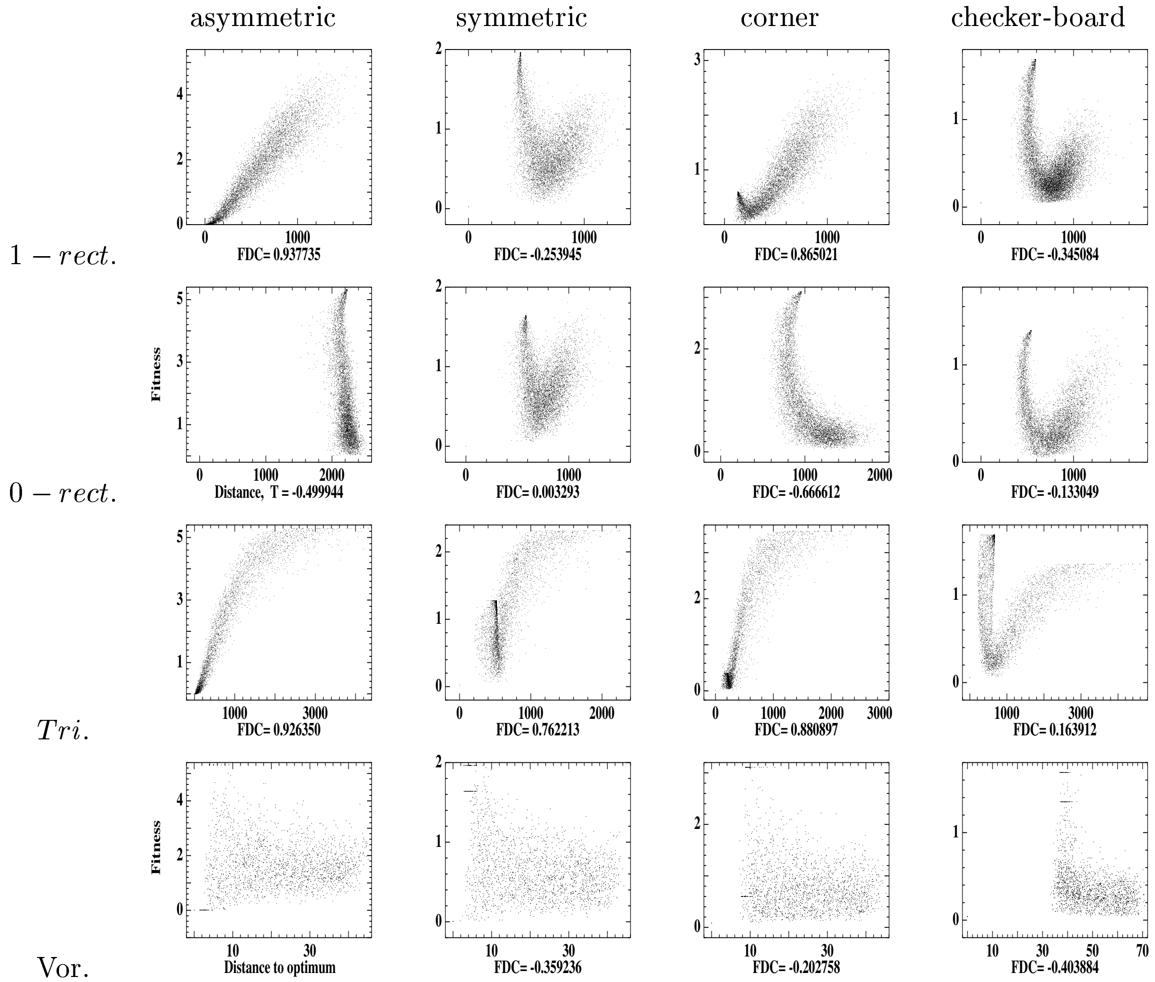


Fig 70 : *Local FDC plots with the Δ_H^φ distance for different representations on the four test problems: fitness (y axis) $\times$ distance to optimum (x axis). Columns should allow to compare representations for a given problem, and rows to compare problem difficulties for a given representations.*

III.6.3.1 FDC and Choice of a representation

In this subsection, FDC is studied from the point of view of *a priori* choice of a representation on a given problem. Actual performance of the evolutionary algorithm are presented in Figure 53.

All FDC plots and coefficients useful for section III.6.3 are presented in Figure 70. Note that some of these plots were presented separately before, to illustrate different aspects of FDC behavior. It seemed however useful to group together now, as would be the case when facing a new problem with stabilized analysis tools. In Figure 70, all plots in the same column address the same problem, have been scaled almost identically on the fitness axis, while all plots on the same line use the same representation/distance, with same order of magnitude on the horizontal axis (the distance between points and local optimum).

One striking fact on Figure 70 regards the asymmetric and corner problems, with both rectangle representations. And indeed such difference in FDC plots and coefficient does reflect the actual situation when it comes to evolutionary runs (Figure 53). This allows to identify which material is worth choosing as a default and which is worth filling with holes.

On the other hand, the situation is not that clear for choosing the representation that fits the inclusions shape in the optimal structure (rectangles for the asymmetric, and triangles for the corner). The situation is even worse for the Voronoï representation. The outlook of FDC plots (Voronoï line of Figure 70) does not seem to show any clear tendency, as the clouds of points are rather dimly spread all over the space.

Hence, FDC considerations does not allow to compare problem difficulty or even representation efficiency on the same problem. The study of operators influence on GA results is mandatory:

III.6.3.2 Operator Ranking and Choice of a representation

Operator ranking method has been proposed in section II.6 to compare crossover operators on binary problems. The method is based on a random sample, among which some parents are selected (ranking scheme) to generate a number of offspring.

While section II.6 restricted to compare different binary crossover operators with the same initialization, the method naturally extends to compare the couples (initialization, genetic operators). In this context we propose to use the method to compare different representations on the same problem. More precisely, each representation involves an initialization procedure (chosen in III.6.2, and used to perform the genetic runs) and some genetic operators (presented in III.3).

Mutation and crossover offspring, with different representations, are presented in Figure III.6.3.2. The ranking is performed on offspring obtained from both operators (applied with equal probability). As explained in chapter 2 and section II.6, only the fittest offspring are considered.

Some clear tendencies emerge from comparing the –best offspring– fitness curves (Fig. III.6.3.2), and agree with the on-line GA results at first generations: It allows to distinguish the shapes of the holes, showing that the triangles are a good choice for the corner, and a bad choice for the checker. Further experiments with mutation and crossover offspring show that:

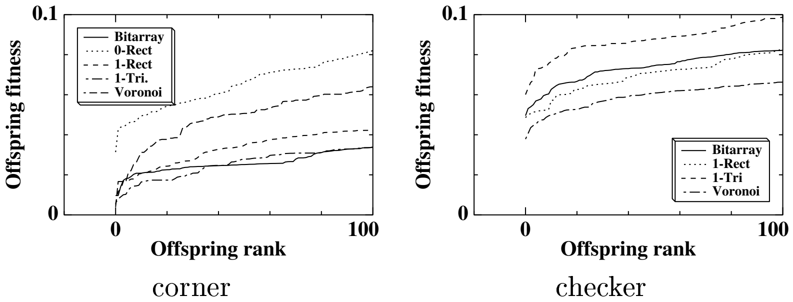


FIG 71 : *The fittest offspring sorted according to their fitness values (method of section II.6). The parents are chosen among a 300 random sample, with a ranking selection scheme. Each couple generates 30 offspring. Only offspring that outperformed their parents are presented.*

- On the corner problem, the mutation offspring (not presented) of the triangle representation, clearly outperform the other (mutation and crossover) offspring of all other representations. The bitarray representation offspring are also outperformed by the rectangles'.
- On the checker problem, the reverse situation happens: The triangle representation offspring are clearly outperformed by the others, in both mutation and crossover cases. And the Voronoï representation gives the best offspring.

But the situation is not clear for the Voronoï representation on the corner problem. Voronoï offspring are scarcely better than the rectangles', while the Voronoï on-line results are better. This is probably due to the initialization difference. The 1-rectangles initialization yields fittest individuals than the Voronoï does.

At more reliable comparison can be achieved if the random samples involve similar fitness distributions. A second alternative is to generate random samples (for different representations) equal in the space of phenotypes. But this requirement is too strong and may also lead to biased results. For instance, the phenotypes are (intended to be) of different kinds according to the representation (convex polygons or rectangle/triangle shaped) ...

As a conclusion, the offspring ranking method, again demonstrates able to distinguish between operators, and hence to compare different representations.
Note that the reliability of the offspring ranking method is far from proved, and this method certainly may lead to misleading results (it should be possible to construct counterexamples). Nevertheless, it allows to distinguish different representations as the Voronoï , bitarray and (different) holes representations, while the FDC correlations plots show no significant difference between these representations. On the other hand, when facing a new problem, GA practicers suffer from the lack of information to guide the choice of the search space and GA parameters. Hence, the information they may obtain from such analysis tool (offspring ranking) is to some extent better than the absence of information.

III.7 Discussion and Related Work

This section discusses the different issues addressed up to now from different points of view, and in the light of related work from EC literature. Meanwhile, some further directions of research are sketched to overcome present limitations.

III.7.1 Genotypic distance

In order to assess and compare genotypic distances, some measures have been proposed and tested, mainly derived from an analogy with the bitstring case. However, referring to the real valued case, other measures could be defined. For instance, in the Euclidean space of real-valued parameters, different crossover operators have been proposed, for which all offspring does not lie on the line between the parents, but in the hypercube whose diagonal is both parents. Hence an alternative measure of the adequacy between a distance and a crossover operator could be the deviation from that locality criterion. However, such alternative measure is experimentally more expensive to compute, requiring the distance between many offspring of the same couple of parents.
Another way to ensure there exists a link between a distance and an operator is to derive the distance from the operator itself. This is what has been done for the mutation operators in [RY96]. Two distances are designed in the context of strings on a given alphabet. First, the Viterbi Edit Distance is related to the **probability of the most likely sequence of operations** that transforms one individual into the other. Second, the Stochastic Edit Distance is related to the sum of probabilities of all possible sequences of operations that transform one individual into the other. In [HR96a, Jon95] in the bitstring case, another point of view is developed: the distance is the shortest path, in terms of **minimal number of mutations**, that transforms one individual into the other.

The transposition of the latter approach to the case of continuous parameter spaces seems hazardous, as it would probably require to compute the paths between two individuals as some integrals over possible continuous transforms in the genotype space starting at one individual and ending at the other one.
However, the stability of the distance between one individual and its mutated offspring – as measured by the δ_{mut} proposed in section III.5.4 for genotypic distances – witnesses some correlation between the minimal number of mutations that transforms one individual into the other, and the genotypic distances between these individuals.

On the other hand, considering a <u>discrete version</u> of the rectangle representation (integer-valued coordinates, and dimensions), it can be easily proved that there is a strong relation between the Δ distance, proposed in Proposition III.5.1, and the number of minimal elementary mutations that transform one individual into another, as explained below.
Relation to the minimal number of mutation applications: In the integer-valued context proposed above, consider an elementary mutation that adds 1 unit to the coordinates or dimensions of a rectangle.

Proposition III.7.1
The minimal number of elementary mutation applications $N^{mut}(R, R')$ required to get from rectangle R to the R', is equal to the Holes based distance δ_{L1} between two rectangles, in linear scale.

Proof. If we note $R = (x, y, l, L)$, then $N^{mut}(R, R')$ is equal to the $L1$ distance between the rectangles, used in the experiments of the thesis.

$$N^{mut}(R,R') = \delta_{L1}(R,R') = |x-x'| + |y-y'| + |l-l'| + |L-L'|$$

If we consider two individuals containing the same number of rectangles, the number of minimal mutations that link these individuals X and X' in $\mathcal{G}$, is given by $\min_{\mu}\big(\sum_i N^{mut}(R_i, R'_\mu(i))\big)$, to be compared to the genotypic distance constructed on $\mathcal{G}$: $\Delta(X,X') = \min_{\mu}\big(\sum_i \delta_{L1}(R_i, R'_\mu(i))\big)$
□

Hence, it is clear that the number of minimal mutations is equal to the –linearly scaled– Holes based distance, in the case of individuals with a fixed number of rectangles. The same applies to the Site based distance, in the Voronoï representation. This strengthens the analogy between Δ and the binary *hamming* distance related to the *1-bit-flip* mutation.

In the case of variable length individuals, the number of minimal mutations that link two individuals is not well defined. The question is: Do we consider a degenerate rectangle (i.e., with null dimensions) in the genotypes, as a rectangle or not? In the implementation used here, we forbid this situation (no degenerate rectangles are allowed, by construction). Hence, it is not possible to get from one individual to the other by using only such elementary mutations.
Nevertheless, it is possible to perform additional mutations to transform the unmatched rectangles of individual X into the closest rectangles of individual X'. In this case, the number of minimal mutations is strongly related to the Δ^ϕ dissimilarity measure defined in Section III.5.2.3 which makes an additional matching for the $|n-n'|$ rectangles, that are left unmatched.

Finally, recall that the heuristical Δ^ϕ has been found more useful that the exact distance Δ to tune EA parameter. This gives further hints for the importance of linking the distance to the number of operators applications (when possible) as already suggested by [Jon95] for example.

Nevertheless, beyond the relationship between distances and elementary mutations, another method to test the adequacy of a distance and the evolution operators and process, is to test whether the process is able to optimize *distance fitness functions* within reasonable time. This follows directly from the analogy with the binary space.

III.7.2 Random sampling

The first lesson from the experiments presented in this paper is that great attention must be paid to the random sampling of the genotype space: The stability-based procedure was designed to adjust the size of the random sample with respect to both the complexity of the search space and the ruggedness of the fitness landscape. This procedure demonstrated a large influence on the FDC results compared to the fixed-size sampling used in [JF95] for large problems.
However, in some problem cases, the "interesting" part of the fitness landscape almost surely escapes any random sampling. And even the problem-dependent procedure proposed in Section III.6.1.1 in such an ugly situation cannot be of any help. This is the case for the extended LongPath problem of [HG95].
Note that this limitation is met by all statistics-based *a priori* estimators developed in the Evolutionary Optimization framework. The Virtual Genetic Algorithm of Grefen-

stette [Gre95b], for instance, uses statistics on the fitness of random samples to derive a simple statistical model for the fitness landscape. This model then allows to perform the trial-and-error process of designing the most efficient algorithm at low cost, replacing the exact computation of the fitness value by the estimation based on that model. It is an elegant alternative to the standard brute-force parameter estimation process (as first described systematically in [SCED89]). However, in the case of large search spaces, the simple statistical model generally proves of poor help, as the fitness of all individuals in the random sample is many orders of magnitude worse than that of the target optimum.

An alternative to – the sometimes – useless increase of the size of the random sample can be to introduce some controlled bias in initialization procedure, or to generate samples at a given fitness level with the Metropolis algorithm as discussed in section II.3.7.2.

III.7.3 FDC and initialization procedures

When no obvious uniform procedure is available, FDC seems a good tool for comparing different initialization procedures, experimentally confirmed in the early stage of the evolutionary optimization (sections II.5 and III.6.2).
Little attention has been paid to initialization procedures in EA literature. One of the main reason is probably that uniformly distributed individuals are fairly easy to draw in the most common representations frames (i.e. bitstrings and real-valued parameters with prescribed bounds). It has been recognized, though, that a biased initialization resulting from a preliminary EA-run could greatly improve the overall results [SX93]. Moreover, recent work in the GP framework [Iba96] demonstrates that a careful choice of initialization procedure can enhance the performance of the further EA.

III.7.4 FDC and Fitness Landscape Analysis

A part of this study regards the analysis of the fitness landscape associated to the variable length representations at hand, using the Fitness Distance Correlation framework [JF95].
While chapter 2 (sections II.3 and II.4) widely investigated the limitations of FDC analysis as a measure of problem difficulty, this chapter demonstrates that the adequacy of a representation for a given problem is also beyond what FDC analysis may capture, as for bitarray representation compared to variable length rectangles representation for example (Fig. 70).

III.7.5 Operators and Fitness Landscape Analysis

Early work by Weinberger [Wei90] introduced the notion of fitness operator correlation, for asexual operators, as the correlation between the fitnesses of the parents and their iterated offspring. Manderick [MdWS91] adapted this correlation measure to crossover operators. He paid attention to the correlation between mean parents fitnesses and mean offspring fitnesses, and demonstrated the usefulness of this measure to assess the role of an operator on the NK-landscapes [Kau89] and the TSP problems. More recent work [FG96] advocate the need to use the mean improvement of offspring, and concludes that the uniform intermediate recombinations yields better results (higher mean improvement) than the one-point crossover, for three selection methods, in the framework of parameter optimization.

Section II.6 of chapter 2 propose a comparative study of these methods in the bitstring case, demonstrating practical limitations of such measures.

This chapter (indirectly) studied an additional method, that measures connection between the fitness and the operators: in the case of *fitness distance functions*, τ_{PO} measures the extent to which the offspring lie in a fitness line between their parents (cf. definition in III.5.4 and results in III.5.5). But the presence of this (strong) connection is demonstrated to be of poor help, as it does not help the operators to optimize the *fitness distance* functions (in the case of Θ_A for example).

The operators ranking method (first proposed in section II.6), again, demonstrates to be a reliable tool to compare operators, and therefore, to compare and choose an *a priori* representation for a given problem.

III.7.6 Representations for 2D shapes

The starting point of this work was the mechanical problem of inclusion identification, and the choice of a variable-length representation among the *holes* or the *Voronoï* variable-length representations. In spite of their high degeneracy and epistasis, to the best of our knowledge, the variable length representations studied here are the best choice for the mechanical problem at hand.
Following the general guidelines of section III.5.1, different distances have been designed for these two representations. The properties of these distances in relation to evolutionary process and degeneracy have been measured, assessing their adequacy to efficiently describe the notion of proximity for an evolutionary algorithm. These considerations allowed to choose one distance for each representation: the Hole-based distance (Section III.5.2.1) for the *holes* representation and the Site-based distance (Section III.5.3.1) for the *Voronoï* representation. Using these distances, FDC results allowed to optimize the initialization procedures, but were loosely related to the overall quality of the representation at hand.
The operators ranking method of section II.6 (confirmed by actual runs of EA) allows to design a general heuristic for the *a priori* choice of a representation, based on available information about the problem class:

- Voronoï representation should be used when the solution contains equal quantities of both materials;
- Hole representation seems preferable in the other case, but great care must be taken when choosing the default value: the holes should correspond to the minority material.
- Further, if using the Hole representation, the shape of the holes is important, too – though in general hardly known in advance in real cases.
- However, the off-line performances – which random-sampled based FDC analysis fails to capture – seem to indicate that the Voronoï representation actually slightly outperform *Holes* representation on all problems but the caricature asymmetric problem.

In the absence of any a priori information, the Voronoï representation seems the best choice, according to the present results.
On-going and further work is concerned by improving the operators (e.g. use ES-like adaptive mutation on all real-valued parameters), and extending those representations to real-valued labels, thus allowing us to deal with more complex identification problems, such as the same mechanical problem, but in the case where the mechanical characteristics of the materials are unknown, or the inverse problem of geological underground identification from seismographic data [SEB98].

III.8 Conclusion of Chapter III

This chapter has presented a case study of evolutionary optimization applied to different variable length representations on the same inverse problem of mechanical engineering: from the definition of the representations themselves, together with the associated evolution operators, to the analysis of the resulting fitness landscapes using some correlation and operators measures, validated – or contradicted – by actual test runs of an evolutionary algorithm.

One of the important issues on that road was the design of a useful genotypic distance for each representation at hand: General guidelines have been given, simply based on a similarity defined on the allele space. Moreover, some measures of adequacy of distances with evolution operators have been proposed, allowing us to choose a distance somehow correlated to the evolutionary algorithm. In particular, the *distance landscape* criterion (defined in section III.5.4) has been found very instructive.

The FDC tool introduced by Jones and Forrest in the bitstring framework [JF95] has then been extended to variable length representation involving continuous parameters. Some limitations of the seminal theory were addressed: The size of the random sample was automatically adjusted to the problem characteristics; A Local FDC analysis was proposed to handle the realistic cases where the global optimum is actually unknown.

All these ideas were then experimented to study the variable length representations at hand: FDC analyses have been found useful for the *a priori* choice of an initialization procedure. Further, FDC demonstrated to be of potential use for choosing between 0-holes and 1-holes representations. For instance, it detected weird situations, on the caricature asymmetric and corner problems for the 0-rectangle representation.

Moreover, we gave further evidence that the operators can have more influence on the convergence than the correlation of the fitness landscape witnessed by FDC:

The offspring ranking method demonstrated to be useful for comparing representations. Yet, the use of different initializations weakens the comparisons, in the long term. This problem does not appear when comparing different operators with the same representation. However, to overcome this problem, one would design samples having similar fitness distributions, with different representations, before comparing the operators. Of course this can be done with the metropolis algorithm (section II.3.7.2), or simply from GA first populations.

Note finally that the methodology used in the case study of this chapter, in order to derive heuristics to choose a good representation (resp. initialization and genotypic distance), is not specific for the inclusion problem, or for shape representations, but have already been successfully used in the binary case, on a host of functions. Hence, one might expect that such tools (FD plot, *Offspring ranking*, adequacy between distance and evolution), may be helpful for identifying further problem dependent heuristics. Precise answers to this question shall be left for further work.

Conclusion and Further Work

Summary and Conclusions

This thesis starts by a theoretical study of some particular fitness functions and algorithms. Further, some practical tools of fitness landscape comparisons are further investigated. Last, some of these tools are used to compare variable length representations, for the inverse problem of inclusion identification.

The theoretical results of the first chapter provide some lower and upper bounds on the complexity of simple search procedures in l-dimensional spaces, for big l values: The time to convergence of mutation only Hill Climbers, ranges from almost linear to an exponential of l (respectively on the unimodal OneMax and Long k-path functions). In the OneMax case, convergence is quicker with the deterministic bit flipping mutation than it is with the stochastic mutation, whereas the opposite happens for the Long k-path problem. On the other hand, random walks directed by either stochastic or deterministic mutations, require the same amount of time to hit a target point of the search space.
Turning to population based algorithms, the first chapter also constructs controlled paths for GAs, showing that increasing the population size is not sufficient to avoid the fatal and narrow controlled path leading to the optimum, and that populations can result in a slower convergence speed on such unimodal fitness functions.
In the context of EAs, a natural direction for further comparison of the above cited mutations, is to investigate whether the presence of populations changes the relative efficiency of these mutations on the OneMax function. This may give insights in the role of populations, widely demonstrated in practice. Such study would be closer to practical concerns as populations are at the core of evolutionary computation.

Aiming at predicting EA behavior, the numerical simulations of the second chapter, suggest that average statistical indicators of fitness landscapes (resp. operators), are easily blinded by a large proportion of completely irrelevant information present in the random sample. This appeals for alternative sampling schemes. For example, the dynamic EA trajectory on a given landscape, can be seen as a particular sampling of the landscape.
Further, special attention is given to identify and compare different function classes, sharing the same indicator value. In fact, the key problem in designing a useful landscape indicator is to find a characterization of a broad class of fitness functions on which the EA behaves similarly. Clearly, the second chapter shows that the class of monotone[59] fitness functions is too small to satisfy this requirement. Note that finding a good definition of "**similar behavior**" is also a major problem. As a first step to meet this requirement, a method of comparison of EA behaviors (trajectory and convergence) has been proposed and experimented in II.2. EA *behavioral measures* are scaled using the same measures, but computed by using *distance fitness* functions. This shows that **genotypic distances** are particularly useful to scale and compare EA's *behavioral measures*.

However, the choice of genotypic distance, relevant with respect to the evolution process, is a crucial and non-trivial issue. The third chapter suggests that the ability of EAs to optimize **distance landscapes** –within a time of bounded complexity in problem size– is a good criterion for the choice of a genotypic distance. Other criteria are given by the relation to the minimal number of elementary mutation applications that transform an individual into another, further, by the dispersion of random points in the scatter 'phenotypic $\times$ genotypic

[59] wrt. to the distance to optimum

distance-to-optimum' plot, or by the distance sensitivity to minimal genotypic changes. Note finally that genotypic distances are also useful to compare initialization procedures, and to tune their parameters (better procedures yield random samples with a higher dispersion the fitness $\times$ distance space). Some indicators are proposed for that purpose.

On the other hand, the choice of a representation for a given problem cannot be decided from the static analysis of the landscape related to the genotypic distances (even designed in relation to mutation). Rather, the atypical properties of evolution operators are responsible for the difference in EA's performance. The *offspring ranking* method demonstrates useful to compare atypical operators properties in order to choose a good representation for the real-world inclusion problem.

The starting point of the third chapter is the mechanical problem of inclusion identification, and the choice of a representation among the *holes* or the *Voronoï* variable-length representations. In spite of their high degeneracy and epistasis, to the best of our knowledge, the variable length representations studied here are the best choice for the mechanical problem at hand. Among these variable length representations, an heuristic for the *a priori* choice of a representation is outlined below. The Voronoï representation seems the most robust choice in the absence of any a priori information. Otherwise, if the solution contains non-equal quantities of both materials, the Hole representation is also efficient (when the holes correspond to the minority material).

Note that the methodology used in the last chapter, in order to derive useful hints for further tuning of the algorithm parameters, is not specific for the inclusion problem, nor for shape representations, but have already been successful on a host of binary functions. Hence, one might expect that such tools (FD plot, *Offspring ranking*, adequacy between distance and evolution), may be helpful for identifying further problem dependent heuristics.

We may again summarize the key issues that emerge from these studies:

1. Genotypic distances are particularly useful to analyze, scale and compare EA behaviors. Genotypic distances are also useful to compare initialization procedures, and to tune initialization parameters (better procedures yield random samples with a big dispersion the fitness $\times$ distance plot). However, it is crucial to choose a genotypic distance that is relevant with respect to the evolution process.

2. Long time behavior of EAs on such big search space, is not reflected by typical properties (ex. expected behavior of both mutations in the first chapter, or expected fitness improvement of operators, or average indicators), but is mainly due to the **deviant atypical properties**. A direct consequence is that the fitness landscape of EA operators is related in a non trivial manner to the exploration mechanism.

   $\longrightarrow$ The *long k-path* problem points out a **caricatural situation** where independent parallel (1+1)-ES processes perform a lot better with no migration than they perform with migration of fit individuals between the processes. Equivalently, increasing the population size decreases convergence speed.

3. The key problem in designing a useful landscape indicator is to find a characterization of a broad class of fitness functions on which the EA behaves similarly. Moreover, finding a good definition of "similar behavior" is also a major problem.

$\longrightarrow$ Fitness functions with "similar EA behavior" can be identified by comparing a number of *behavioral measures* derived from EA run, scaled using the same measures, when computed on the corresponding *distance fitness* functions.

However, the efficiency of this comparison method relies on the choice of a genotypic distance on the search space, which can be non-trivial in non-binary representations, as illustrated in the third chapter.

We may finally mention some of the open issues, implicitly raised by these studies, and that shall be left for future work.

First, can population-based search enhance convergence on unimodal fitness functions, since these functions are optimizable by a single Hill Climber; Can we quantify the benefit of using niching selection and/or crossover operators? Does populations change the exploration properties of deterministic and stochastic mutations on the OneMax function?

Second, can we identify further GA-easiness classes beyond monotone (and unimodal) fitness functions. Note that within the class of functions involving second order interactions only (commonly named N2-landscapes), there already exist functions that prove difficult-to-intractable for GAs (as illustrated by the NNI sub-class of the second chapter).

Third, a thorough investigation of distance landscapes, comparing distances related to the evolution process is mandatory. It should give insights into the properties that most influence EA behavior. Although some hints are given in the third chapter, it is important to further identify the most important criteria, among those proposed (and possibly some others).
In particular, consider a sequence of operator-applications that transforms an individual into another. Which property better reflects EA behavior on distance landscapes: Is it the minimal size of such sequences, or the probability of occurrence of such a sequence, or the probability of occurrence of the most likely of these sequences?

Next section gives further issues for further work directions, and details a new research project.

Further Work

Some broad directions for further work are first sketched in the following, while a research work dealing with the design of a "similarity basis" is detailed in section 2.

1 Some Further Directions

The following points are either natural directions to extend or to improve the results of the thesis, or new directions aiming at designing more practical tool to detect EA stagnation, or to visualize EA-behavior. Finally, all these points can be used for the investigation of new representation spaces.

1. Does the difference in efficiency of the binary mutations –studied in the first chapter for (1,1)-ES and (1+1)-ES algorithms– still holds:

   - for parallel processes, in the presence of migration?
   - in the presence of populations?
   - when coupling mutation and crossover?

   Note that statistical physicists are able to approximate the cumulants (k^{th} order means) of successive populations for EAs using mutation and crossover in the Onemax case [Bor96]. At this point they are technically able to partially address the above issues. On the other hand, experimental investigation of these issues is also possible.

   $\rightarrow$ Such study would be closer to practical concerns as populations are at the core of evolutionary computation.

2. The existence of controlled paths for population based EAs using both crossover and stochastic mutation, appeals for the construction of EA instances that are able to find the shortcuts in such a path without *a priori* knowledge on hyperplanes structure. Selection schemes that maintain diversity in the population (e.g. crowding[DeJ75], sharing [GR87]) coupled with adaptive operators may help to design more powerful EAs.

3. The extension of the method of comparison of EA behavior –proposed in section II.2– to different representations requires some more work in order to find statistical comparison tools more robust to distance scale than simple ratios. For instance, $\frac{(1-\exp d)}{(1-expd')}$ is certainly not proportional to $\frac{d}{d'}$ when d and d' vary. Yet both d and $1 - \exp d$ yield distances in the space of variable length representations studied in the third chapter.

4. Developing **dynamic tools** based on the indicators studied in this work, to improve EA performance:

   - Adaptive tuning of the parameters of parametrized fitness functions (as in the context of penalty constrained optimization), by enforcing those parameters that yield a maximal dispersion of the populations in the FD-plot.
   - Extension of the *offspring ranking* indicator –proposed in section II.6– to design algorithms with adaptive choice of operators.

5. Developing **dynamic tools** based on additional indicators to give insights in EA behavior during the run:
   - In order to detect stagnation of the running algorithm, the **correlation between successive populations** (be it in the fitness space, the genotype space or the phenotype space) could be more useful than the usual autocorrelation function computed in the fitness space from random mutation walks.
   - In order to visualize some aspects of EA trajectory in the search space, a two dimensional plot could illustrate the radial and effective trajectory measures of EA populations for a given run (as in figure 20).
6. Study other representations, especially those dynamically **maintaining an adaptive linkage**, such as Messy GAs [Kar95a] or some transposition of the rectangles representation to binary spaces. This can be achieved using all comparison methods of chapter II. Ultimately these tools could be used together with Machine Learning techniques, to help developing new representations, by extracting both significant and measure-relevant features from runs of EAs on initial standard representations [Ger98].
7. Comparing **distance-landscapes** (i.e., the fitness is proportional to the distance-to-solution), and EA behavior on these landscapes, for different operator-based distances such us:

   - the minimal number of operator applications to get from one individual to the other,

   - the probability of the most likely sequence of operator applications that get from one individual to the other,

   - the probability that a random sequence of operator applications gets from one individual to the other.
8. Estimation of the **distribution of the sizes of the basins of attractions** of all local optima for a given landscape. A probabilistic technique has been developed for DNA sequencing problems in biology [Gar98], using random samples to obtain an approximation of the number of local optima, and the distribution of their basins of attraction. Some guidelines are detailed in Section 2.2.1, together with the following point:
9. Identifying a **representative family of reference functions**, with known landscape characteristics. The a posteriori comparison with the Hamming fitness function of section II.2 could then be applied to each reference function of that family, hopefully giving a more precise a posteriori description of EA-hardness, and of the landscape of the fitness at hand. A possible approach would be to use permutations in the search space applied to an easy well known function(as the Onemax) in order to generate reference functions, with known characteristics.

   This issue is addressed in the context of the research project of Further Work II.

The two last points respectively involve static and dynamic comparisons of landscapes, and can be coupled: the ultimate goal is to design adaptive optimization strategies. This strong research direction is detailed in the next pages.

2 Function Similarity to tune EA Parameters

In the following, some hints are given regarding the definition of a **new indicator** to compare fitness landscapes. This indicator is based on a theoretical estimation of the number of local optima that can be precisely derived for different sets of fitness functions.

Another (closely related) direction of research consists in designing a **family of reference functions** on which EAs exhibit a wide range of behaviors. Using the methodology proposed in section II.2, any fitness function could be classified among that family, hopefully giving some hints about the type and parameterization of Evolutionary Algorithms best suited for that particular function.
The main goal consists in using our knowledge of EA behavior on such a family of reference functions, for optimizing unknown fitness functions:

A set of landscape indicators: Suppose we have a set of landscape indicators in the context of evolutionary optimization. Such indicators can be obtained by comparing the behaviors of some EA instance when applied respectively to either function (following the guidelines of section II.2); eventually coupled with a static comparison of the distributions of the size of the basins of attraction of both functions' landscapes (on going work).
These indicators should allow one to identify similar functions, with respect to the indicators value.

In order to efficiently use such a set of indicators, we propose to construct a family of reference functions, representative of a wide range of EA behaviors. Knowing the characteristics of the reference function that share the same indicators values than a given fitness f, we might determine an efficient optimization strategy for f.

Designing a family of reference functions: As a first step towards the construction of such a family of reference functions (g_k), we propose to use a set of transformations (as permutations in the search space). Applied to a well known easy function g, each transformation yields a different fitness landscape. Hence, by construction the inverse transformation is known, and can be used to optimize new fitness functions that *resemble* a given reference function g_i.

$\longrightarrow$ We shall first propose possible transformations for the construction of the reference family, then we shall detail some candidate indicators.

2.1 Construction of the Family of Reference Functions

We now address a crucial point about the reference functions: supposing that there is a known efficient search strategy associated to each reference function limits the possible choices for the reference functions. Therefore, it is crucial for the reference family to be diversified and representative of a large number of landscapes.
<u>METHOD</u> To overcome this limitation, we propose to define each reference function as the transformation of an easy function (e.g. the fitness is the distance[60] to optimum), such that

[60] The third chapter of the thesis demonstrates that there are examples of distances (e.g. the *Hausdorff* distance) such that the associated *distance functions* are not easy to optimize for typical EAs. It also proposes heuristics to compare genotypic distance relevance with the operators of the EA at hand.

the inverse mapping is known. In the binary framework, this mapping can be seen as a **permutation** [61] of the points of the search space.

<u>Example:</u> Consider the initial fitness function $g(X) = l - d(X, Optimum)$, straightforward to optimize for algorithms that use a deterministic *1-bit-flip* mutation operator (flips one bit randomly chosen in the bitstring).
Let's define a permutation μ in $E = \{0,1\}^k$, $k = l/d$, for some integer d. For example (for $k = 2$),
$\mu(11) = 10$,
$\mu(10) = 00$,
$\mu(00) = 01$,
$\mu(01) = 11$.

Then for each $X \in E$, note,

$$X_\mu = \mu(X_1, \ldots X_k) \ \ldots \ \mu(X_{l-k+1}, \ldots X_l)$$

. In the previous example and $l = 6$, we have,

$$(11\,10\,00)_\mu = (10\,00\,01)$$

$\rightarrow$ This permutation highly modifies the fitness landscape, as $d(X, Y) = 1$ does not imply $d(X_\mu, Y_\mu) = 1$. The advantage of this particular kind of permutation is that the complexity of the inverse mapping is of 2^k rather than 2^l for mappings defined on the whole search space.

$\longrightarrow$ A set of such mapping μ_j defined in a $\{0,1\}^k$ leads to a family of binary fitness functions with different landscapes. For each of these reference functions, we know that the inverse of mapping μ_j turns it into a straightforward landscape for the simple 1-bit-flip Hill Climber.

2.2 Indicators of Fitness Functions

As a first step, we suggest two kinds of indicators of fitness functions, motivated from the point of view of EAs optimization:

1. <u>Static landscape indicators</u>: We propose a new comparison scheme of fitness landscapes: using a sample of the search space, we "derive" information about the distribution of the **sizes of the basins of attraction** of a given landscape, with means of a X2 statistical test. Other examples are landscape indicators studied in section II.3 of the thesis. But, defined as averages, they are of limited efficiency.

2. <u>Dynamic landscape indicators</u>: Function similarity from the point of view of EAs, can be described by the **difference of behavior of a given EA**, when applied to both functions. A method of comparison of EA behaviors has been proposed and experimented in section II.2 of the thesis [62]

In the following we only precise the new method for static landscape comparison. In fact, the dynamic comparison have already been detailed and experimented in section II.2 of the thesis.

[61] For instance, for each problem, there is a permutation on the search space that turns it into a – hill shaped– unimodal function optimizable by a Hill Climber.

[62] Section II.2 proposes some –fitness independent– behavioral measures that characterize EA trajectory and results, then proposes an absolute basis to scale the behavioral measures, for instance, the behavioral measures obtained when the fitness is defined as the distance to solution.

2.2.1 Static indicators: Comparison of the Basins of Attraction

Suppose that the search space $E = \{0,1\}^l$ (binary for the moment) can be defined as a partition E_1,...,E_N corresponding to the basins of attraction of the local maxima m_1,...,m_N of a given fitness function (Fig 72).

This can be achieved, simply by defining the basin of attraction of a local optima m, as the set of points x of the search space, such that a steepest ascent algorithm starting at x ends at point m.

• **An inverse problem: Identifying the sizes of the basins of attraction** This part is based on a probabilistic technique that has been developed for DNA sequencing problems in biology [Gar98]. Based on random sampling, it gives an empirical approximation of the number of local optima, and distribution of their basins of attraction. The method and some preliminary results are given in the following:

Let f be a fitness function. Our goal is to find the number N of local optima of f, as well as some information about the distribution of the size of the basins of attraction of the landscape. Without *a priori* knowledge about the landscape of f, we propose to

1. `choose a random sample` $(X_i)_{i=1,...,M}$ `uniformly in` E.
2. `perform a steepest ascent starting at each point` X_i.
3. `compute the number of local optima that have been reached by exactly` j `different initial points` X_i `(see example in figure 72).`

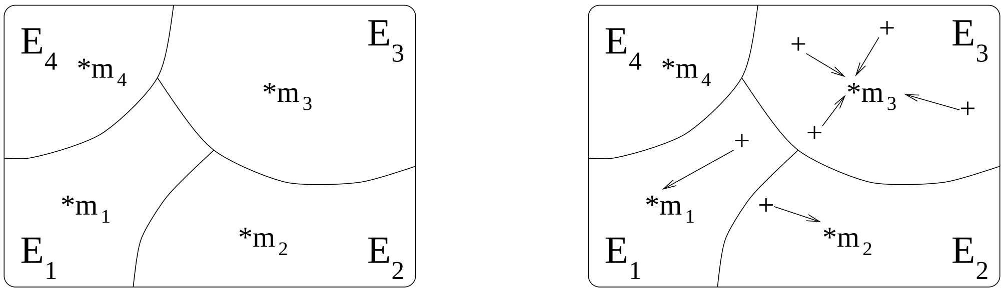


FIG 72 : *A partition of the search space* E *with* $N = 4$ *basins of attraction and* 4 *local maxima* m_1,...,m_4. *On leftmost picture,* 6 *random points have been chosen. After performing the steepest ascent runs, we find* $\beta_1 = 2$, $\beta_2 = 0, \beta_3 = 0$, $\beta_4 = 1$.

The main idea is described below:

Note first that $\bar{N} := \sum_{j=1}^{\infty} \beta_j$ is the number local optima actually detected. $\bar{N}$ is obviously a lower bound on the number of local optima of the landscape.
On the other hand, $\bar{N}$ gives less information that the set $(\beta_j)_{j\geq 1}$. We may exploit the information given by the experimental values of $(\beta_j)_{j\geq 1}$, in order to find an estimator of N better than $\bar{N}$, and some information about the number and size of the basins of attraction. In fact, the distribution of the variables β_j and the distribution of S –the size of the basins on attraction– are strongly related.

→ The comparison of simulated and theoretical distributions of the variables β_j (calculated for some particular landscapes) allows to find landscapes likely to resemble to f (in terms of the distributions of the sizes of the basins of attraction). A statistical $\mathcal{X}^2$ test can be used to achieve such a comparison.

$\longrightarrow$ We need to solve the direct problem of identifying the distribution of β for a family of possible distributions of the size of the basins.

• **The direct problem** Suppose that M points have been chosen uniformly in the search space E. The probability $p_{N,M}$ of sampling at least one point in each basin of attraction of E, can be computed for particular landscapes.

Proposition 2.1
Let's note $\alpha_j := |E_j|/|E|$ the normalized size of the basins of attraction. Then: 1. If the basins of attraction have the same size (Deterministic configuration), then for $N \gg 1$ and $M = N(\ln N + \ln \alpha)$, we have:

$$p_{N,M} = e^{-\alpha^{-1}}$$

2. If the size of the basins is completely random (Random configuration), then for $N \gg 1$ and $M = N^2\alpha$, $\alpha > 0$, we have:

$$p_{N,M} = e^{-\alpha^{-1}}$$

The fact that the required sample size ranges from $N \ln N$ in the Deterministic configuration to N^2 in the Random configuration, is due to the presence of very small basins, for the Random configuration, that are difficult to detect.

$\rightarrow$ The crucial point here is that the β_j values can be computed:
Under the Deterministic configuration, the distribution of the β_j follows a Poisson law:

$$\beta_{j,\infty} = Ne^{-M/N}\frac{1}{j!}\left(\frac{M}{N}\right)^j$$

Under the Random configuration, the distribution of the β_j follows a geometric law:

$$\beta_{j,1} = N\frac{1}{1+\frac{M}{N}}\left(\frac{\frac{M}{N}}{1+\frac{M}{N}}\right)^j$$

$\longrightarrow$ Given a fitness landscape, we can already use this result to determine whether the distribution of the sizes of the basins of attraction are more resemblant to the Random or to the Deterministic configuration: We can use a $\mathcal{X}^2$ statistical test to compare the simulated distribution of β for f, to each of the distributions derived above.

However, this result can be improved with some more work: we shall consider a family of γ-parametrized laws for S (normalized sizes of the basins), and derive as a function of γ, the law of the corresponding β. For example, we shall consider the parametrized density:

$$p_\gamma(z) = \frac{\gamma^\gamma}{\Gamma(\gamma)}z^{\gamma-1}e^{-\gamma z}.$$

The configurations A et D described above correspond to particular cases of the family p_γ. For $\gamma = 1$, we get the configuration R, whereas for $\gamma \to \infty$ we get the configuration D.

3 Discussion

To sum up, our main goal is to determine EA variants likely to efficiently optimize a given fitness function f. To achieve this, we first construct (section 2.1) a family of fitness functions for which we know an efficient optimization strategy (by construction, we know the mapping that turns it into an easy function). Then, we use the optimization strategy corresponding to the reference function that most resembles f (in terms of landscape, and EA behavior on this landscape as explained in section 2.2).

Of course, This optimization process can be iterated, eventually in the same time as the family of reference functions evolves. We then perform coevolution between the optimization of f and the optimization of the strategy of optimization ...
Note however that we still need to investigate the relevance of the indicators proposed above, and find alternative ones in the context of EAs. Note that indicators can also be coevolved dynamically. This direction may be worth trying as the relevance of indicators might vary with different problem domains ...
Note finally that a number of issues require further investigation :

- How does the choice of set of permutations influence the diversity of the family of reference functions.
- How to define significant and relevant indicators in the context of EAs? Note that the proposed indicators, suffer a number of drawbacks :
  1. Functions that have the same distribution for the size of the basins of attraction, still can give different EA behaviors.
  2. Functions that are similar in terms of "the behavior of some EA instance", aren't necessarily similar for a different algorithm, or in a different representation.
- To which extent the coupling of such indicators can improve the reliability of the optimization.

Hence a number of fundamental issues have to investigated in order to guide numerical implementation. On the other hand, we can use numerical simulation to guide intuition for further theoretical investigation.

Appendix

A Appendix: Test cases

OneMax problem: In the well-known *OneMax* problem, the fitness function (to maximize) is simply the number of 1's in the bitstring. Hence, the fitness is equal to l minus the distance to the global optimum:

$$f(x) = l - d(x, (1, \dots, 1)$$

The **Zeromax** problem is defined analogously by maximizing

$$f(x) = l - d(x, (0, \dots, 0)$$

Distance fitness functions: in the case of a minimization problem, the fitness is the distance to the optimum . We say that f is a *distance fitness* to x^*, when

$$f(x) = d(x, x^*)$$

In the case of a maximization problem, the fitness is $d_{max} - d(x, x^*)$, where d_{max} is the maximal distance value between two points of the search space.

- *Hamming fitness functions*, are defined in the binary space, with the *Hamming distance.*

- *Hausdorff fitness functions*, are defined with a *Hausdorff* distance.

The LongPath problem: The LongPath problem [HGD94] was purposely designed to discourage standard hill-climbers (1-bit-mutation (1+1)-ES) though being gently unimodal.

The fitness landscape is composed of two different regions; the first one is a Zeromax landscape, and culminates at $(0, \dots, 0)$; over that smooth hill, a path of exponential length in the length of the bitstring, climbs toward the global optimum by 1-bit steps. The Hamming distance between two non-adjacent points on the path is greater than 2: local hill-climber will have to climb all the way up the path, while global methods like GAs can take short-cuts to reach the optimum quickly. The optimum is at point $(1100\dots00)$.

The Long k-Path problem: Rudolph [Rud94b] presents a generalization of Horn's longpath to a class of fitness functions called *long k-paths.*

The long k-path $P(l, k)$ is defined by a recursion on the length l of the strings, with $P(1, 1) = \{0, 1\}$ as the base path. Given the long k-path $P(l, k)$, $P(l + k, k)$ is constructed as follows: A sub-path S_0 of $P(l + k, k)$ is created by taking the path $P(l, k)$ and by prepending k zeros to each point on $P(l, k)$. A sub-path S_1 of $P(l + k, k)$ is created by taking $P(l, k)$ *in reverse order* and prepending k ones to each point on the reversed path. A third path B consists of $k - 1$ points created by prepending the following substrings of length k to the last point in the path $P(l, k)$: $0\dots01$, $0\dots011$, $\dots$, $001\dots11$ and $01\dots11$. The resulting long k-path $P(l+k, k)$ on strings of length $l+k$ is then defined by the concatenation of S_0, the "bridge path" B and S_1. Horn's longpath is the long k-path for $k = 2$.

As an example, consider the construction of the path $P(7,3)$. Starting from $P(1,3) = \{0,1\}$, we obtain the sequence $P(4,3)$:

$$\underbrace{0000_1 \quad 0001_2}_{S_0} \quad \underbrace{0011_3 \quad 0111_4}_{Bridge} \quad \underbrace{1111_5 \quad 1110_6}_{S_1} \tag{A.1}$$

from which the path $P(7,3)$ is constructed:

$$\begin{array}{lll} 0000000_1 & 0011110_7 & 1111110_9 \\ 0000001_2 & 0111110_8 & 1111111_{10} \\ 0000011_3 & & 1110111_{11} \\ 0000111_4 & & 1110011_{12} \\ 0001111_5 & & 1110001_{13} \\ 0001110_6 & & 1110000_{14} \\ \\ S_0 & \text{Bridge} & S_1 \end{array} \tag{A.2}$$

- The Hamming distance between two consecutive points of the path is 1.
- A mutation of $i < k$ bits can only lead to a point which is i positions away from the original point. To jump over more than k consecutive positions in the path, all the bits of at least one of the blocks $2j+1, \ldots, 2(j+1)$, for $j \in \{0, \ldots, (l-1)/k\}$, have to be flipped. Such a jump is called a *shortcut.*
- The length of the long k-path is given by $|P(l,k)| = (k+1)2^{(l-1)/k} - k + 1$. Note that for a fixed string length, the path becomes shorter with increasing k.

Gray-coded Baluja functions F1 and F3: Consider the functions of k variables $(x_1, \ldots x_k)$, with $x_i \in [-2.56, 2.56]$ [Bal95]:

$$F_1(\vec{x}) = \frac{100}{10^{-5} + \sum_{i=0}^{k-1} |y_i|}, y_0 = x_0 \text{ and } y_i = x_i + y_{i-1} \text{ for } i = 1, \ldots, k-1$$

$$F_3(\vec{x}) = \frac{100}{10^{-5} + \Sigma_i |.024 * (i+1) - x_i|}$$

They reach their maximum value of 10^7 at point $(0, \ldots 0)$. The Gray-encoded versions of F_i, with 100 variables encoded on 9 bits each are considered.

The 4 peaks problem: In the FourPeaks problem [BC95], the fitness is the maximum of the length of the sequence of 0's starting at the first bit position and the length of the sequence of 1's ending at the last position, plus a reward of n if both sequences are larger than a given threshold T. There are two global optima, made of a block of 0's followed by a block of 1's, of lengths T and $n-T$ or $n-T$ and T. However, there are two local optima, the all 1's and all 0's bitstrings, from which it is difficult for a GA to escape. The most difficult instance of that problem (i.e. $T = n/4$) is used here.

The Ugly problem: The Ugly problem [Whi91] is defined from an elementary deceptive 3-bit problem (F(x) = 3 if x = 111; F(x) = 2 for x in $0\star\star$, and F(x) = 0 otherwise). The – deceptive – full problem is composed of 300 concatenated elementary deceptive problems. The maximum is the all 1's bitstring.

The Royal Road problem: The 64-bits Royal Road problem used here was conceived to study into details the combination of features most adapted to GA search (laying a *Royal Road*). A precise definition as well as an analysis of the unexpected difficulties of this problem can be found in [MH93].

B Appendix: Some useful results of probability theory

Markov property

A Markov chain $Z(n)$ satisfies the so-called weak and strong Markov properties [Rev75].
Weak Markov property
For any integer n_0 and for any function ϕ which depends on the whole path:

$$\mathbb{E}\left[\phi\left((Z(n))_{n\geq n_0}\right)/(Z(n))_{n\leq n_0}\right] = \mathbb{E}\left[\phi\left((Z(n))_{n\geq n_0}\right)/Z(n_0)\right].$$

Besides, if the chain is homogeneous (i.e. its transition matrix is time-independent), then:

$$\mathbb{E}\left[\phi\left((Z(n))_{n\geq n_0}\right)/Z(n_0)\right] = \mathbb{E}_{Z(n_0)}\left[\phi\left((Z(n))_{n\geq 0}\right)\right].$$

Strong Markov property
For any stopping time τ and for any function ϕ which depends on the whole path:

$$\mathbb{E}\left[\phi\left((Z(n))_{n\geq \tau}\right)/(Z(n))_{n\leq \tau}\right] = \mathbb{E}\left[\phi\left((Z(n))_{n\geq \tau}\right)/Z(\tau)\right].$$

Besides, if the chain is homogeneous, then:

$$\mathbb{E}\left[\phi\left((Z(n))_{n\geq \tau}\right)/Z(\tau)\right] = \mathbb{E}_{Z(\tau)}\left[\phi\left((Z(n))_{n\geq 0}\right)\right].$$

If τ is a stopping time of the form $\tau = \inf\{n, Z(n) = k\}$, then:

$$\mathbb{E}_{Z(\tau)}\left[\phi\left((Z(n))_{n\geq 0}\right)\right] = \mathbb{E}_k\left[\phi\left((Z(n))_{n\geq 0}\right)\right].$$

Remark B.1
If ϕ depends only on one point, say $Z(n_0+k)$, and is of the form $\mathbb{1}_{z\geq a}$, then the above expectations read as probabilities:

$$\mathbb{E}[\phi((Z(n))_{n\geq n_0})/(Z(n))_{n\leq n_0}] = \mathbb{P}\left(Z(n_0+k)\geq a/Z(n_0)\right)$$

Renewal theory

Let $(N_j)_{j\in\mathbb{N}}$ be a sequence of independent and identically distributed random variables with distribution $\mathbb{P}(N_1 = k) = p_k$, $k\in\mathbb{N}$.

If we denote by S_n the partial sums $S_n = \sum_{j=1}^{n} N_j$, then for any k, there exists a unique n_k such that $S_{n-1}\leq k < S_n$. The sequence n_k then satisfies [GS86]:

$$\frac{\mathbb{E}[n_k]}{k} \overset{k\to\infty}{\longrightarrow} \frac{1}{\mathbb{E}[N_1]}, \tag{B.1}$$

$$\mathbb{P}\left(\exists n, S_n = k\right) \overset{k\to\infty}{\longrightarrow} \frac{1-p_0}{\mathbb{E}[N_1]}. \tag{B.2}$$

The lack of memory of exponential random variables

Lemma B.2
Let τ be an exponential random variable with mean m. Then for any stopping time τ' (for instance an independent exponential random variable), the distribution of τ given $\tau>\tau'$ is still an exponential distribution with mean m.

Proof. This is proved in [Fel71, Vol 1, ch. III-9]. □

A lemma about exponential random variables

The following characterization of exponential random variables is well-known in probability theory. We express it in a suitable way for our purpose.

Lemma B.3
Let T be a $\mathbb{R}^+$-valued random variable. T obeys an exponential distribution if and only if for every $a, b \geq 0$:

$$\mathbb{P}(T \geq a + b / T \geq a) = \mathbb{P}(T \geq b). \tag{B.3}$$

Proof. If T obeys an exponential distribution, then it obviously satisfies (B.3). Let us now consider a $\mathbb{R}^+$-valued random variable satisfying (B.3). We denote:

$$\alpha := \mathbb{P}(T \geq 1)$$

Step 1. For any $n \in \mathbb{N}$, $\mathbb{P}(T \geq n) = \alpha^n$.
Applying (B.3) with $b = 1$ and $a = n$ yields:

$$\mathbb{P}(T \geq n+1) = \mathbb{P}(T \geq n + 1/T \geq n)\mathbb{P}(T \geq n) = \alpha \mathbb{P}(T \geq n).$$

By induction with respect to n we get the result.
Step 2. For any $q \in Q$, $\mathbb{P}(T \geq q) = \alpha^q$.
Let $q \in \mathbb{Q}$. q is of the form $q = r/n$ where r and n are integers. Applying (B.3) with $b = q$ and $a = mq$ yields:

$$\mathbb{P}(T \geq q(m+1)) = \mathbb{P}(T \geq q)\mathbb{P}(T \geq mq).$$

By induction with respect to $m = n-1, ..., 0$, we get

$$\mathbb{P}(T \geq qn) = \mathbb{P}(T \geq q)^n.$$

Since $qn = r$ is an integer we have $\mathbb{P}(T \geq qn) = \alpha^r$, which establishes the result.
Step 3. For any $t \in \mathbb{R}^+$, $\mathbb{P}(T \geq t) = \alpha^t$.
The function $t \mapsto \mathbb{P}(T \geq t)$ is monotone over $\mathbb{R}^+$. Step 2 then obviously implies the result of Step 3. □

Stirling's formula

If n is a large integer, then [AS65]:

$$n! \simeq \left(\frac{n}{e}\right)^n \sqrt{2\pi n}$$

Laplace technique

If $f \in \mathcal{C}^3(a, b)$ reaches its minimal value at c which lies inside the interval (a, b), if $f''(c) < 0$, and if $g \in \mathcal{C}^0(a, b) \cap L^1(a, b)$, then:

$$\left| \sqrt{l} \int_a^b e^{-lf(s)} g(s) ds - \frac{g(c)\sqrt{2\pi}}{\sqrt{|f''(0)|}} \right| \xrightarrow{l \to 0} 0$$

LIST OF PAPERS

Published papers in journals/book chapters (2) or proceedings of conferences with review comitees:

[1] M. Schoenauer, L. Kallel, and F. Jouve. Mechanical inclusions identification by evolutionary computation. *European Journal of Finite Elements*, 5(5-6):619–648, 1996.

[2] M. Schoenauer, F. Jouve, and L. Kallel. Identification of mechanical inclusions. In D. Dasgupta and Z. Michalewicz, editors, *Evolutionary Computation in Engeneering*, pages 477–494. Springer Verlag, 1997.

[3] L. Kallel and M. Schoenauer. A priori predictions of operator efficiency. In *Artificial Evolution'97*, LNCS, pages 287–299. Springer Verlag, October 1997.

[4] L. Kallel and M. Schoenauer. Alternative random initialization in genetic algorithms. In Th. Bäeck, editor, *Proceedings of the* 7^{th} *International Conference on Genetic Algorithms*. Morgan Kaufmann, 1997.

[5] L. Kallel and B. Naudts. Longpaths for the genetic algorithm. In *Foundations of Genetic Algorithms 5*. Morgan Kaufmann, 1998. to appear.

[6] L. Kallel. Inside ga dynamics: Ground basis for comparision. In *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*. Morgan Kaufmann, 1998.

[7] J. Garnier, L. Kallel, and M. Schoenauer. Rigourous results of the first hitting times some Markov chains. In *Journal of Evolutionary Computation*. 1999. to appear.

Papers submitted to journals:

[8] B. Naudts and L.Kallel. Some facts about so called ga-hardness measures. Technical Report 379, CMAP, Ecole Polytechnique, Mars 1998. submitted.

[9] J. Garnier and L. Kallel. Statistical distribution of the convergence time for long k-path problems. Technical Report 378, CMAP, Ecole Polytechnique, Avril 1998. submitted.

Other Papers:

[10] L. Kallel, B. Naudts and M. Schoenauer. Generic functions with a fixed fitness–distance relation. 1999. submitted.

[11] L. Kallel and M. Schoenauer. Genotypic distances for variable length representations and comparison with the *hausdorff* distance. 1998. in progress.

References

[Ack87] D. H. Ackley. *A connectionnist Machine for Genetic Hill-climbing.* Kluwer, Boston, 1987.

[Aga97] Alexandru Agapie. Genetic algorithms: minimal conditions for convergence. In J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution'97*, LNCS. Springer Verlag, October 1997.

[Alt95] L. Altenberg. The schema theorem and Price's theorem. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 23–49, San Mateo, CA, 1995. Morgan Kaufmann.

[Alt97] L. Altenberg. Fitness distance correlation analysis: an instructive counterexample. In Th. Bäeck, editor, *Proceedings of the* 7^{th} *International Conference on Genetic Algorithms*, pages 57–64. Morgan Kaufmann, 1997.

[Ank91] Carol A. Ankenbrandt. An extention to the theory of convergence and approof of the time complexity of genetic algorithms. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 53–68. Morgan Kaufmann, 1991.

[Ant89] J. Antonisse. A new interpretation of schema notation that overturns the binary encoding constraint. In J. D. Schaffer, editor, *Proceedings of the* 3^{rd} *International Conference on Genetic Algorithms*, pages 86–91. Morgan Kaufmann, June 1989.

[AS65] M. Abramowitz and I. Stegun. *Handbook of mathematical functions.* Dover publications, New York, 1965.

[ASP93] P. J. Angeline, G. M. Saunders, and J. B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(2):86–91, 1993.

[B9̈5] T. Bäck. *Evolutionary Algorithms in theory and practice.* New-York:Oxford University Press, 1995.

[Bäc92] T. Bäck. The interaction of mutation rate, selection, and self-adaptation in genetic algorithm. In R. Manner and B. Manderick, editors, *Proceedings of the* 2^{nd} *Conference on Parallel Problems Solving from Nature*, pages 85–94. Morgan Kaufmann, 1992.

[Bäc93] T. Bäck. Optimal Mutation Rate in Genetic Search. In S. Forrest, editor, *Proceedings of the* 5^{th} *International Conference on Genetic Algorithms*, pages 2–8. Morgan Kaufmann, 1993.

[Bal95] S. Baluja. An empirical comparison of seven iterative and evolutionary function optimization heuristics. Technical Report CMU-CS-95-193, Carnegie Mellon University, 1995.

[BC95] S. Baluja and R. Caruana. Removing the genetics from the standard genetic algorithms. In A. Prieditis and S. Russel, editors, *Proceedings of ICML95*, pages 38–46. Morgan Kaufmann, 1995.

[Bey93] H.-G. Beyer. Toward a theory of evolution strategies: Some asymptotical results for the $(1, +\lambda)$-theory. *Evolutionary Computation*, 1(2):165–188, 1993.

[Bey94] H.-G. Beyer. Toward a theory of evolution strategies: The (μ, λ)-theory. *Evolutionary Computation*, 2(4):381–407, 1994.

[Bey95a] H.-G. Beyer. Toward a theory of evolution strategies: On the benefit of sex - the $(\mu/\mu, \lambda)$-theory. *Evolutionary Computation*, 3(1):81–111, 1995.

[Bey95b] H.-G. Beyer. Toward a theory of evolution strategies: Self-adaptation. *Evolutionary Computation*, 3(3):311–347, 1995.

[BNV97] D. Suys B. Naudts and A. Verschoren. Epistasis as a basic concept in formal landscape analysis. In ICGA97editors, editor, *ICGA97*, pages 65–72. mk, 1997.

[Bor84] G. Borgefors. Distance transformation in arbitrary dimention. *Computer Vision, Graphics, and Image Processing*, 27:321–345, 1984.

[Bor96] S. Bornholdt. Probing genetic algorithm performance of fitness landscapes. In R. K. Belew and M. D. Vose, editors, *Foundations of Genetic Algorithms 4*, pages 141–154. Morgan Kaufmann, 1996.

[Bra91] M. F. Bramlette. Initialization, mutation and selection methods in genetic algorithmes for function optimization. In R. K. Belew and L. B. Booker, editors, *Proceedings of the 4^{th} International Conference on Genetic Algorithms*, pages 100–108. Morgan Kaufmann, 1991.

[BS93] T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[BS96] T. Bäck and M. Schütz. Intelligent mutation rate control in canonical gas. In Z. W. Ras and M. Michalewicz, editors, *Foundation of Intelligent Systems 9th International Symposium, ISMIS '96*, pages 158–167. Springer Verlag, 1996.

[Bui94] H. D. Bui. *An introduction to inverse problems in Structural Mechanics.* CRC Press, Boca Raton, 1994.

[Bui95] H. D. Bui. Sur quelques problèmes inverses élastiques en mécanique de l'endommagement. In *Deuxième Colloque National en Calcul des Structures*, 1995.

[BY95] J.-D. Boissonnat and M. Yvinec. *Géométrie algorithmique.* Ediscience International, 1995.

[CDC96] U. Chakraborty, K. Deb, and M. Chakraborty. Analysis of selection algorithms: A markov chain approach. *Evolutionary Computation*, 4(2):133–168, Summer 1996.

[Cer94] R. Cerf. *Une théorie assymptotique des algorithmes génétiques.* PhD thesis, Université de Montpellier II, March 1994.

[Cer96] R. Cerf. An asymptotic theory of genetic algorithms. In J.-M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution*, volume 1063 of *LNCS*. Springer Verlag, 1996.

[Cho86] Debashish Chowdhury. *Spin glasses and other frustrated systems.* World Scientific, Singapore, 1986.

[Cia78] P. G. Ciarlet. *Mathematical Elasticity, Vol I : Three-Dimensional Elasticity.* North-Holland, Amsterdam, 1978.

[CJ91] Robert J. Collins and David R. Jefferson. Selection in massively parallel genetic algorithm. In R. K. Belew and L. B. Booker, editors, *Proceedings of the 4^{th} International Conference on Genetic Algorithms*, pages 249–256. Morgan Kaufmann, 1991.

[CJ95] C. D. Chapman and M. J. Jakiela. Genetic algorithm-based structural topology design with compliance and topology simplification considerations. *Journal of Mechanical Design*, 1995. Submitted.

[CL96] J. Culberson and Jonathan Lichtner. On searching 'alpha'-ary hypercubes and related graphs. In R. K. Belew and M. D. Vose, editors, *Foundations of Genetic Algorithms 4*, pages 263–290. Morgan Kaufmann, 1996.

[Con94] A. Constantinescu. *Sur l'identification des modules élastiques.* PhD thesis, Ecole Polytechnique, June 1994.

[CS88] R. A. Caruna and J. D. Schaffer. Representation and hidden bias : Gray vs binary coding for genetic algorithms. In *Proceedings of ICML-88, International Conference on Machine Learning.* Morgan Kaufmann, 1988.

[CSJ94] C. D. Chapman, K. Saitou, and M. J. Jakiela. Genetic algorithms as an approach to configuration and topology design. *Journal of Mechanical Design*, 116:1005–1012, 1994.

[Dav90] Y. Davidor. Epistasis variance: A viewpoint on representations, GA hardness, and deception. *Complex systems*, 4:369–383, 1990.

[Dav91a] Y. Davidor. Epistasis variance: a viewpoint on ga-hardness. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 23–35. Morgan Kaufmann, 1991.

[Dav91b] Y. Davidor. *Genetic algorithms and robotics:a heuristic strategy for optimisation*. Signapore: World Scientific Publishing Co. (Pte.) Ltd., 1991.

[DeJ75] K.A. DeJong. *The Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Ann Harbor, 1975. *Dissertation Abstract International*, 36(10), 5140B. (University Microfilms No 76-9381).

[DeJ92] K. A. DeJong. Are genetic algorithms function optimizers ? In R. Manner and B. Manderick, editors, *Proceedings of the* 2^{nd} *Conference on Parallel Problems Solving from Nature*, pages 3–13. North Holland, 1992.

[dG90] H. de Garis. Genetic programming : building artificial nervous systems using genetically programmed neural networks modules. In R. Porter and B. Mooney, editors, *Proceedings of the* 7^{th} *International Conference on Machine Learning*, pages 132–139. Morgan Kaufmann, 1990.

[DJW98] Stefan Droste, Thomas Jensen, and Ingo Wegener. A rigorous comlexity analysis of the (1+1) evolutionary algorithm for linear functions with boolean inputs. In *Proceedings of the Fifth IEEE International Conference on Evolutionary Computation*. IEEE Press, 1998.

[DP93] T. E. Davis and J. C. Principe. A markov chain framework for the simple genetic algorithm. *Evolutionary Computation*, 1(3):269–292, 1993.

[DS95] K. A. DeJong and J. Sarma. On decentralizing selection algorithms. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference on Genetic Algorithms*, pages 17–23. Morgan Kaufmann, 1995.

[DSG95] K. A. DeJong, W. M. Spears, and D. M. Gordon. Using markov chains to analyse gafos. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 115–137. Morgan Kaufmann, 1995.

[DW91] R. Das and D. Whitley. The only challenging problems are deceptive: Global search by solving order-1 hyperplanes. In R. K. Belew and L. B. Booker, editors, *Proceedings of the* 4^{th} *International Conference on Genetic Algorithms*, pages 166–173. Morgan Kaufmann, 1991.

[DWP95] K. Mathias D. Whitley and L. Pyeatt. Hyperplane ranking in simple genetic alghoritms. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference on Genetic Algorithms*, pages 231–238. Morgan Kaufmann, 1995.

[EAH91] A.E. Eiben, E.H.L. Aarts, and K.M. Van Hee. Global convergence of genetic algorithms: a markov chain analysis. In Hans-Paul Schwefel and Reinhard Männer, editors, *Proceedings of the* 1^{st} *Parallel Problem Solving from Nature*, pages 4–12. Springer Verlag, 1991.

[EK86] S. Ethier and T. Kurtz. *Markov processes*. Wiley, New York, 1986.

[ES93] L. Eshelman and J. D. Schaffer. Real-coded genetic algorithms and interval-schemata. In L. D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 187–202, Los Altos, CA, 1993. Morgan Kaufmann.

[Fel71] W. Feller. *An introduction to probability theory and its applications*. Wiley, New York, 1971.

[FF96] C. M. Fonseca and P. J. Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*, number 1141 in LNCS, pages 584–593. Springer Verlag, Sept. 1996.

[FG96] D. B. Fogel and A. Ghozeil. Using fitness distributions to design more efficient evolutionary computations. In T. Fukuda, editor, *Proceedings of the Third IEEE International Conference on Evolutionary Computation*, pages 11–19. IEEE, 1996.

[FM91] S. Forrest and M. Mitchell. The performance of genetic algorithms on walsh polynomials: Some anomalous results and their explanation. In R. K. Belew and L. B. Booker, editors, *Proceedings of the* 4^{th} *International Conference on Genetic Algorithms*, pages 182–189. Morgan Kaufmann, 1991.

[FM93] S. Forrest and M. Mitchell. Relative building-block fitness and the building-block hypothesis. *Foundations of Genetic Algorithms 2*, pages 109–126, 1993.

[FM96] B. Freisleben and P. Merz. New genetic local search operators for the tsp. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*, LNCS 1141, pages 890–899. Springer Verlag, 1996.

[Fog92] D. B. Fogel. An analysis of evolutionary programming. In D. B. Fogel and W. Atmar, editors, *Proceedings of the* 1^{st} *Annual Conference on Evolutionary Programming*, pages 43–51. Evolutionary Programming Society, 1992.

[Fog95] D. B. Fogel. *Evolutionary Computation. Toward a New Philosophy of Machine Intelligence.* IEEE Press, Piscataway, NJ, 1995.

[FOW66] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution.* New York: John Wiley, 1966.

[Gar98] J. Garnier. Approche probabiliste du séquençage à grande échelle, 1998. note technique.

[GD91] D. E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

[GDC92] D. E. Goldberg, K. Deb, and J. H. Clark. Genetic algorithms, noise and the sizing of populations. *Complex Systems*, 6:333–362, 1992.

[Ger98] J.S. Gero. Adaptive systems in designing: New analogies from genetics and developmental biology. In I. Parmee, editor, *Adaptive Computing in Design and Manufacture*, pages 3–12. Springer Verlag, 1998.

[GKD89] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivations, analysis and first results. *Complex Systems*, 3:493–530, 1989.

[GKS98] J. Garnier, L. Kallel, and M. Schoenauer. Rigourous results of the first hitting times some Markov chains. Technical Report 389, CMAP, Ecole Polytechnique, Juin 1998. submitted.

[GMP96] C. Ghaddar, Y. Maday, and A. T. Patera. Analysis of a part design procedure. *Submitted to Nümerishe Mathematik*, 1996.

[Gol87] D. E. Goldberg. Simple genetic algorithms and the minimal deceptive problem. In Davis L., editor, *Genetic Algorithms and simulated annealing.* Morgan Kaufmann, 1987.

[Gol89] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison Wesley, 1989.

[Gol91] D. E. Goldberg. A theory of virtual alphabets. In Hans-Paul Schwefel and Reinhard Männer, editors, *Proceedings of the* 1^{st} *Parallel Problem Solving from Nature*, pages 13–22, 1991.

[Gol93] D. E. Goldberg. Making genetic algorithms fly:a lesson from the wright brothers. *Advanced Technology for Developpers*, 2:1–8, February 1993.

[Gol98] D. E. Goldberg. The race, the hurdle and the sweet spot: Lessons from genetic algorithms for the automation of design innovation and creativity. Technical Report 98007, Illinois University, 1998.

[GR87] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multi-modal function optimization. In J. J. Grefenstette, editor, *Proceedings of the* 2^{nd} *International Conference on Genetic Algorithms*, pages 41–49. Lawrence Erlbaum Associates, 1987.

[GR91] D. E. Goldberg and M. Rudnick. Genetic algorithms and the variance of fitness. *Complex Systems*, 5:266–278, 1991.

[Gre87] J. J. Grefenstette. Incorporating problem specific knowledge in genetic algorithms. In Davis L., editor, *Genetic Algorithms and Simulated Annealing*, pages 42–60. Morgan Kaufmann, 1987.

[Gre93] J. J. Grefenstette. Deception considered harmful. In L. D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 75–92. Morgan Kaufmann, 1993.

[Gre95a] J. J. Grefenstette. Predictive models using fitness distributions of genetic operators. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 139–161. Morgan Kaufmann, 1995.

[Gre95b] J. J. Grefenstette. Virtual genetic algorithms: First results. Technical Report AIC-95-013, Navy Center for Applied Research in Artificial Intelligence, February 1995.

[Gru94] F. Gruau. *Neural Network Synthesis using Cellular encoding and the Genetic Algorithm.* PhD thesis, Ecole Normale Superieure de Lyon, 1994.

[GS86] I. Guikhman and A. Skorokhod. Introduction à la théorie des processus aléatoires. Mir, Moscou, 1986.

[HG95] J. Horn and D.E. Goldberg. Genetic algorithms difficulty and the modality of fitness landscapes. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 243–269. Morgan Kaufmann, 1995.

[HGD94] J. Horn, D.E. Goldberg, and K. Deb. Long path problems. In Y. Davidor, H.-P. Schwefel, and R. Manner, editors, *Proceedings of the* 3^{rd} *Conference on Parallel Problems Solving from Nature*, pages 149–158. Springer Verlag, 1994.

[Hil90] H. J. Hilhorst. A theory of aging in spin glasses. In H. van Beijeren, editor, *Fundamental Problems in Statistical Mechanics*, volume VII, pages 311–319. Elsevier Science Publishers, 1990.

[Hol62] J. H. Holland. Outline for a logical theory of adaptive systems. *Journal of the association of computing machinery*, 3, 1962.

[Hol75] J. H. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, 1975.

[HR96a] C. Höhn and C. Reeves. Are long path problems hard for genetic algorithms? In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*, volume 1141 of *LNCS*, pages 134–143. Springer Verlag, 1996.

[HR96b] J. Horn and C.R Reeves. The genetic crossover landscape for the *onemax* problem. In J.Alander, editor, *Proceedings of the* 2^{nd} *Nordic Workshop on Genetic Algorithms and their Applications*, pages 27–43. University of Vaasa Press,Vaasa, Finland, 1996.

[Iba96] H. Iba. Random tree generation for genetic programming. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*, volume 1141 of *LNCS*, pages 144–153. Springer Verlag, 1996.

[Ios] M. Iosifescu. Finite Markov processes and their applications.

[Jen92] E. Jensen. *Topological Structural Design using Genetic Algorithms.* PhD thesis, Purdue University, November 1992.

[JF95] T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference on Genetic Algorithms*, pages 184–192. Morgan Kaufmann, 1995.

[JM91] C. Z. Janikow and Z. Michalewicz. An experimental comparison of binary and floating point representations in genetic algorithms. In R. K. Belew and L. B. Booker, editors, *Proceedings of the 4^{th} International Conference on Genetic Algorithms*, pages 31–36. Morgan Kaufmann, 1991.

[Jon95] T.C. Jones. *Evolutionary Algorithms, Fitness Landscapes and Search.* PhD thesis, University of New Mexico, Albuquerque, NM, 1995.

[Jr94] K. E. Kinnear Jr. A perspective on gp. In Jr K. E. Kinnear, editor, *Advances in Genetic Programming*, pages 3–19. MIT Press, Cambridge, MA, 1994.

[JSC91] J. Jäckle, R. B. Stinchcombe, and S. Cornell. Freezing of nonequilibrium domain structures in a kinetic Ising model. *J. Stat. Phys.*, 62:425–433, 1991.

[Kan96] C. Kane. *Algorithmes génétiques et Optimisation topologique.* PhD thesis, Université de Paris VI, July 1996.

[Kar95a] H. Kargupta. *SEARCH, polynomial complexity, and the fast messy genetic algorithm.* PhD thesis, University of Illinois at Urbana-Champaign, 1995.

[Kar95b] H. Kargupta. Signal-to-noise, crosstalk, and long range problem difficulty in genetic algorithms. In L. J. Eshelman, editor, *Proceedings of the 6^{th} International Conference on Genetic Algorithms*, pages 193–200. Morgan Kaufmann, 1995.

[Kau89] S. A. Kauffman. *Lectures in the Sciences of Complexity*, volume I of *SFI studies*, chapter Adaptation on Rugged Fitness Landscape, pages 619–712. Addison Wesley, 1989.

[KG96] H. Kargupta and D. E. Goldberg. Serch, blackbox optimisation, and sample complexity. In R. K. Belew and M. D. Vose, editors, *Foundations of Genetic Algorithms 4*, pages 291–324. Morgan Kaufmann, 1996.

[KJS95] C. Kane, F. Jouve, and M. Schoenauer. Structural topology optimization in linear and nonlinear elasticity using genetic algorithms. In *Proceedings of the ASME 21st Design Automation Conference*. ASME, Boston, Sept. 1995.

[KM90] R. V. Kohn and A. McKenney. Numerical implementation of a variational method for electric impedance tomography. *Inverse Problems*, 6:389–414, 1990.

[KNS98] L. Kallel, B. Naudts, and M. Schoenauer. On functions with a fixed fitness distance relation. Technical Report 399, CMAP, Ecole Polytechnique, 1998.

[Knu81] D. E. Knuth. *The art of Computer Programming*, volume 2: Seminumerical Algorithms. Addison Wesley, 1981.

[Koz94] J. R. Koza. *Genetic Programming: On the Programming of Computers by means of Natural Evolution.* MIT Press, Massachussetts, 1994.

[KS] J.G. Kemeny and J.L. Snell. Finite Markov chains.

[KS95] C. Kane and M. Schoenauer. Genetic operators for two-dimensional shape optimization. In J.-M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution*, number 1063 in LNCS. Springer Verlag, Septembre 1995.

[KS96] C. Kane and M. Schoenauer. Topological optimum design using genetic algorithms. *Control and Cybernetics*, 25(5):1059–1088, 1996.

[Lez] P. Lezaud. *Etude de la convergence des processus Markoviens par perturbationde leur noyau.* PhD thesis, University of Paul Sabatier.

[LN95] P. Lee and T. Nagao. Hierarchical description of two dimensional shapes using a genetic algorithm. In D. B. Fogel, editor, *Proceedings of the Second IEEE International Conference on Evolutionary Computation*, 1995.

[LR93] S. J. Louis and G. J. E. Rawlins. Predicting convergence time for genetic algorithms. In L. D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 141–161. Morgan Kaufmann, 1993.

[LRN93] P. Ladeveze, N. Reynier, and D. Nedjar. Parametric corrections of finite element models using modal tests. In M. Tanaka and H. D. Bui, editors, *IUTAM Symposium on Inverse Problems in Engeneering Mechanics.* Springer Verlag, May 1993.

[LV91] G. E. Liepins and M. D. Vose. Deceptiveness and genetic algorithm dynamics. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 36–52. Morgan Kaufmann, 1991.

[M9̈2] H. Mühlenbein. How genetic algorithms really work: I. mutation and hill-climbing. In R. Manner and B. Manderick, editors, *Proceedings of the* 2^{nd} *Conference on Parallel Problems Solving from Nature*, pages 15–25. Morgan Kaufmann, 1992.

[Mah92] S. W. Mahfoud. Crowding and preselection revisited. In R. Manner and B. Manderick, editors, *Proceedings of the* 2^{nd} *Conference on Parallel Problems Solving from Nature*, pages 27–36. Springer Verlag, 1992.

[Mah95] S. W. Mahfoud. A comparison of parallel and sequential niching techniques. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference on Genetic Algorithms*, pages 136–143. Morgan Kaufmann, 1995.

[MC92] M. Manela and J. A. Campbell. Harmonic analysis, epistasis and genetic algorithms. In R. Manner and B. Manderick, editors, *Proceedings of the* 2^{nd} *Conference on Parallel Problems Solving from Nature*, pages 57–64. North Holland, 1992.

[MDG+95] J. Maresky, Y. Davidor, D. Gitler, G.Aharoni, and A. Barak. Selectively destructive restart. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference on Genetic Algorithms*, pages 144–150. Morgan Kaufmann, 1995.

[MdWS91] B. Manderick, M. de Weger, and P. Spiessens. The genetic algorithm and the structure of the fitness landscape. In R. K. Belew and L. B. Booker, editors, *Proceedings of the* 4^{th} *International Conference on Genetic Algorithms*, pages 143–150. Morgan Kaufmann, 1991.

[Mei] C. Meise. On the convergence of parallel simulated annealing. Technical report, Bielefeld University.

[MF97] P. Merz and B. Freisleben. Genetic local search for the tsp: New results. In T. Bäck, Z. Michalewicz, and X. Yao, editors, *Proceedings of the Fourth IEEE International Conference on Evolutionary Computation*, pages 159–164. IEEE Press, 1997.

[MF98] P. Merz and B. Freisleben. Memetic algorithms and the fitness landscape of the graph bipartitioning problem. In A.E. Eiben, Th. Bäeck, M. Schoenauer, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*, volume 1498 of *LNCS*, pages 765–774. Springer Verlag, 1998.

[MFH93] M. Mitchell, S. Forrest, and J.H. Holland. The Royal Road for genetic algorithms : Fitness landscapes and GA performance. In F. J. Varela and P. Bourgine, editors, *Proceedings of the First European Conference on Artificial Life-93*, pages 245–254. MIT Press/Bradford Books, 1993.

[MH93] M. Mitchell and J.H. Holland. When will a genetic algorithm outperform hill-climbing ? In S. Forrest, editor, *Proceedings of the* 5^{th} *International Conference on Genetic Algorithms*, page 647, 1993.

[Mic96] Z. Michalewicz. *Genetic Algorithms+Data Structures=Evolution Programs.* Springer Verlag, New-York, 1996. 3rd edition.

[Mor] P.D̃el Moral. On the convergence and the applications of the generalized simulated annealing. Technical report, University of Paul Sabatier.

[MPV87] M. Mezard, G. Parisi, and M. A. Virasoro. *Spin glass theory and beyond.* World Scientific Press, Singapore, 1987.

[NV92] A.E. Nix and M.D. Vose. Modeling genetic algorithms with markov chains. *Annals of Mathematics and Artificial Intelligence*, 5(1):79–88, 1992.

[NV98] B. Naudts and A. Verschoren. SGA search dynamics on second order functions. In J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution'97*. Springer Verlag, 1998.

[Oba94] J. Obalek. *Rekombinationsoperatoren für Evolutionsstrategieren*. PhD thesis, Universität Dortmund, Fachbereich Informatik, 1994.

[PBS97] A. Prügel-Bennett and J. L. Shapiro. The dynamics of a genetic algorithm for simple Ising systems. *Physica D*, 104:75–114, 1997.

[PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry: an introduction*. Springer Verlag, 1985.

[QVSG98] R.J. Quick, V.J..Rayward-Smith, and G.D.Smith. Fitness distance correlation and ridge functions. In A.E. Eiben, Th. Bäeck, M. Schoenauer, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*, volume 1498 of *LNCS*, pages 77–86. Springer Verlag, 1998.

[Rad91a] N. J. Radcliffe. Equivalence class analysis of genetic algorithms. *Complex Systems*, 5:183–20, 1991.

[Rad91b] N. J. Radcliffe. Forma analysis and random respectful recombination. In R. K. Belew and L. B. Booker, editors, *Proceedings of the* 4^{th} *International Conference on Genetic Algorithms*, pages 222–229. Morgan Kaufmann, 1991.

[REA96] H. Rosé, W. Ebeling, and T. Asselmeyer. The density of states - a measure of the difficulty of optimization problems. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*, volume 1141 of *LNCS*, pages 208–217. Springer Verlag, 1996.

[Rec73] I. Rechenberg. *Evolutionstrategie: Optimierung Technisher Systeme nach Prinzipien des Biologischen Evolution*. Fromman-Holzboog Verlag, Stuttgart, 1973.

[Rev75] D. Revuz. Markov chains. North-Holland, Amsterdam, 1975.

[Ron97] S. Ronald. Distance functions for order-based encodings. In Proceedings of the Fourth IEEE International Conference on Evolutionary Computation, editor, *Proceedings of the Fourth IEEE International Conference on Evolutionary Computation*, pages 49–54. IEEE Press, 1997.

[RS95a] N. J. Radcliffe and P. D. Surry. Fitness variance of formae and performance prediction. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 51–72. Morgan Kaufmann, 1995.

[RS95b] N. J. Radcliffe and P. D. Surry. Fundamental limitations on search algorithms: Evolutionary computating in perspective. In J. van Leeuwen, editor, *Computer Science Today: Recent Trends and Developements*, number 1000 in LNCS, pages 275–291. Springer Verlag, 1995.

[Rud94a] G. Rudolph. Convergence analysis of canonical genetic algorithm. *IEEE Transactions on Neural Networks*, 5(1):96–101, 1994.

[Rud94b] G. Rudolph. Convergence of non-elitist strategies. In Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel, and H. Kitano, editors, *Proceedings of the First IEEE International Conference on Evolutionary Computation*, pages 63–66. IEEE Press, 1994.

[Rud96] G. Rudolph. How mutation and selection solve long path problems in polynomial expected time. *Evolutionary Computation*, 4(2):195–205, Summer 1996.

[Rud97a] G. Rudolph. Asymptotical convergence rates of simple evolutionary algorithms under factorizing mutation distributions. In J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution'97*, pages 223–233. Springer Verlag, 1997.

[Rud97b] G. Rudolph. *Convergence Properties of Evolutionary Algorithms*. Kovac, Hamburg, 1997.

[RVSK97] S. Rochet, G. Venturini, M. Slimane, and E.M El Kharoubi. A critical and empirical study of epistasis measures for predicting ga performances: a summary. In J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution'97*, LNCS, pages 287–299. Springer Verlag, 1997.

[RW95a] Colin Reeves and Christine Wright. Epistasis in genetic algorithms: an experimental design perspective. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference on Genetic Algorithms*, pages 217–230. Morgan Kaufmann, 1995.

[RW95b] Colin Reeves and Christine Wright. An experimental design perspective on genetic alghorithms. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 7–22. Morgan Kaufmann, 1995.

[RY96] E. S. Ristad and P. N. Yanilos. Learning string edit distance. Technical Report CS-TR-532-96, Princeton University, 1996.

[Sal96] R. Salomon. Some comments on evolutionary algorithm theory. *Evolutionary Computation*, 1996.

[SCED89] J. D. Schaffer, R. A. Caruana, L. Eshelman, and R. Das. A study of control parameters affecting on-line performance of genetic algorithms for function optimization. In J. D. Schaffer, editor, *Proceedings of the* 3^{rd} *International Conference on Genetic Algorithms*, pages 51–60. Morgan Kaufmann, 1989.

[Sch81] H.-P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley & Sons, New-York, 1981. 1995 – 2^{nd} edition.

[Sch95] M. Schoenauer. Representations for evolutionary optimization and identification in structural mechanics. In J. Périaux and G. Winter, editors, *Genetic Algorithms in Engineering and Computer Sciences*, pages 443–464. John Wiley, 1995.

[Sch96] M. Schoenauer. Shape representations and evolution schemes. In L. J. Fogel, P. J. Angeline, and T. Bäck, editors, *Proceedings of the* 5^{th} *Annual Conference on Evolutionary Programming*, pages 121,129. MIT Press, 1996.

[SEB98] M. Schoenauer, A. Ehinger, and B. Braunschweig. Non-parametric identification of geological models. In D.B. Fogel, editor, *Proceedings of the Fifth IEEE International Conference on Evolutionary Computation*. IEEE Press, 1998.

[SG90] A.C. Schultz and J.J. Grefenstette. Improving tactical plans with genetic algorithms. In *Proceedings of IEEE Conference on Tools for AI*, pages 328–334. Morgan Kaufmann, 1990.

[SJK97] M. Schoenauer, F. Jouve, and L. Kallel. Identification of mechanical inclusions. In D. Dasgupta and Z. Michalewicz, editors, *Evolutionary Computation in Engeneering*, pages 477–494. Springer Verlag, 1997.

[SKJ96] M. Schoenauer, L. Kallel, and F. Jouve. Mechanical inclusions identification by evolutionary computation. *European Journal of Finite Elements*, 5(5-6):619–648, 1996.

[SPB96] J. Shapiro and A. Prugel-Bennet. Genetic algorithm dynamics in a two well potential. In R. K. Belew and M. D. Vose, editors, *Foundations of Genetic Algorithms 4*, pages 101–115. Morgan Kaufmann, 1996.

[Spe95] W. M. Spears. Adapting crossover in evolutionary algorithms. In J. R. McDonnell, R. G. Reynolds, and D. B. Fogel, editors, *Proceedings of the* 4^{th} *Annual Conference on Evolutionary Programming*, pages 367–384. MIT Press, March 1995.

[SR96] P.D. Surry and N.J. Radcliffe. Formal algorithms + formal representations = search strategies. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the* 4^{th} *Conference on Parallel Problems Solving from Nature*, number 1141 in LNCS, pages 366–375. Springer Verlag, 1996.

[SRV96] M. Slimane S. Rochet and G. Venturini. Epistasis for real encoding in genetic algorithms. In V.L. Narasimhan and L.C. Jain, editors, *Proceedings of the IEEE ANZIIS 96*, 1996.

[SRV97] M. Slimane S. Rochet and G. Venturini. A critical study of epistasis measures. In *in preparation*, 1997.

[SS98] F. Sourd and M. Schoenauer. Evolutionary mesh numbering: Preliminary results. In I. Parmee, editor, *Proceedings of ACDM'98*, LNCS. Springer Verlag, 1998. To appear.

[Str84] D. W. Stroock. *An introduction to the theory of large deviations.* Springer, New York, 1984.

[Suz93] J. Suzuji. A markov chain analysis on a genetic algorithm. In S. Forrest, editor, *Proceedings of the 5^{th} International Conference on Genetic Algorithms.* Morgan Kaufmann, 1993.

[SX93] M. Schoenauer and S. Xanthakis. Constrained GA optimization. In S. Forrest, editor, *Proceedings of the 5^{th} International Conference on Genetic Algorithms*, pages 573–580. Morgan Kaufmann, 1993.

[Tan89] Tanese. *Phd thesis.* PhD thesis, US, 1989.

[TG93] D. Thierens and D.E. Goldberg. Mixing in genetic algorithms. In S. Forrest, editor, *Proceedings of the 5^{th} International Conference on Genetic Algorithms*, pages 38–55. Morgan Kaufmann, 1993.

[VL91] M.D. Vose and G. E. Liepins. Punctuated equilibria in genetic search. *Complex systems*, 5(1):31–44, 1991.

[VW95] Michael D. Vose and Alden H. Wright. Stability of vertex fixed points and applications. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 103–11. Morgan Kaufmann, 1995.

[Wei90] E.D. Weinberger. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics*, 63:325–336, 1990.

[Whi91] D. Whitley. Fundamental principles of deception in genetic search. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms.* Morgan Kaufmann, 1991.

[Wil91] S.W. Wilson. Ga-easy does not imply steepest-ascent optimizable. In R. K. Belew and L. B. Booker, editors, *Proceedings of the 4^{th} International Conference on Genetic Algorithms.* mk, 1991.

[WM95] D.H. Wolpert and W.G. Macready. No free lunch theorems for search. Technical report, Santa Fe Institute, 1995.

[XMZT97] J. Xiao, Z. Michalewicz, L. Zhang, and K. Trojanowski. Adaptive evolutionary planner/navigator for mobile robots. *IEEE Transactions on Evolutionary Computation*, 1(1):18–28, 1997.

[Yao93] X. Yao. A review of evolutionary artificial neural networks. *Int. Jour. of Intelligent Systems*, 8:539–567, 1993.