Algorithmes Evolutionnaires pour l'Optimisation topologique de Formes

Hatem Hamda

Table des matières

Chapitre 0

Introduction Générale

Déterminer la forme appropriée d'une structure est un problème de première importance pour l'ingénieur. Dans tous les domaines de la mécanique des structures, l'impact de la bonne conception d'une pièce est très important sur sa résistance, sa durée de vie et son utilisation en service. On trouve de nombreux exemples d'optimisation de formes, notamment dans le milieu industriel. Les objets à optimiser peuvent être par exemple une aile d'avion, une carrosserie de voiture, un pont, ... L'objectif est dans ce cas double ; il s'agit d'améliorer le comportement de la structure tout en réduisant son coût.

Le problème type de l'optimisation de formes en mécanique des structures est de trouver la forme optimale d'une structure, qui soit à la fois de poids minimal et de rigidité maximale lorsque celle-ci est soumise à des sollicitations données.
Dans les bureaux d'études, on abordait ce problème en procédant par essais successifs, en testant des prototypes dont le **design** relevait du savoir faire et de l'expérience de l'ingénieur. Bien que des gains considérables en performance aient été acquis par cette approche, mais elle s'avère longue et très coûteuse : Il faut d'abord concevoir l'objet, construire un prototype, le tester, le modifier, en construire un nouveau, ...
Les techniques de conception automatique de formes sont donc nécessaires pour accélérer et réduire le coût de ces phases de conception. Des logiciels de modélisation numérique et d'optimisation, qui permettent d'analyser de nombreuses possibilités sans avoir à fabriquer de prototypes et qui automatisent la recherche de la forme optimale, ont été développés.

Parmi les problèmes d'optimisation de formes on peut distinguer trois grandes familles de difficulté et de généralité croissante. La première est **l'optimisation de dimensionnement**) : les formes sont paramétrées par un nombre fini de variables (par exemple, une

épaisseur, un diamètre, des dimensions). La deuxième est **l'optimisation de formes géométrique**[1], elle présuppose que la forme de la structure finale est compatible avec une topologie fixée au départ et interdit donc des modifications de la configuration de la structure comme les changements de sa nature où même de la connectivité de ses constituants. La troisième catégorie est **l'optimisation topologique de formes**, elle permet de modifier plus fondamentalement la nature de la structure, et ainsi optimiser automatiquement une structure sans aucune restriction explicite ou implicite sur sa topologie. Ce dernier type d'optimisation est donc le plus général mais aussi le plus difficile.

Nous nous intéressons dans ce travail au problème d'optimisation sous contraintes dans le contexte de l'optimisation topologique de formes en mécanique des solides. Le problème consiste à trouver la forme optimale d'une structure contenue dans un domaine fixé (i.e une distribution optimale de matière dans ce domaine) de telle sorte que le comportement mécanique de cette structure respecte certaines contraintes - ici par exemple que le déplacement maximal de la structure soumise à une ou plusieurs sollicitations données ne dépasse pas une valeur limite (déterminée par le cahier des charges), mais on pourrait aussi imaginer une borne sur des fréquences propres ou une combinaison de critères mettant en jeu la rigidité et le comportement vibratoire. Le critère à optimiser est ici le poids de la structure, mais on pourrait aussi optimiser d'autres critères : minimisation d'un état de tension, d'un déplacement ; maximisation d'une fréquence fondamentale ou d'une charge critique ; conception pour un coût imposé. Pour aborder ce problème il existe deux familles de méthodes : les approches déterministes, dont la plus connue est la méthode d'homogénéisation [18], et les méthodes fondées sur les techniques d'optimisation stochastique [99].

Toutefois, les approches déterministes sont principalement restreintes à l'élasticité linéarisée ; elles ne permettent de trouver qu'une seule solution (optimum local) ; et elles sont incapables de traiter les problèmes d'optimisation topologiques **multi-critères** (par exemple optimiser à la fois la rigidité et les fréquences propres d'une structure) autrement qu'en se ramenant à un seul objectif par agrégation des différents critères. Une approche possible pour dépasser ces limites est offerte par les méthodes d'optimisation stochastique, comme les **algorithmes évolutionnaires** (**AE**). L'optimisation topologique de formes est un domaine ou les AE ont été la source d'un grand pas en avant, permettant de résoudre des problèmes sur lesquels les méthodes traditionnelles n'ont pas de prises. Ainsi, des résultats originaux en optimisation topologique de formes ont été obtenus dans [99, 100] sur l'optimisation de structures en élasticité non-linéaire, ou pour des cas de chargements multiples.

1. encore appelée analyse de sensibilité ou méthode de variation de domaine

Les AE (dont les plus connus sont les **algorithmes génétiques**) sont des méthodes stochastiques d'optimisation globale qui utilisent une métaphore – grossière – de l'évolution darwinienne des espèces (*les individus les plus adaptés se reproduisent et survivent*) pour rechercher des optima d'une fonction à l'aide de transformations aléatoires. Introduites par Fogel en 1965 [57], Rechenberg en 1973 [139], Holland en 1975 [88], et popularisé par Goldberg en 1989 [65], ces méthodes se distinguent des autres méthodes globales de par la manipulation d'une population de solutions et non d'un seul point de l'espace de recherche. Elles mettent alors en jeu des opérateurs d'évolution agissant de manière stochastique sur une partie de la population en imitant les mécanismes avancés de l'évolution.

Très souvent, l'espace de **représentation** sur lequel on fait effectivement la recherche (sur lequel les opérateurs d'évolution opèrent), appelé également l'espace des **génotypes**, est différent de l'espace dans lequel la performance est calculée, appelé l'espace des **phénotypes**. Par exemple, en optimisation de structures, si on cherche une forme optimale définie par sa frontière représentée par des splines s'appuyant des points de contrôle en nombre fixe, les génotypes sont des vecteurs réels de taille (finie) donnée. En revanche, l'espace des phénotypes est l'ensemble des formes dont le comportement mécanique sert à calculer la fonction-objectif.

Dans le contexte de l'optimisation paramétrique, i.e. lorsque l'espace de recherche est de dimension finie, les AE sont simplement une technique de plus, à la fois une puissante méthode d'ordre 0 (seules les valeurs de la fonction-objectif sont nécessaires), et une méthode globale stochastique capable de s'échapper de minima locaux pour trouver un optimum global, ou même des optima globaux multiples dans le cas de fonctions multimodales. Les AE ont été employés dans de nombreux domaines : pour des problèmes de combinatoire discrète (depuis la TSP standard [115] ou les problèmes de coloration de graphes [49] jusqu'à des problèmes réels d'attribution de fréquences [48] ou de l'emploi du temps [128]), aussi bien que des problèmes continus (depuis la célèbre expérience d'optimisation de la forme d'une tuyère [139, 158] jusqu'à de plus récentes applications industrielles [65, 176, 126]).
Toutefois, une particularité des AE est son aptitude à traiter des génotypes très inhabituels comme les espaces de graphes, de listes non ordonnées... La seule contrainte est de fournir une procédure d'initialisation et des opérateurs d'évolution qui respectent quelques propriétés [165]. En effet, plus l'espace de recherche est grand, meilleure seront les solutions mais elles seront d'autant plus difficiles à atteindre. Pourtant, les succès les plus spectaculaires des AE ont été obtenus en utilisant des représentations non structurées, c'est-à-dire des représentations non paramétriques. Reprenons l'exemple des splines et considérons maintenant un nombre de nœuds variable (avec des positions variables

également) : cela donne une représentation non structurée et il n'est pas très difficile d'imaginer des opérateurs d'évolution adaptés permettant de faire évoluer de telles représentations. Par exemple, la fameuse expérience pionnière de la tuyère par Rechenberg et Schwefel utilisait une représentation de longueur variable [139, 158].

Mais il y a une étape supplémentaire qui nous éloigne des représentations directes : dans les approches appelées *morphogénétiques*, au lieu de chercher une solution du problème de départ, l'idée consiste à chercher un "programme" qui, en s'exécutant, va construire une solution. La performance du programme étant celle de la solution qu'il permet de construire. Toute la **programmation génétique** (Genetic Programming) [103] appartient à cette catégorie de représentations, et des résultats impressionnants ont été obtenus en utilisant ces techniques, par exemple dans le domaine de la conception de circuits électroniques analogiques [105].

Dans le cadre de l'optimisation topologique de formes, cette thèse s'intéresse dans un premier temps aux problèmes de représentation pour les structures. L'approche la plus "naturelle", utilisée dans tous les travaux antérieurs [27, 99], est la représentation paramétrique directe basée sur un maillage donné du domaine de travail. Cette représentation par tableau de bits, bien que très employée et utile pour une première comparaison avec les approches déterministes [99, 100] , montre ses limites lorsque l'on veut augmenter la complexité des maillages. En effet, la taille d'un individu (le nombres de bits nécessaires pour décrire un individu) est égale à la taille du maillage. Malheureusement, les résultats théoriques [32] comme les constatations empiriques [66] indiquent que la taille critique de population nécessaire pour atteindre la convergence augmente au moins linéairement avec la taille de chaque individu. De plus, les populations plus nombreuses nécessitent souvent un plus grand nombre de générations pour converger. Il est donc clair que cette approche doit restreindre son domaine d'application à de maillages grossiers bidimensionnels, alors que les ingénieurs ont besoin de fins maillages tridimensionnels!

Ces considérations conduisent à la recherche de représentations plus compactes, dont la complexité (le nombre de variable de design) ne dépend pas de celle de la discrétisation. Pour obtenir une représentation indépendante de la complexité une ultime étape consiste à la faire évoluer elle-même en l'ajustant par AE.

Pour remédier à cet inconvénient, nous avons exploré de nombreuses représentations originales pour les structures, basées sur la théorie des **diagrammes de Voronoï** [25, 148], dont la complexité est indépendante de celle de tout maillage, et auto-adaptative (i.e. la complexité des solutions est ajustée par l'algorithme lui-même), mais qui nécessitent tout

de même des “gènes” élémentaire définis par le programmeur. Dans ces nouveaux espaces de représentations, un individu est une liste non ordonnée et de longueur variable: ces représentations sont non structurées.

Par ailleurs, pour prendre en compte les contraintes dans les algorithmes évolutionnaires, plusieurs méthodes ont été mises au point [86][77]. Toutefois, la méthode la plus générale reste la méthode de pénalisation et c'est aussi la plus simple à mettre en œuvre: elle ne demande que la modification de la fonction objectif en introduisant des termes de pénalisation. Cependant, cette approche nécessite de l'utilisateur un choix délicat de ces coefficients sensibles, qui influent grandement sur la qualité des résultats obtenus. D'où l'intérêt d'essayer d'appliquer une technique **adaptative** qui ajuste automatiquement les valeurs des paramètres en fonction de l'histoire de l'évolution.

Nous proposons ici deux nouvelles méthodes de pénalisation adaptative, en se basant sur le principe qui consiste à effectuer la mise à jour des paramètres de pénalisation en utilisant les statistiques globales de faisabilité[2] dans la population. Le but de cette démarche est d'explorer les environs de la frontière de la région admissible en essayant de conserver dans la population les individus qui se situent "de part et d'autre" de cette frontière. En effet, dans de nombreux problèmes d'optimisation, on sait que la solution se situe sur la frontière de la région admissible. Des techniques spécifiques de traitement des contraintes ont été proposées pour explorer uniquement cette frontière [152, 151]. Toutefois, pour l'optimisation topologique de formes, avec la rigidité comme fonction-objectif, bien qu'il n'a pas été établi si la solution est sur la frontière du domaine admissible ou pas pour le problème continu, le bon sens suggère qu'il se trouve "proche" de cette frontière. De plus, cette frontière est difficilement caractérisable pour ce problème, mais on peut explorer la zone admissible voisine de la frontière grâce à la méthode de pénalisation adaptative.

Outre les représentations à base de diagrammes de Voronoï, nous proposons une autre représentation – une approche morphogénétique basée sur la théorie des fractales – des **IFS** (Iterated Function Systems)[39], qui est une tentative pour élargir l'espace de recherche de telle sorte qu'aucune hypothèse a priori sur la forme des blocs constitutifs d'une solution n'est plus nécessaire: Chaque structure est définie par l'attracteur d'un ensemble de fonctions contractantes définies sur le domaine de travail. La propriété, d'un tel objet mathématique, qui motive leur utilisation pour représenter des formes est leur capacité à décrire des formes complexes avec un petit nombre de paramètres. Cette propriété est à l'origine du succès de la théorie des IFS dans le domaine de la compression du signal, mais elle a de nombreuses autres applications en analyse d'image ou de signal [171].

2. Un individu respectant l'ensemble des contraintes est appelé faisable ou admissible

Nous nous intéressons aux **IFS mixtes** [110], et la représentation est basée sur les arbres de la programmation génétique [104].

Des résultats originaux utilisant la représentation Voronoï confirment la puissance de l'approche non structurée. Des comparaisons sur le benchmark du cantilever permettent de choisir parmi les différentes représentations à base de diagrammes de Voronoï et de cerner les limites de l'approche morphogénétique, au moins pour des problèmes simples d'optimisation topologique de structures.

Mais une autre question importante est celle de la modularité: par exemple, lors de l'optimisation de structures élancées, les solutions optimales retenues par les ingénieurs jusqu'alors sont constituées de la juxtaposition de modules élémentaires. Or, si l'approche évolutionnaire à l'aide de représentations de type Vorornoï permet de trouver des solutions satisfaisantes, l'algorithme doit "redécouvrir" autant de fois qu'il est nécessaire de tels modules. Avant d'aller vers des représentations totalement modulaires, nous esquissons ce que pourrait être une représentation modulaire permettant la réutilisation d'une partie de la représentation.

Dans la dernière partie de cette thèse nous nous intéressons au problème d'optimisation topologique multi-critères (ou multi-objectifs). Les problèmes d'optimisation sous contraintes sont en fait des problèmes multi-objectifs – il existe des similarités certaines entre contraintes et multi-objectifs: les contraintes inégalité peuvent être considérées comme autant d'objectifs, avec la différence que seuls les solutions en deçà d'une certaine valeur sont considérées comme valides.

L'optimisation multi-objectifs cherche donc à optimiser plusieurs fonctions objectifs (souvent contradictoires) simultanément. Contrairement à l'optimisation mono-objectif, la solution d'un problème multi-objectifs n'est pas une solution unique, mais un ensemble de solution, connu comme l'ensemble des solutions Pareto optimales.
Ainsi, le problème considéré dans cette thèse, est en fait un problème multi-objectifs: on cherche à minimiser **à la fois** le poids et le déplacement maximal sous l'action du chargement. Il est bien connu que ces objectifs sont contradictoires (c'est-à-dire la diminution du poids entraîne généralement la diminution de la rigidité et vice-versa). Le développement des nouvelles méthodes d'optimisation multi-objectifs [43] permet de considérer les deux objectifs simultanément et d'essayer d'identifier l'ensemble des meilleurs compromis possibles entre les objectifs, aussi appelé le **front de Pareto**. Ce qui permet au décideur de choisir la solution à partir d'une information complète.

Nous allons maintenant donner le plan de la thèse qui comporte six chapitres.

Le premier chapitre propose une introduction au domaine de l'évolution artificielle avec une synthèse des principales approches. Après une illustration des origines et du principe des AE, nous présentons les notions de darwinisme artificiel, les techniques d'amélioration. Nous exposons notamment différents moyens de représenter un individu, et différents opérateurs de variation. Ce chapitre présente aussi les quatres grandes familles historiques d'AE et les principaux résultats théoriques des algorithmes génétiques.

Le deuxième chapitre est consacré à la présentation des différents types d'optimisation de formes de structures en mécanique de solide, accompagnée d'une discussion de leurs avantages et leurs limites.

Le troisième chapitre permet d'introduire le problème mécanique considéré ainsi qu'un rappel du cadre théorique défini par Ghaddar et al. [62] pour l'optimisation topologique de formes. Nous présentons notamment la fonction performance et les différentes techniques de pénalisation pour les AE. Nous introduisons ensuite les deux nouvelles approches adaptatives proposées ici. Une étude expérimentale comparative montre, pour le problème d'optimisation de formes, la supériorité de la stratégie adaptative par rapport aux autres méthodes.

Dans le quatrième chapitre nous introduisons trois représentations basées sur les diagrammes de Voronoï, ainsi que les opérateurs de variations associés. Nous présentons les résultats numériques obtenus, avec une représentation de Voronoï simple, sur le problème de l'optimisation du moment d'inertie pour une première validation de l'approche, et sur plusieurs problèmes de l'optimisation topologique de formes comprenant en particulier des résultats originaux sur un problème tridimensionnel. Par ailleurs, nous discutons les avantages de ces représentations par rapport à la représentation standard par tableaux de bits. Une étude expérimentale comparative est effectuée sur des problèmes test simples dans le but de comparer les mérites des trois représentations. La dernière section de ce chapitre est consacré à la représentation modulaire permettant la réutilisation d'une partie de la représentation.

Le cinquième chapitre est consacré à l'étude de l'emploi de la représentation à base d'IFS mixtes. Nous introduisons brièvement les bases de la théorie des IFS et les algorithmes numériques les plus connus, pour le calcul de l'attracteur. Nous présentons ensuite les différents types d'IFS – en particulier les IFS mixtes – ainsi que les résultats préliminaires montrant les avantages potentiels de cette approche.

Finalement, dans le sixième chapitre, nous rappelons les différentes approches multi-objectifs utilisées dans la littérature, accompagnées d'une discussion de leurs avantages et leurs limites – en particulier, l'approche **NSGA-II** que nous avons choisi d'appliquer dans la suite au problème d'optimisation topologique de formes avec deux objectifs (poids et rigidité). Enfin, nous montrons les résultats obtenus sur deux problèmes test de la plaque console.

La dernière partie tire les conclusions de ce travail, et ouvre des voies pour des recherches ultérieures.

Chapitre 1

Évolution artificielle

Sommaire

1.1 Introduction

Les algorithmes évolutionnaires (AE) [16] sont des méthodes d'optimisation stochastique basées sur une imitation grossière de l'évolution naturelle des populations. Méthodes globales d'ordre 0, leur robustesse et leur souplesse leur permettent d'attaquer la résolution numérique de problèmes difficiles à résoudre autrement. Mais c'est leur capacité à travailler sur des espaces de recherche non standards qui leur ouvre les perspectives les plus originales. Bien que simplistes du point de vue d'un biologiste, ces algorithmes sont suffisamment complexes pour fournir des mécanismes de recherche adaptatifs et robustes.

Les algorithmes d'évolution ont un mécanisme de base commun : celui-ci consiste à faire évoluer une population par transformation aléatoire de certains de ses éléments, et application du principe de la sélection naturelle. Plusieurs techniques ont été élaborées dont les principales sont les suivantes :

- Les Algorithmes Génétiques (*Genetic Algorithms* : GA), ce sont probablement les algorithmes les plus connus et utilisés dans le calcul évolutionnaire. Ils ont été développés dans les années 60 par Holland [87] [88], pour étudier le processus complexe d'adaptation des espèces naturelles. Plus précisément, pour l'optimisation paramétrique, par De Jong en 1975 [46] et Goldberg en 1989 [65].
- Les Stratégies d'évolution (*Evolution Strategies* : ES) ont été développés en Allemagne (Rechenberg 1973 [139], Schwefel 1981 [158]), pour résoudre des problèmes numériques d'optimisation dans l'espace des paramètres réels.
- La Programmation évolutionnaire (*Evolutionnary Programming* : EP), est apparue aux Etat-Unis (L.J Fogel 1966 [57]) dans l'espace des automates à états finis pour le prédiction de séries temporelles.
- La Programmation Génétique (*Genetic Programming* : GP) (Koza 1992 [103]) consiste à faire évoluer des structures d'arbres représentant des programmes.

Bien que les applications des algorithmes évolutionnaires soient diverses et variées (c'est

ainsi qu'ils ont donné de bons résultats dans différents domaines : optimisation de forme en mécanique des solides et en aérodynamique, recherche opérationnelle et génie industriel, automatique, ...), l'étude mathématique de ces algorithmes reste encore bien limitée face à leur complexité théorique et il a fallu attendre les années 90 pour que des démonstrations complètes et rigoureuses de convergence en probabilité soient établies (Cerf [31, 32], Rudolph [141, 142]). Néanmoins, ces résultats théoriques sont difficilement exploitables dans la pratique.

1.2 Principe d'un algorithme d'évolution

Cette section présente les idées de base des algorithmes évolutionnaires. Pour optimiser une fonction-objectif donnée $\mathcal{F}$ (appelée aussi *performance* ou *fitness*) définie sur un espace de recherche E, une *population* d'*individus* (points de E) est soumise à une suite de *générations* (la population initiale est tirée au hasard dans E). Une génération commence par la *sélection* de quelques individus bien *adaptés* (par rapport à $\mathcal{F}$) pour la reproduction. Ces individus engendrent une *descendance* en utilisant des opérateurs stochastiques appelés *croisement* pour les opérateurs binaires, et *mutations* pour les opérateurs unaires (agissant sur un seul individu). Enfin, quelques-uns des descendants *remplacent* certains des parents pour terminer le processus de génération. Les paradigmes de sélection et de remplacement, qui représentent les étapes de la règle Darwinienne de la *survie du mieux adapté*, peuvent être stochastiques ou déterministes. Comme dans l'évolution naturelle, on espère l'émergence progressive d'individus de mieux en mieux adaptés : les meilleurs individus de la population finale (au regard de $\mathcal{F}$) devraient être proches de solutions du problème d'optimisation posé.

L'espace de *représentation* sur lequel on fait effectivement la recherche (sur lequel les opérateurs d'évolution opèrent), appelé également l'espace des *génotypes*, est souvent différent de l'espace dans lequel la performance est calculée, appelé l'espace des *phénotypes*. Pour passer de l'espace des *phénotypes* à l'espace des *génotypes*, une étape de modélisation supplémentaire, ou *codage*, est nécessaire. Par exemple, en optimisation de structures, si on cherche une forme optimale définie par sa frontière représentée par des splines s'appuyant sur des points de contrôle en nombre fixe, les génotypes sont des vecteurs réels de taille (finie) donnée. En revanche, l'espace des phénotypes est l'ensemble des formes dont le comportement mécanique sert à calculer la fonction-objectif.

L'algorithme se déroule comme suit (cf. la figure 1.1),

1. Une population de p individus est initialisée aléatoirement.
2. La fitness de chaque individu de la population est évaluée.

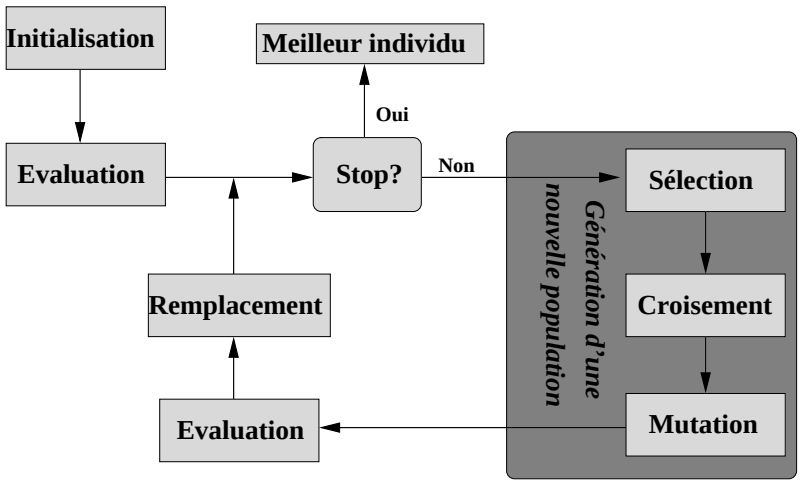


FIG. 1.1 – *Principe général de fonctionnement d'un algorithme d'évolution*

3. Le processus suivant est itéré jusqu'à satisfaction d'un critère d'arrêt.
   - Sélectionner les individus les plus performants (au sens de $\mathcal{F}$) de la population, les recopier pour former une nouvelle population de même taille. Les modes de sélection peuvent être déterministes ou stochastiques.
   - Créer une nouvelle population en appliquant les opérateurs de variation :

     Opérateur de croisement : recombiner des parties de deux individus pour en obtenir deux nouveaux. Cet opérateur est appliqué avec une probabilité donnée p_c.

     Opérateur de mutation : modifier aléatoirement un individu avec une probabilité p_m. La mutation introduit la *diversité* dans la population.
   - Évaluer la fitness de chaque individu de la nouvelle population.
   - Remplacer certains individus de l'ancienne population par les meilleurs individus de la nouvelle population.

Le critère d'arrêt peut prendre plusieurs formes : par exemple nombre maximum d'itérations ou d'évaluations, stagnation de la valeur de la fitness du meilleur individu.

Sous la pression du milieu, les individus se reproduisent, se croisent (échangent des informations) et mutent (*explorent* de nouvelles régions de l'espace de recherche). Ainsi, au bout d'un certain nombre de générations, on espère que les individus les plus performants vont apparaître dans la population (les *optima* de $\mathcal{F}$).

Si les individus d'une population se ressemblent trop, les populations suivantes risquent de devenir de plus en plus homogènes. Dans ce cas, l'évolution d'une population risque de se résumer à l'évolution d'un seul individu *dominant*, réduisant ainsi l'*exploration* de l'espace de recherche (convergence *prématurée*). Pour effectuer une recherche efficace, il faut donc maintenir un équilibre entre l'*exploitation* des bonnes solutions rencontrées et l'exploration de zones inconnues de E. Un excès d'exploitation peut conduire à une convergence prématurée (enlisement dans un optimum local) tout comme un excès d'exploration pourrait conduire à une quasi recherche aléatoire (pas de convergence).

Nous allons dans la suite de ce chapitre détailler les divers étapes d'un algorithme d'évolution.

1.3 Initialisation de la population

Le choix d'une population de N individus $P(0) = \{X_1, \dots X_N\}$ se fait, en général, par des tirages uniformes dans l'espace de recherche E en veillant éventuellement à ce que les individus produits respectent les contraintes. Par ailleurs, si on dispose d'informations a priori sur le problème indiquant une région où l'on est sûr de trouver l'optimum, il paraît bien évidemment naturel d'ajouter manuellement de bonnes solutions dans la population initiale, tout en assurant une diversité suffisante de la population. La population initiale peut aussi être le résultat d'une évolution précédente.

1.4 Darwinisme artificiel / moteur d'évolution

La partie darwinienne de l'algorithme évolutionnaire comprend deux étapes : l'étape de reproduction afin de sélectionner les parents qui vont se reproduire et l'étape de remplacement, qui remplace les individus non sélectionnés par ceux désignés à survivre (figure 1.1).

La sélection est un opérateur essentiel dont le principe consiste à permettre aux meilleurs individus d'une population de se reproduire. Le réglage de ce mécanisme est déterminant dans le comportement de l'algorithme d'évolution : un excès de sélection conduit à une perte de diversité (et résulte en des zones inatteignables de l'espace de recherche), et une insuffisance peut mener à une marche aléatoire (pas de convergence). On trouve dans la littérature un nombre important de stratégies de sélection plus ou moins adaptées aux problèmes qu'elles traitent. Une étude de plusieurs procédures de sélection a été réalisée [64, 24] et a permis de comparer différents principes possibles en fonction de plusieurs quantités que sont :

– **la pression de sélection** : l'espérance du nombre de copies du meilleur individu;

- **la variance de la sélection** : variance normalisée de la distribution des fitness de la population après application de la sélection ;
- **la perte de diversité** : le nombre d'individus non sélectionnés.

Nous présentons ici les procédures de sélection plus fréquemment rencontrées.

1.4.1 Sélection proportionnelle

La sélection par roulette

Introduite par Holland [88], elle consiste à représenter sur une roulette chacun des individus de la population $P(t) = \{X_1, \ldots, X_N\}$ par une section (cf. figure 1.2) qui est proportionnelle à leur fitness. Ensuite, on lance N fois la roulette et on sélectionne chacun des gagnants. Autrement dit la probabilité de sélectionner l'individu X_i de la population est $\frac{\mathcal{F}(X_i)}{\sum_{j\in\{1\ldots N\}}\mathcal{F}(X_j)}$.
L'espérance n_i de nombre de copies d'un élément X_i de la population courante est donné par l'expression :

$$n_i = \frac{N.\mathcal{F}(X_i)}{\sum_{j\in\{1\ldots N\}}\mathcal{F}(X_j)}$$

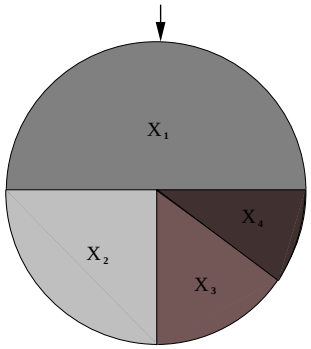


FIG. 1.2 – *Roue de loterie pour une population de 4 individus avec* $\mathcal{F}(X_i) = \{50{,}25{,}15{,}10\}$. *Pour tirer un parent il suffit de "faire tourner" la roue ; si elle s'arrête sur la case i,* X_i *est sélectionné.*

Cette méthode de sélection favorise les meilleurs individus, mais les mauvais ont tout de même des chances d'être sélectionnés. Par contre, le coût d'exécution et la variance sont élevés. La perte de diversité est aussi possible car le nombre de copies obtenues des meilleurs individus (voir uniquement du meilleur) peut représenter l'ensemble de la prochaine population.

La sélection avec reste stochastique

La sélection avec reste stochastique (Baker [12]) s'inspire fortement de la sélection par roulette, sauf qu'elle ajoute un aspect déterministe. Dans cette approche le nombre effectif de copies de l'individu X_i sera au moins égal à la partie entière de son espérance n_i :

$$E(N\frac{\mathcal{F}(X_i)}{\sum_{j\in\{1...N\}}\mathcal{F}(X_j)})$$

Ainsi, un individu avec une fonction d'adaptation (fitness) supérieure ou égale à la moyenne des fonctions d'adaptation de la population aura un nombre de copies supérieur à 1. Le reste décimal est sélectionné par roulette (si le nombre d'individus nécessaires n'est pas atteint, la partie décimale est utilisée comme probabilité de sélection de l'individu et ce jusqu'à obtention du nombre d'individus requis). L'intérêt de cette sélection est qu'elle diminue la variance (pour une petite population).

Au cours de l'évolution d'une population, avec la sélection proportionnelle, les individus ayant les meilleurs performances sont reproduits plus souvent que les autres et remplacent généralement les moins bons. Si la pression de sélection est élevée, le risque de convergence prématurée est important. C'est le cas si un *super individu*, plus performant que les autres, devient majoritaire dans la population qu'il finit par envahir complètement. Dans l'exemple de la figure 1.3 le second mode M_2 risque d'être le seul représentant pour la génération suivante et seule la mutation pourra aider à atteindre l'objectif global M_1 au prix de nombreux essais successifs.

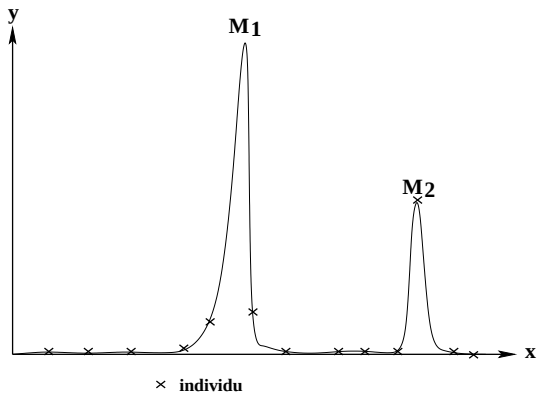


FIG. 1.3 – *Exemple où les sélections classiques risquent de ne reproduire qu'un individu.*

Pour éviter cette situation , il a été développé d'autres procédés, sans modifier le principe de sélection, qui empêche les meilleurs individus d'éliminer complètement les mauvais.

Mise à l'échelle (scaling)

La mise à l'échelle est la technique la plus répandue pour éviter une trop rapide convergence ou une uniformisation de la population. Le principe consiste à déformer artificiellement le critère original afin de réduire les écarts de fitness entre les individus au début du processus d'évolution (exploration plus large de l'espace de recherche) et de les accentuer à la fin (lorsque la population commence à converger et les fitness des individus deviennent presque toutes comparables). Pour cela la fitness réelle $\mathcal{F}_r$ est remplacée par une fitness mise à l'échelle $\mathcal{F}_s$, qui sera utilisée lors de la sélection.

Parmi les fonctions de mise à l'échelle on peut envisager

1. **la mise à l'échelle linéaire** : $\mathcal{F}_s = a(t).\mathcal{F}_r + b(t)$, où a et b sont deux fonctions de la génération courante t. En début de l'évolution, le coefficient a est inférieur à 1, ce qui permet de réduire les écarts de fitness et donc favoriser l'exploration de l'espace. En fin de l'évolution, a est supérieur à 1, ce qui permet d'accentuer les écarts de fitness et favoriser les meilleurs.
2. **la mise à l'échelle exponentielle** (cf. figure 1.4) : $\mathcal{F}_s = \mathcal{F}_r^{b(t)}$, où t désigne la génération courante.
   Suivant les valeurs de b on observe :
   - b proche de zéro : une réduction importante des écarts de fitness; aucun individu n'est vraiment favorisé.
   - b proche de 1 : une mise à l'échelle inopérante.
   - $b > 1$: des écarts accentués, seuls les bons individus sont sélectionnés ce qui produit l'émergence des optimum (locaux).

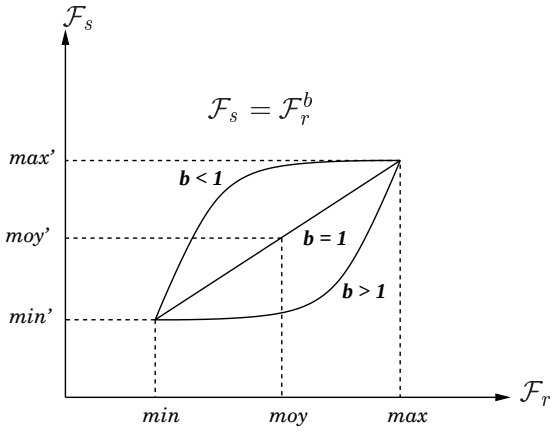


FIG. 1.4 – *Fonction de mise à l'échelle exponentielle.*

Une autre méthode également très utilisée est la sélection par le rang, méthode qui élimine le problème de l'influence d'un seul individu.

Sélection par Rang

La sélection par rang a été proposée par Baker [13, 69]. Le principe de cette sélection commence par un tri des individus par ordre de fitness, puis associe une probabilité de sélection p_i à chaque individu en fonction de son rang.

Parmi les distributions typiques, il y a :

1. **la distribution linéaire** [13] définie par

$$p_i = \frac{1}{N}(\eta_{min} + (\eta_{max} - \eta_{min})\frac{i-1}{N-1}) \ ,$$

   où $\frac{\eta_{max}}{N}$ est la probabilité de sélectionner le meilleur individu et $\frac{\eta_{min}}{N}$ est la probabilité de sélectionner le moins bon. La somme des probabilités doit être 1, et donc $\eta_{max} = 2 - \eta_{min}$, ce qui entraine $0 < \eta_{min} < \eta_{max} \leqslant 2$.
2. **la distribution exponentielle** : $p_i = a.e^{b.r(i)} + c$, où $r(i)$ est le rang de l'individu i.

Cette sélection ne génère pas de convergence prématurée, ni de stagnation en fin d'évolution. Mais elle requiert un tri des individus qui peut être coûteux en temps de calcul dans le cas de grandes populations.

1.4.2 Sélection par tournoi

La sélection par tournoi n'utilise aussi que des comparaisons entre les individus, et ne nécessite même pas de tri de la population. Elle possède un paramètre T, qui est la taille du tournoi. Pour sélectionner un individu, on en tire T uniformément dans la population, et on sélectionne d'une manière déterministe le meilleur de ces T individus. Au cours d'une génération il y a autant de tournois que d'individus à sélectionner. Cette méthode est caractérisée par une pression de sélection en général plus forte que les méthodes proportionnelles (pour qu'un individu peu performant puisse être sélectionné, il faut que ses adversaires soient encore moins bon que lui). De plus, elle est la moins chère en terme de coût d'exécution ; elle est peu sensible aux erreurs sur $\mathcal{F}$, facilement paramétrable par la valeur de T, et ne conduit pas à une convergence prématurée. Par contre, sa variance est élevée [45]. A noter que le tournoi de taille 2 et la sélection linéaire par le rang ($\eta_{max} = 2$) ont la même espérance.

1.4.3 Le remplacement

Diverses stratégies de remplacement peuvent être utilisées, le principe étant de remplacer l'ancienne population par une nouvelle, obtenue après application d'opérateurs de variation. Dans les algorithmes génétiques standards, le remplacement est *générationnel*, c'est-à-dire que la population des enfants remplace purement et simplement la population parente. Par contre, dans le Steady State GA - SSGA (cf. section 1.8.1), seul un pourcentage des meilleurs enfants remplace des individus de l'ancienne génération. Il existe dans SSGA d'autres stratégies de remplacement dont :

- le remplacement systématique du plus mauvais individu ;
- le remplacement aléatoire (en faisant attention à maintenir une stratégie de recherche cohérente).

Le but du SSGA est d'augmenter la vitesse de convergence de l'algorithme génétique simple, mais il peut cependant engendrer une convergence rapide vers des optima locaux.

Une autre technique de remplacement est le remplacement *déterministe*, utilisé dans les stratégies d'évolution (ES). Son caractère purement déterministe lui donne un rôle clef dans l'évolution vu qu'il guide la recherche vers les zones des meilleurs individus. Schwefel a introduit deux schémas distincts [157, 158] :

- Le schéma (μ,λ)-ES : on désigne par μ le nombre d'individus de la population qui génère $\lambda > \mu$ nouveaux individus (par mutation et croisement). Le remplacement s'opère en sélectionnant les μ meilleurs individus parmi l'ensemble des λ enfants.
- Le schéma $(\mu + \lambda)$-ES : avec ce schéma, la sélection cherche les μ meilleurs individus parmi l'union des μ parents et λ enfants.

Le remplacement $(\mu + \lambda)$ est élitiste et il garantit une amélioration monotone de la fitness du meilleur individu au cours des générations, mais il s'adapte mal à un éventuel changement d'environnement. Par contre, avec le remplacement (μ,λ), on peut perdre les meilleurs individus, mais l'algorithme est plus flexible en cas de changements d'environnement (optimisation dynamique).

Il est à noter que l'*élitisme* est souvent utilisé. Ce mécanisme permet de maintenir les meilleurs individus (souvent le meilleur uniquement) de la population à la génération t dans la prochaine population (génération $t + 1$) s'il n'y a pas d'enfants meilleurs que lui.

1.5 Le nichage

Pour certains problèmes présentant plusieurs optima quasi-équivalents, la technique de mise à l'échelle, en fin de convergence, va favoriser le mode dominant (super individu) et ne permet pas de donner les modes quasi-optimaux. Pour éviter le rassemblement des individus autour d'un mode dominant et assurer une certaine diversité dans la population, plusieurs techniques de nichage ont été proposées [144]. Ces techniques ont pour objectif la formation et le maintien de sous-populations (niches) dans la même population (figure 1.5) dont le but de favoriser la répartition des individus sur les différents optima de la fonction à optimiser $\mathcal{F}$.

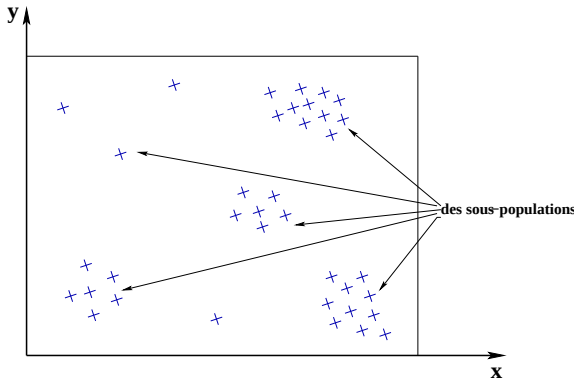


FIG. 1.5 – *Effet du nichage sur une population.*

1.5.1 Le partage (Sharing)

Le partage a été introduit par Goldberg et Richardson en 1987 [68]. Son but principal est de pénaliser les fitness en fonction du taux d'agrégation de la population dans le voisinage d'un individu: plus un individu a de semblables dans son voisinage, plus sa fitness sera pénalisée. Il faut donc fixer une distance σ_{share}, permettant de repérer les individus proches. La pénalisation des individus proches se fait en utilisant une fitness auxiliaire (comme pour la mise à l'échelle) qui sera utilisée par la sélection. Cette nouvelle fitness est calculée de la façon suivante:

$$\mathcal{F}_s(X_i) = \frac{\mathcal{F}_r(X_i)}{\sum_{j=1}^{N} S(d(X_i,X_j))}$$

avec $S(d(X_i,X_j))$ désigne la similarité (distance) entre deux individus. Elle est égale à 1 si les deux solutions sont identiques, 0 si la distance d entre eux dépasse le seuil σ_{share} et

une valeur intermédiaire pour les degrés de dissimilarité intermédiaires. La fonction S la plus utilisée est :

$$S(d(X_i,X_j)) = \begin{cases} 1 - (\frac{d}{\sigma_{share}})^{\alpha} & si\ d(X_i,X_j) < \sigma_{share} \\ 0 & sinon \end{cases}$$

La figure 1.6 donne l'allure de $S(d)$ pour différentes valeurs de α.

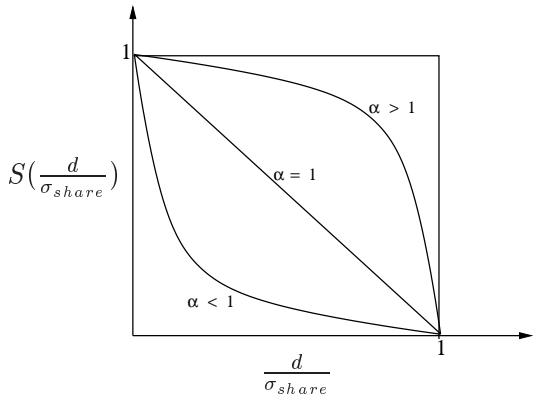


FIG. 1.6 – *Allure de* $S(\frac{d}{\sigma_{share}})$.

Le sharing classique nécessite une comparaison deux à deux des individus et induit une complexité en $O(N^2)$. Pour réduire cette complexité, une autre technique de sharing a été développée (cf. X.Yin et N.Germay [175]), basée sur un clustering préalable, dont la complexité est $O(N \log N)$. On trouvera plus de détails sur le sharing dans les références [65] [144].

1.5.2 L'éclaircissement (Clearing)

L'éclaircissement est une autre version de nichage développée par Petrowski en 1996 [133], où toutes les ressources d'une sous-population sont réservées au meilleur du groupe, au lieu d'être partagées par tous les membres (cf. [133] pour plus de détails). Cette méthode a l'avantage de contrôler la densité des points dans chaque région de l'espace de recherche. En plus, sa complexité est inférieure à celle du partage classique ($O(C.N)$, où C est le nombre de sous-populations).

1.5.3 Le surpeuplement (Crowding)

Initialement proposé par De Jong [46], le crowding est basé sur l'analogie naturelle avec la compétition entre les individus pour l'acquisition de ressources [111]. Ainsi des individus dissemblables occupant des domaines différents de l'espace d'état ne sont pas en position de rivalité pour accéder aux ressources (fitness). À l'inverse, des individus proches dans l'espace d'état visent à occuper le même sous-domaine et sont donc en compétition pour faire augmenter leur fitness. Dans ce cas les individus les plus forts vont donc chasser les plus faibles à condition qu'ils soient en compétition avec eux. En terme d'algorithme d'évolution, on simule ce principe de la façon suivante.

On commence par sélectionner deux parents dans la population (P_1 et P_2). On génère ensuite deux enfants (E_1 et E_2) par croisement et mutation. Ensuite, on les met en compétition (tournoi) avec les parents. Pour réaliser ce tournoi, on recherche d'abord les couples parents-enfants les plus proches puis on applique la loi du plus fort. Les individus ainsi sélectionnés sont ensuite insérés dans la nouvelle population correspondant à la génération suivante. Ainsi, lorsqu'un enfant est meilleur que les parents, il remplace le parent le plus proche afin d'assurer la descendance dans cette zone d'espace. Ce procédé est ensuite itéré $N/2$ fois afin de compléter la nouvelle génération.

1.6 Représentation d'un individu

1.6.1 Le principe

Le codage (ou représentation) d'un individu (cf. section 1.2) doit englober les caractéristiques fondamentales du problème, il doit aussi être manipulable par des opérateurs de variation, minimiser l'épistasie (indépendance des gènes entre eux), permettre une transformation facile sur l'espace de recherche et générer, si possible, des solutions admissibles. Un bon codage doit ainsi :

- faciliter la définition et l'application d'opérateurs de variation (transformations génétiques : mutation, croisement, ...) permettant de couvrir correctement l'espace des individus ;
- être cohérent par rapport au problème traité et simple dans sa construction ;
- assurer une transition simple et efficace vers l'espace de recherche (et vice-versa).

1.6.2 La représentation binaire

Historiquement le codage utilisé par les algorithmes génétiques était un codage en vecteurs binaires (appelés aussi chaînes de bits) de longueur fixe l. Pour un problème mettant en jeu des variables entières, on peut représenter chacune de ces variables par un vecteur binaire

$a = (a_1,\ldots,a_l) \in \{0,1\}^l$. Dans le cadre des problèmes d'optimisation paramétrique continue, si on se donne une fonction à optimiser,

$$\mathcal{F} : \prod_{i=1}^{n}[u_i,v_i] \longrightarrow \mathbb{R} \quad (u_i < v_i)$$

on découpe la chaîne de bits en n segments binaires, généralement, de longueur égale : $l = nl_x$. Ensuite, on considère chacun des segments $(a_{(i-1)l_x+1},\ldots,a_{il_x})$ $(i = 1\ldots n)$ comme la représentation binaire d'une variable réelle $x_i \in [u_i,v_i]$. Le décodage du segment binaire se fait via l'utilisation d'une fonction $\Gamma^i : \{0,1\}^l \longrightarrow [u_i,v_i]$ définie par (cf. [10])

$$\Gamma^i(a_{(i-1)l_x+1},\ldots,a_{il_x}) = u_i + \frac{v_i - u_i}{2^{l_x} - 1}\left(\sum_{j=0}^{l_x-1} a_{(il_x-j)}2^j\right)$$

La figure 1.7 représente un exemple de codage binaire sur l'intervalle [0,1].

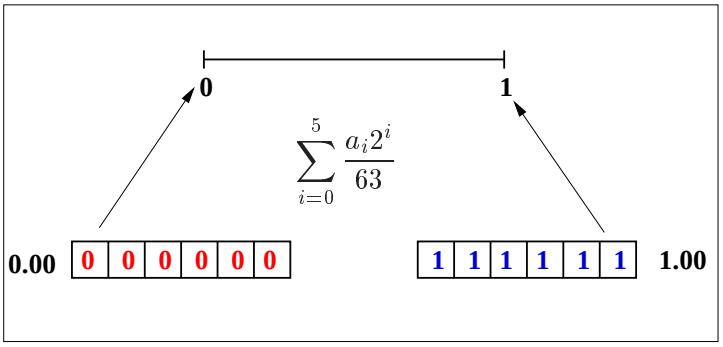


FIG. 1.7 – *Exemple de codage binaire*

Cependant, ce type de codage présente des inconvénients :

- deux éléments proches dans l'espace de recherche ne décodent pas nécessairement deux individus voisins en terme de distance de Hamming (nombre de bits différents). On évite cet inconvénient en utilisant un codage de Gray (cf. [41] pour les détails de ce type de codage) qui conserve une distance de Hamming de "1" entre deux individus consécutifs quelconques.
- de plus, pour des problèmes où l'on veut une grande précision, le codage binaire peut rapidement devenir inadapté.

Ces inconvénients amènent à utiliser un autre type de représentation : la représentation réelle.

1.6.3 La représentation réelle

Introduite indépendamment par l'école allemande des stratégies d'évolution [138][157], la représentation réelle a été introduite dans les années 80-90 notamment par Eshelman et Schaffer [51], Michalewicz [90] et Radcliffe [135] pour les autres types d'algorithmes évolutionnaires.

Le principe de cette représentation consiste à coder directement les variables du problème dans l'individu sans passer par le codage binaire intermédiaire. Ainsi, les individus ne sont plus des chaînes de bits mais des vecteurs réels. L'un des avantages majeur de cette représentation est de conserver les variables du problème dans le codage lui-même, ce qui lui permet une meilleure prise en compte de la structure même du problème [135]. Cette représentation directe des paramètres réels nécessite de définir de nouveaux opérateurs spécifiques (section 1.7.1). Nous présentons ci-après les principaux opérateurs génétiques classiques qui agissent sur les codages binaire et réel. Cette présentation permet d'introduire les opérateurs standards, pour le cas échéant, les utiliser ou s'en inspirer pour des problèmes spécifiques (cf. par exemple chapitre 4).

1.7 Opérateurs de variation

Les opérateurs de variation ont pour but de produire de nouveaux individus à partir de ceux préalablement sélectionnés. On distingue les opérateurs de croisement et les opérateurs de mutation.

1.7.1 Le croisement

Il est analogue à une reproduction sexuée en s'appuyant sur le principe que les enfants héritent des qualités de leurs parents. La forme standard de l'opérateur de croisement est $c : E \times E \to E \times E$, qui croise, avec une certaine probabilité p_c ($0 \leqslant p_c \leqslant 1$), deux parents $(P_1,P_2) \in E \times E$. D'autres formes de croisement sont possibles comme celle où un seul enfant est produit par plusieurs parents. Parmi les différents types majeurs de croisement, il y a :

Croisement binaire

C'est un opérateur sur $E \times E \to E \times E$, avec $E = \{0,1\}^l$. Il correspond à un échange de gènes (bits) entre les parents. Il en existe plusieurs variantes :

Le croisement 1-point

C'est le croisement le plus simple et le plus classique dans les algorithmes génétiques. Il consiste à sélectionner aléatoirement un point de coupure dans chacun des deux parents P_1 et P_2, et construire deux nouveaux individus (enfants) E_1 et E_2 en échangeant leurs gènes de part et d'autre de ce point (cf. figure 1.8).

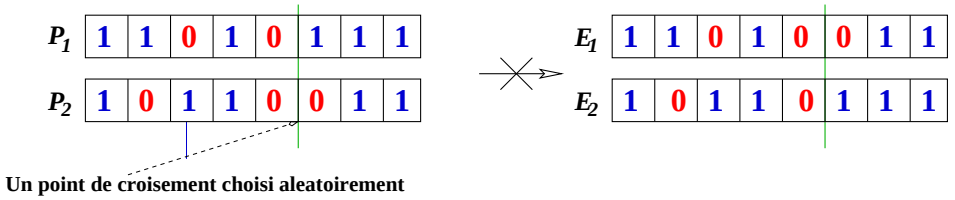


FIG. 1.8 – *Le croisement binaire à un point*

Le croisement multi-points

Le croisement multi-points est une généralisation du croisement à un point : au lieu de choisir un seul point de coupure, on en sélectionne k. La figure 1.9 représente un croisement à deux points : on sélectionne aléatoirement deux points de coupure dans P_1 et P_2 et on échange leurs gènes qui se trouvent entre ces deux points pour former E_1 et E_2.

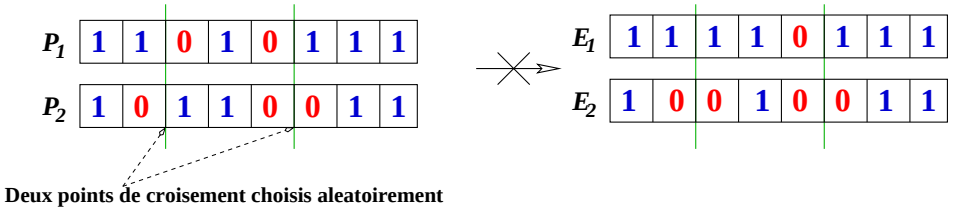


FIG. 1.9 – *Le croisement binaire à deux points*

Le croisement uniforme

Proposé par Syswarda [167], le principe de cette technique est le suivant.

Soient P_1 et P_2 les parents et E_1 et E_2 les enfants. Pour chaque position de E_1, on détermine aléatoirement (avec une probabilité 0.5) le parent qui donne son gène. Le second enfant E_2 se voit affecter le gène du parent qui n'a pas été retenu. En pratique, on choisit

uniformément un “masque de croisement”, pour chaque couple d'individus, qui représente un vecteur binaire de même longueur que les parents. La présence d'un “0” à la $i^{\text{ème}}$ position du masque laisse inchangés les bits des deux parents se situant à cette même position, et la présence d'un “1” entraîne un échange des bits correspondants (cf. figure 1.10).

D'autres opérateurs de croisement existent [117], ils peuvent soit amener des modifications à ceux présentés ci-avant, soit être spécifiques à une classe de problèmes (voir par exemple [99] pour un type tableau), mais obéissent néanmoins à un principe commun, l'échange d'information entre individus.

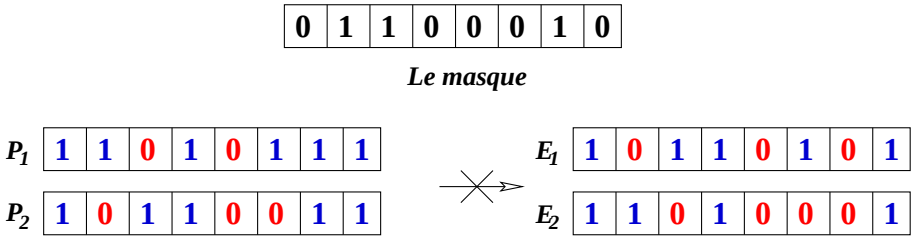


FIG. 1.10 – *Le croisement binaire uniforme*

Le croisements réel

Le croisement standard

Le croisement réel standard est très proche de celui décrit pour le codage binaire dans la section précédente. Il ne se différencie du croisement binaire que par la nature des éléments qu'il altère. La figure 1.11 montre un exemple de croisement réel à un point. Bien évidemment ce ne sont plus des bits qui sont échangés de part et d'autre du point de croisement mais des valeurs réelles.

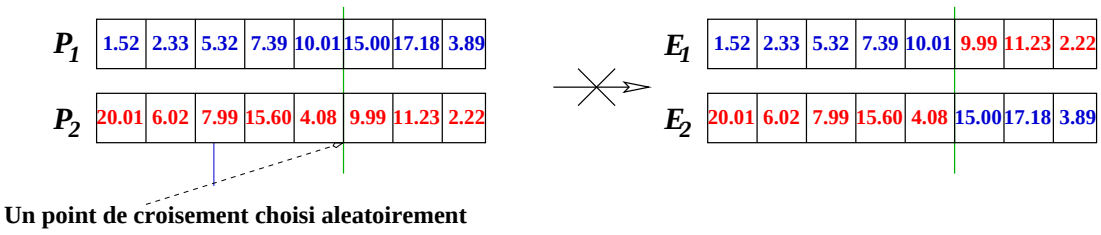


FIG. 1.11 – *Exemple de croisement réel à un point*

Le croisement arithmétique (barycentrique)

La représentation réelle permet cependant de développer toute une série de nouveaux types de croisement, principalement à base de combinaison linéaire de deux individus (vecteurs réels). On peut trouver dans [51][117] une étude du croisement arithmétique. Il consiste à choisir deux gènes $P_1(i)$ et $P_2(i)$ dans chacun des parents à la même position i, et à définir les gènes correspondants $E_1(i)$ et $E_1(i)$ chez les enfants par combinaison linéaire :

$$\begin{cases} E_1(i) = \alpha P_1(i) + (1-\alpha)P_2(i) \\ E_2(i) = (1-\alpha)P_1(i) + \alpha P_2(i) \end{cases}$$

où α est la réalisation d'une variable aléatoire uniforme appartenant à l'intervalle [0,1].

Diverses variantes de croisement relatives à la manière de choisir les enfants sur le segment joignant les deux parents sont possibles. On peut ainsi, pour permettre de générer des individus entre ou à l'extérieur du segment joignant les 2 parents, prévoir un paramètre α entre $(-\epsilon)$ et $(1+\epsilon)$, tout en faisant attention à se maintenir dans les bornes du domaine admissible. Ainsi, sur la figure 1.12, les parents P_1 et P_2 peuvent engendrer un enfant en E.

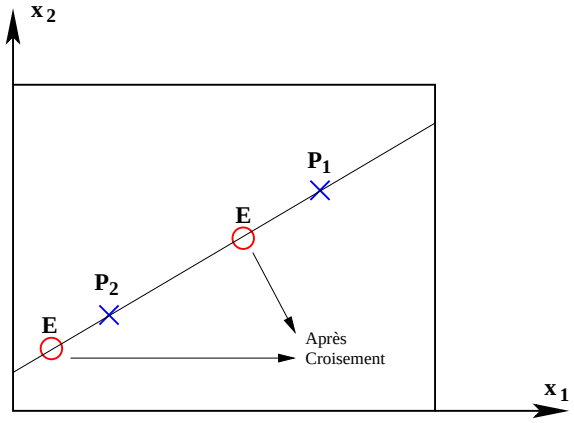


FIG. 1.12 – *possibilités de croisement barycentrique*

D'autres types de croisement peuvent être définis mais il faut toujours essayer de développer des opérateurs adaptés au problèmes que l'on traite (cf. section 4.2, chapitre 4).

1.7.2 La mutation

L'idée générale de la mutation est la modification (avec une certaine probabilité p_m, $0 \leqslant p_m \leqslant 1$) d'un ou plusieurs gènes de l'individu sélectionné, afin d'introduire de la variabilité dans la population. Cet opérateur agit de E dans E. Il peut :

- favoriser l'exploitation (si l'individu muté est proche de l'individu original).
- favoriser l'exploration (si l'individu muté est éloigné de l'individu original).

La mutation apporte aux algorithmes évolutionnaires la propriété d'ergodicité de parcours d'espace, et la réintroduction de diversité perdue.

La mutation binaire

La mutation binaire est une modification aléatoire de la valeur d'un gène qui se produit avec une probabilité fixée p_m par individu.

Les mutations les plus utilisées sont [95] :

- **1-bit mutation** : elle consiste à choisir une position uniformément dans un individu et changer la valeur du bit correspondant (cf. figure 1.13).

- $\frac{c}{l}$ **mutation** : Il s'agit de changer la valeur du bit de chaque position indépendamment avec une probabilité $\frac{c}{l}$, où l est la taille de l'individu (nombre de bits) et $c > 0$.

C'est un opérateur qui complémente aléatoirement les bits de la chaîne qui code la solution. Pour que la mutation soit effective, il faut que : $p_m.\frac{c}{l} > \frac{1}{l.l}$.

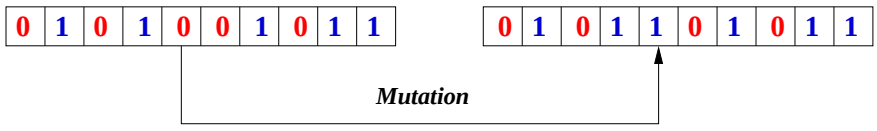


FIG. 1.13 – *Mutation binaire*

Intuitivement, le taux de mutation doit être lié à la qualité des enfants générés, en moyenne sur la population on peut ainsi définir des règles de mutation adaptatives telles que si beaucoup d'enfants mutés sont bons, il faut augmenter p_m, sinon il faut la diminuer.

La mutation réelle

Le principe de l'opérateur de mutation réelle consiste généralement à ajouter une perturbation aléatoire tirée selon une distribution de probabilité Gaussienne aux différentes composantes de l'individu X

$$x_i := x_i + \sigma N(0,1)$$

où σ et $N(0,1)$ sont respectivement l'écart type (déviation standard) de la mutation et une loi normale centrée d'écart type 1.

La difficulté de cette approche est l'ajustement des déviations standards des variables Gaussiennes utilisées. En effet, si la déviation standard est trop petite, les déplacements dans l'espace de recherche sont insuffisants au début de l'algorithme, et l'algorithme peut rester au voisinage d'un optimum local et ne permet pas de visiter des nouvelles régions de l'espace de recherche. Par contre, si l'écart est élevé, l'algorithme pourra accéder à une région contenant l'optimum mais la qualité de convergence ne sera pas bonne. Ainsi au début de l'évolution, la déviation standard σ doit être assez élevée pour explorer rapidement l'espace de recherche, et en fin de convergence devenir plus faible pour permettre une meilleure exploration des solutions.

Plusieurs stratégies adaptatives et auto-adaptatives ont été proposées pour ajuster l'écart au cours de l'évolution. Nous présentons les plus connues.

La mutation Gaussiène adaptative :

C'est la règle des 1/5 proposée par Rechenberg en 1973 [139]. Elle consiste à mettre à jour la valeur de σ toutes les générations comme suit :

$$\sigma(t) = \begin{cases} \sigma(t-1)\beta & si\ p_s < 0.2 \\ \sigma(t-1)/\beta & si\ p_s > 0.2 \\ \sigma(t-1) & si\ p_s = 0.2 \end{cases}$$

où $0 < \beta < 1$ est le taux d'adaptation de σ et p_s l'observation des mutations réussies[1] aux générations précédentes. Cette règle peut être appliquée toutes les k générations au lieu de toutes les générations.

1. une mutation est réussie si la performance de l'individu muté est meilleur que celle de l'individu avant mutation.

La mutation log-normale auto-adaptative isotrope [158]

Elle consiste à associer à chaque individu une déviation standard σ (σ est codé dans l'individu comme les variables du problème). Un individu est ainsi représenté par (X,σ). σ n'est plus fixe durant l'évolution mais varie stochastiquement selon une loi log-normale en utilisant un taux d'adaptation τ ($0 < \tau < 1$). La mise a jour se fait avant la mutation de la manière suivante :

$$\begin{cases} \sigma := \sigma exp(\tau N(0,1)) \\ x_i := x_i + N(0,\sigma) \end{cases}$$

Selon Schwefel une valeur optimale pour τ serait : $\tau \propto \frac{1}{\sqrt{n}}$.

La mutation log-normale auto-adaptative anisotrope

Elle consiste à attacher à chaque variable x_i de l'individu X une déviation standard σ_i. La relation qui liait une variable avant et après l'opération de mutation est à présent de la forme :

$$\begin{cases} \sigma_i := \sigma_i exp(\tau' N(0,1) + \tau N_i(0,1)) \\ x_i := x_i + \sigma_i N(0,\sigma) \end{cases}$$

$0 < \tau' < 1$ est le taux d'adaptation local et $0 < \tau < 1$ est le taux d'adaptation global. Selon les travaux de Schwefel il est recommandé d'utiliser les valeurs de τ et τ' suivantes :

$$\tau \propto \frac{1}{\sqrt{2\sqrt{n}}} \qquad \tau_l \propto \frac{1}{\sqrt{2n}}$$

La mutation corrélée : écart type matriciel [158]

Un individu est représenté par (X,[σ]) où [σ] est la matrice de covariance de la loi gaussienne. Le but de cette mutation est de permettre l'orientation des axes principaux des gaussiennes vers les directions d'amélioration du critère (cf. figure 1.14). Cette méthode peut être utilisée pour des espaces de recherche de petites dimensions mais reste très coûteuse dans le cas général car elle nécessite le codage complet de la partie triangulaire inférieure de la matrice de covariance [σ].

1.8 Familles d'algorithmes évolutionnaires

On distingue quatre grandes familles historiques d'algorithmes évolutionnaires – et les différences entre elles ont laissé des traces dans le paysage évolutionnaire actuel, en dépit d'une unification de nombreux concepts.

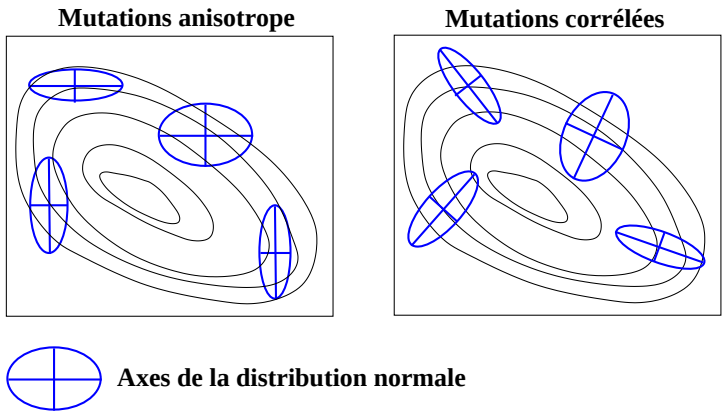


FIG. 1.14 – *Ellipsoïdes pour les directions de mutation*

1.8.1 Les algorithmes génétiques : GA

Les GA sont les algorithmes les plus connus et les plus couramment utilisés dans le calcul évolutionnaire. Ils ont été développés dans les années soixante par Holland [87], pour étudier, dans le cadre binaire, les mécanismes d'adaptation de populations dans le cadre de la psychologie/biologie. Ils ont été appliqués à l'optimisation paramétrique pour la première fois par De Jong en 1975 [46], qui a introduit de nombreux raffinements de cette technique. Ces raffinements sont devenus par la suite des principes de base, à considérer lors de toute application des AG à un problème réel. Cependant, le manque de puissance des ordinateurs à l'époque ne permettait pas leur application sur des problèmes réels de grande taille. Ce n'est que pendant les années 90, précisément avec l'apparition de l'ouvrage de référence écrit par Goldberg [65], que les GA se sont fait connaître dans la communauté scientifique.

Les algorithmes génétiques travaillent dans l'espace des chaînes de bits $[0,1]^n$ avec deux schémas :

- L'algorithme génétique standard (*Simple Genetic Algorithm* - SGA) : il utilise un schéma de sélection proportionnelle à la fitness (section 1.4.1) et un remplacement générationel.
- Le schéma stationnaire (*Steady-state GA* - SSGA) : ce schéma consiste à produire, à chaque génération, seulement un ou deux enfants à partir d'un ou deux parents sélectionnés selon leurs performances. Ces enfants remplacent alors des individus de la population parente, sélectionnés soit d'une manière déterministe (les pires dans la population), soit par tournoi inversé, ou selon les âges (les plus vieux).

1.8.2 La programmation évolutionnaire : EP

La programmation évolutionnaire a d'abord été développée par L.J. Fogel aux États-Unis dans les années 60 [57]. Elle a été mise au point pour la prédiction de séries temporelles par automates à états finis (figure 1.15). La table de transition des automates est modifiée grâce à des mutations uniformes dans l'alphabet discret correspondant. L'évaluation de la performance des individus correspond au nombre de symboles prédits correctement. Chaque automate de la population parente génère un enfant par mutation, et les meilleures solutions entre les parents et les enfants sont sélectionnées pour survivre. La principale caractéristique des EP est qu'il n'existe pas d'opérateur de croisement contrairement aux algorithmes génétiques.

Récemment, c'est D.B. Fogel qui a enrichi cette classe d'algorithmes en travaillant notamment sur des représentations réelles [55, 56] et des mutations basées sur des lois normales. En plus, la sélection déterministe est remplacée par un tournoi stochastique. Comme pour la théorie des GA (cf. section 1.9), celle concernant les EP est encore pauvre.

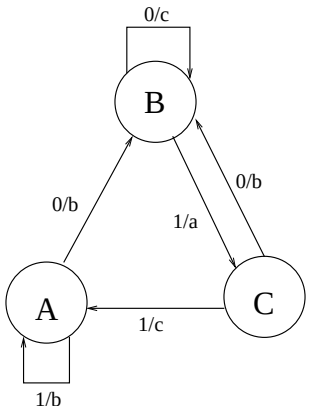


FIG. 1.15 – *Exemple d'un automate à états finis ayant trois états différents* $S = \{A,B,C\}$, *un alphabet d'entrée* $I = \{0,1\}$, *et un alphabet de sortie* $O = \{a,b,c\}$. *Chaque arête entre deux états indique une transition possible, et la fonction de transition* $\delta : S \times I \rightarrow S \times O$ *est spécifiée par les labels au niveau des arêtes ayant la forme* i/o, *signifiant que* $\delta((s_i,i)) = (s_j,o)$.

1.8.3 Les stratégies d'évolution : ES

Les stratégies d'évolution ont été développées en Allemagne par Rechenberg [138][139] et H.P. Schwefel [157][158] vers le milieu des années 60. Elles sont dédiées à la résolution

de problèmes d'optimisation numérique dans l'espace des vecteurs de réels. En 1965, Rechenberg a introduit l'algorithme (1+1)-ES qui fait évoluer un seul individu et utilise la mutation Gaussienne (section 1.7.2) pour assurer cette évolution. Il a proposé la règle 1/5 pour l'adaptation de la déviation standard de la mutation [138]. En 1977, Schwefel a introduit deux types d'ES : (μ,λ)-ES et $(\mu+\lambda)$-ES, avec $1 \leqslant \mu < \lambda$. Ces deux stratégies se différencient au niveau de la sélection déterministe utilisée (cf. section 1.4.3). Par ailleurs, des nouvelles approches de mutation et de croisement ont été mises en place. Sont alors apparues les notions d'auto-adaptativité pour la mutation ainsi que différents types de croisement entre vecteurs réels illustrés dans la section 1.7. Contrairement aux EP le croisement joue un rôle important dans les ES auto-adaptatives.

Du point de vue théorique nous pouvons citer les travaux de Bäck, Rudolph et Schwefel [11] qui ont donné un résultat de convergence forte sur les fonctions convexes dans le cas des (1+1)ES et $(1,\lambda)$ES.

1.8.4 La programmation génétique : GP

Dans cette section nous présentons plus en détail la programmation génétique, que nous utiliserons dans le chapitre 5 pour la représentation par des systèmes de fonctions itérées.

La Programmation Génétique a été introduite par John Koza en 1992 [103]. Son objectif initial était de faire évoluer des sous-programmes du langages LISP (figure 1.16). Grâce aux ouvrages de Koza [103, 104], l'utilisation de GP s'est étendue pour la résolution de nombreux types de problèmes dont les solutions peuvent être représentées par des structures arborescentes, des graphes [169, 143], des images [38], des modèles rhéologiques [159] ...

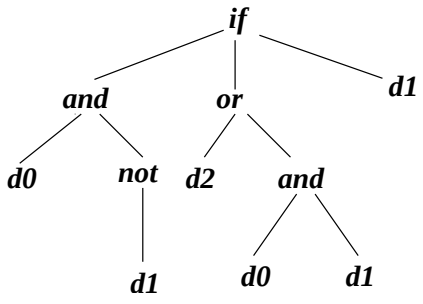


FIG. 1.16 – *Exemple d'une solution GP en LISP :* $\{d0,d1,d2\}$ *est un ensemble d'instructions constituant les terminaux, et* $\{if,and,or\}$ *sont des opérateurs LISP constituant les nœuds.*

Représentation d'un individu

Une solution dans l'espace génotypique est représentée par une structure arborescente dynamique, considérée comme un *programme*.

En tant que structures d'arbres –graphes acycliques–, les programmes génétiques requièrent la donnée de l'ensemble des fonctions de base et des terminaux, décrivant alors le "langage de programmation" du problème à résoudre. En effet, avec les fonctions prenant place au niveau des nœuds et des terminaux s'assimilant à des feuilles de l'arbre, nous obtenons un jeu d'instructions constituant les briques élémentaires qui, assemblées judicieusement, doivent fournir une solution. Par exemple, dans une représentation fonctionnelle, une fonction à deux dimensions ($f(x,y)$) peut être construite à partir de :

1. un ensemble de terminaux ($\mathcal{T}$) : les variables, les constantes universelles, fonctions sans arguments (rnd(), time(), ...).
2. un ensemble de nœuds ($\mathcal{N}$) : des opérateurs : $*$, $-$, $+$, des fonctions : $sin()$, $cos()$, $log()$, $exp()$ ...

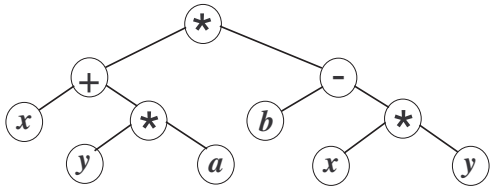


FIG. 1.17 – *Exemple d'une solution en GP : $\mathcal{N} = \{*, +, -\}$ et $\mathcal{T} = \{x,y,\mathbb{R}\}$. L'espace exploré est celui des polynômes réels à 2 variables. Ici $f(x,y) = (x + ay)(b - xy)$*

Algorithme

Le principe de la programmation génétique est celui d'un algorithme génétique appliqué aux arbres. On part d'un ensemble de programmes choisis d'une manière aléatoire parmi tous les programmes possibles de l'espace de recherche. La performance de chaque programme est obtenue à l'aide d'une fonction d'évaluation. Puis on élabore le meilleur programme possible au cours des différentes générations grâce à la sélection et aux opérateurs de croisement et de mutation. Le schéma d'évolution utilisé est souvent de type SSGA (section 1.8.1), mais avec une très grande taille de population.

Opérateurs de variation en GP

Les principes des opérateurs de variation utilisés pour la programmation génétique sont analogues à ceux utilisés pour les algorithmes génétiques. Cependant, leur mise en oeuvre peut s'avérer techniquement plus difficile du fait des données qu'ils manipulent.

Le croisement

Pour effectuer le croisement entre deux individus, qui sont des programmes, on choisit au hasard (généralement fait par tirage uniforme) un noeud de l'arbre dans chacun des programmes et on échange les sous-arbres obtenus à partir de chacun de ces noeuds d'un arbre à l'autre (cf. figure 1.18).

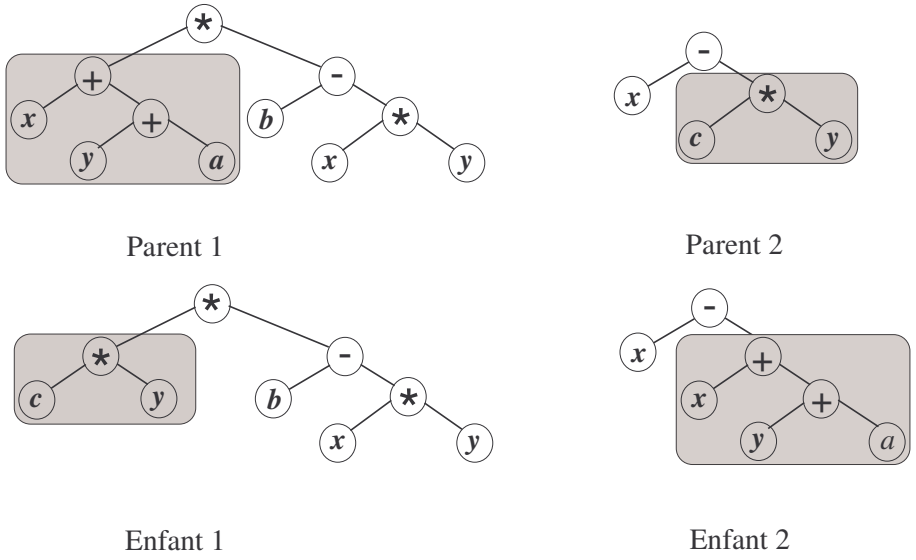


FIG. 1.18 – *Exemple de croisement de deux individus en GP*

La mutation en GP

Le GP traditionnel proposé par Koza [103] n'utilise pas d'opérateurs de mutation. Pour assurer l'accès à toutes les primitives du langage de recherche (e.g. LISP) et assurer la diversité génétique, on utilise des populations de très grande taille, pour avoir le maximum d'information. C'est en 1996 que la mutation a été introduite par Angeline [8, 7] dont le but de réduire la taille de la population.

On distingue de multiples sortes de mutations du fait de la complexité des structures arborescentes, dont les capacités exploratoires peuvent être locales ou, à l'inverse, de grande envergure. Parmi les différentes mutations, les plus courantes sont :

– **Mutation par insertion** (*grow mutation*) : on ajoute un nœud et des feuilles complémentaires n'importe où dans l'arbre (figure 1.19).

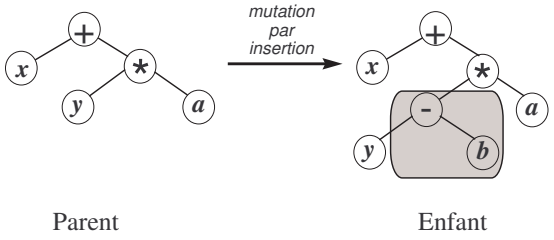


FIG. 1.19 – *Exemple de la mutation par insertion en GP.*

– **Mutation par promotion** (*shrink mutation*) : on supprime un nœud interne et l'un de ses fils remonte prendre sa place (figure 1.20).

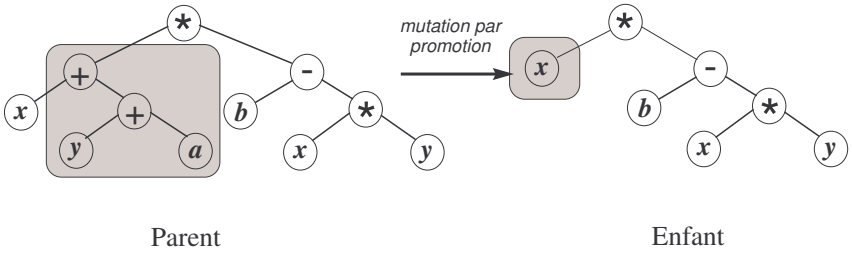


FIG. 1.20 – *Exemple de la mutation par promotion en GP.*

– **Mutation d'un nœud** (*cycle mutation*) : on change un nœud de l'arbre en un autre (figure 1.21).

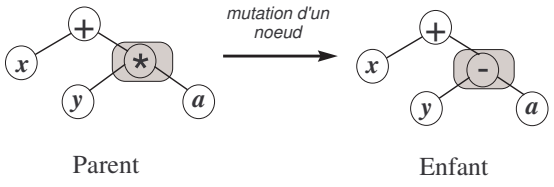


FIG. 1.21 – *Exemple de mutation d'un nœud en GP*

– **Mutation d'une branche** (*random mutation*) : on élague une branche de l'arbre et on la remplace par un sous-arbre généré aléatoirement (figure 1.22).

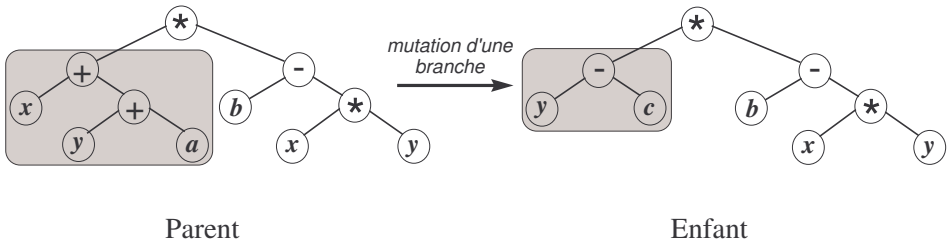


FIG. 1.22 – *Exemple de mutation d'une branche en GP*

Dans le cas où les terminaux peuvent être des constantes numériques, d'autres mutations ont été introduites :

– **Mutation des constantes** : on modifie quelques constantes selon une loi Gaussienne ou uniforme [7].
– **Mutation optimisée des constantes** : on applique quelques itérations d'un *hill climber* aléatoire en vue d'affiner les constantes [154].

1.9 Théorie des algorithmes génétiques

1.9.1 Théorie des schémas

Historiquement, les algorithmes génétiques binaires ($E = \{0,1\}^l$) sont les plus anciens et ont donc été les plus étudiés sur le plan théorique. La théorie des schémas est l'une des premières analyses heuristiques de ces algorithmes. Nous commençons dans cette section par donner quelques définitions fondamentales afin de donner le résultat principal de cette théorie.

Définition 1.1

On appelle séquence A de longueur l une suite $A = a_1 a_2 \dots a_l$ avec $\forall i \in [1,l]$, $a_i \in \{0,1\}$.

Définition 1.2

Un schéma de longueur l est une séquence $\mathcal{H} \in \{0,1,*\}^l$, où le symbole $*$ désigne indifféremment 0 ou 1.

Définition 1.3

On dit qu'une séquence A est une instance d'un schéma $\mathcal{H} = b_1 b_2 \dots b_l$ si pour tout i tel que $b_i \neq *$ on a $a_i = b_i$.

Définition 1.4

L'ordre d'un schéma $\mathcal{H}$ est le nombre de bits fixes de $\mathcal{H}$, c'est le nombre de caractères autres que $*$ qu'il contient. On note l'ordre de $\mathcal{H}$ $o(\mathcal{H})$.

EXEMPLE— Le schéma $\mathcal{H} = 1 * 111 = \{1011,1111\}$ a pour ordre $o(\mathcal{H}) = 3$

Définition 1.5

La longueur utile d'un schéma, noté $l(\mathcal{H})$, est la distance entre la première et dernière position des bits fixes.

Dans l'exemple précédent $l(\mathcal{H}) = 4$

Théorème 1.1

Si on note $m(\mathcal{H},t)$ le nombre de représentants de $\mathcal{H}$ dans la population à la génération t, la probabilité de survie d'un schéma après les opérations de reproduction, croisement et mutation s'obtient par :

$$E(m(\mathcal{H},t+1)) \geqslant m(\mathcal{H},t)\frac{\overline{f}(\mathcal{H},t)}{\overline{f}(P_t)}(1 - p_c\frac{l(\mathcal{H})}{l-1} + p_m o(\mathcal{H}))$$

où p_m est la probabilité de mutation d'un bit dans une séquence et p_c la probabilité de croisement. $\overline{f}(\mathcal{H},t)$ est la performance moyenne des représentants A_i de $\mathcal{H}$ présents dans la population P_t, donnée par

$$\overline{f}(\mathcal{H},t) = \sum_{A_i \in \mathcal{H} \cap P_t} \frac{f(A_i)}{m(\mathcal{H},t)}$$

et $\overline{f}(P_t)$ est la valeur moyenne de la performance pour les individus de toute la population P_t.

Ce théorème montre que le nombre de représentants d'un schéma dans la population suit une progression géométrique si $c_t = \frac{\overline{f}(\mathcal{H},t)}{\overline{f}(P_t)} \geq \alpha > 1$. Une conséquence directe est que les schémas courts (de longueur utile et d'ordre faibles) de fitness au dessus de la moyenne de la population à l'instant t ont tendance à envahir les futures populations: de tels schémas sont classiquement appelés blocs de construction (*building blocks*). Cette remarque permet alors de dire que si un optimum de la fonction fitness correspond à l'intersection de tels blocs, l'algorithme le trouvera facilement.

Ce résultat n'est qu'une introduction à la théorie des schémas. Pour d'autres modèles théoriques et des extensions basées sur cette théorie on pourra se référer à [65][67][172].

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 P_t est différente de la fitness moyenne du schéma [134]. 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- Si les blocs de constructions mènent vers un optimum secondaire, l'algorithme convergera avec difficultés vers un optimum global. Goldberg a formalisé cette idée en introduisant la notion de "*deception*" [65] : une fonction est dite trompeuse (deceptive) si les schémas les plus performants ne contiennent pas l'optimum. Il existe des exemples de fonctions trompeuses faciles à optimiser pour les algorithmes génétiques [174]. Par ailleurs, seuls les fonctions à variables séparables sont totalement (pour tout ordre de schéma) non trompeuses [40].

D'autres approches ont été développées dans les années 90 dont l'approche probabiliste, utilisant une modélisation par des chaînes de Markov, dont nous présentons les principaux résultats.

1.9.2 Modélisation par chaînes de Markov

On se place dans l'espace binaire ($E = \{0,1\}^l$). Soit $X_t = (x_1, x_2 \dots x_m)$, $x_i \in E$, la population à l'instant t de taille m. Le passage de X_t à X_{t+1} se décompose en trois étapes : Mathématiquement la suite $(X_t)_{t\in\mathrm{N}}$ est une chaîne de Markov d'espace d'état E^m. La loi

de X_t est déterminé de manière unique par la donnée de la loi de X_0 (en général, la loi uniforme sur E^m) et de la matrice de transition qui donne les probabilités de passer de la génération t à la génération $t+1$.

Le mécanisme de transition possède toutefois des propriétés essentielles qui font l'intérêt et la puissance de cette modélisation [31]

- Il est homogène : la population à l'instant t dépend uniquement de la population à l'instant $t-1$, les probabilités de transition étant indépendantes du temps et des populations antérieures.

– Il est irréductible : étant données deux populations arbitraires x et y, la probabilité partant de x d'obtenir y en un nombre fini de transition est non nulle :

$$\forall x,\, y \in E^m \; \exists r \in \mathbb{N} \; P(X_{n+r} = x / X_n = y) > 0.$$

Autrement dit, le mécanisme de transition permet d'explorer tout l'espace des populations.

Ces propriétés permettent de conclure l'ergodicité de cette chaîne, et qu'elle possède une unique mesure de probabilité stationnaire et invariante, i.e, que le comportement de la chaîne converge asymptotiquement vers une loi de probabilité indépendante de la population de départ x : pour tout $y \in E^m$

$$\lim_{n \to \infty} P(X_t = y / X_0 = x) = \mu(x) \; avec \; \mu(x) > 0$$

Plusieurs résultats théoriques, faisant intervenir cette modélisation par chaîne de Markov, ont été obtenus. Dans ce cadre nous pouvons citer les travaux de Davis et Principe [42], Nix, Vose [125], Rudolph [141], et l'un des résultats les plus avancés est celui publié par Cerf [31] qui a appliqué la théorie de Freidin-Wentzell [120] pour prouver qu'un algorithme génétique simple permet de revenir à l'optimum après une perturbation d'un état initial. Plus précisément il a démontré, sous des hypothèses très techniques, que si la population dépasse une taille critique dépendant des paramètres de l'algorithme et de la fonction fitness $\mathcal{F}$ alors il y a convergence (en un temps fini) vers les optima globaux de $\mathcal{F}$. Ces résultats comptent parmi les premiers résultats mathématiques se rapportant à la convergence des algorithmes génétiques. On en retiendra que cette taille critique croît linéairement avec la longueur des chaînes de bits.

Cependant, ces résultats restent inapplicables dans la pratique où des opérateurs plus compliqués et des techniques de raffinement (mise à l'échelle, partage, adaptativité, élitisme, ...) sont utilisés.

1.10 Conclusion

L'avantage des algorithmes évolutionnaires est d'être applicables à une vaste gamme de problèmes, sans caractéristiques particulières : fonction et/ou contraintes chahutées, multimodales, multi-objectifs (cf chapitre 6). De plus, ils sont capables de travailler sur n'importe quel espace de recherche, continu, discret, mixte, espace d'arbre, ... Par contre, le succès et la durée de la recherche dépendent fortement de la représentation (espace des génotypes) et des opérateurs de variation (mutation, croisement) choisis. Le chapitre 4

donnera des exemples de représentations et d'opérateurs spécifiques ainsi validés. Comme le choix de la représentation, le choix de la fonction de performance représente un point crucial dans le succès de la méthode d'évolution utilisée, car toutes ces variantes d'algorithmes requièrent un nombre important d'évaluations de la fonction objectif. De plus, si le caractère robuste de ces méthodes nous permet d'appréhendre des problèmes difficiles, le temps de calcul de chacune des fonctions coût peut être une barrière à leur utilisation. Une possibilité pour traiter ce problème est bien sûr la distribution des calculs sur plusieurs processus, qui est très facile conceptuellement.

Chapitre 2

Optimisation de formes

Sommaire

2.1 Introduction

Nous nous intéressons dans ce chapitre à l'état de l'art de l'optimisation de formes de structures en mécanique des solides. Le problème consiste, par exemple, à trouver la forme optimale d'une structure, qui soit à la fois de poids minimal et de rigidité maximale, mais on pourrait aussi optimiser d'autres critères : minimisation d'un état de tension, d'un déplacement ; maximisation d'une fréquence fondamentale ou d'une charge critique, conception pour un coût imposé. Bien entendu on doit aussi respecter d'autres contraintes, qui peuvent porter sur la réponse de la structure à son environnement : les limitations fixent alors des valeurs extrêmes aux déplacements, aux fréquences propres de vibration, aux charges critiques d'instabilité locale ou globale, le coût de fabrication ou encore l'aérodynamisme de la structure, etc.

Traditionnellement, les formes étaient réalisées par essais successifs, sous la direction d'experts qui guidaient la sélection de chaque modification de la forme. Cette façon de faire "manuelle" est très coûteuse et imprécise. L'utilisation de logiciels de modélisation numérique et d'optimisation s'est ainsi développée, permettant d'automatiser les étapes de recherche de la forme optimale.

Dans tous les domaines de la mécanique des structures, l'impact de la bonne conception d'une pièce est très important sur sa résistance, sa durée de vie et son utilisation en service. On trouve de nombreux exemples de ce genre dans les secteurs de pointe que sont la recherche spatiale, l'aéronautique, l'automobile, la conception navale, la micromécanique, la mécanique de précision ou les ouvrages d'art en génie civil..., pour lesquels toute économie de poids représente un gain en matière première et en coût.

L'optimisation de structures peut se scinder en trois grandes familles. Historiquement, chacune a été abordée par ordre croissant de difficulté et de généralité (cf. figure 2.1) :

- **optimisation des dimensions**, voir [52], [54], [60] et [61] : ne prend en compte que l'optimisation des dimensions des éléments d'une structure. Pour ce type de

problème la géométrie et la topologie de la structure restent inchangées au cours de l'optimisation.

- **l'optimisation géométrique**, voir [28] [130] et [179] : appelée aussi méthode de variation de domaine, permet de faire évoluer la géométrie d'une structure tout en conservant sa topologie fixée au début de l'optimisation.
- **l'optimisation topologique**, voir [4], [18], [59], [91] et [99] : appelée aussi optimisation de forme généralisée, permet de contrôler à la fois la géométrie et la topologie de la structure. Le problème consiste a trouver la répartition optimale de matière dans un domaine initial donné. Contrairement à l'optimisation géométrique, elle n'impose pas un choix a priori de la connectivité du domaine.

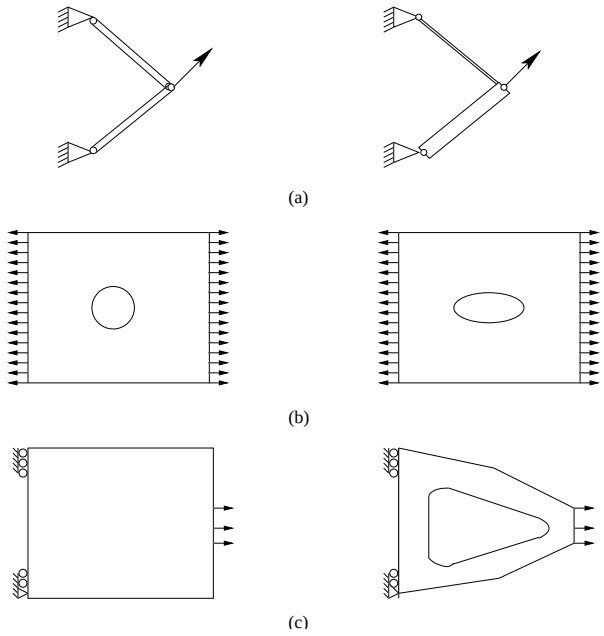


FIG. 2.1 – *Les 3 classes de problèmes d'optimisation de structures [72]. (a) Dimensionnement. (b) Forme avec topologie fixée. (c) Topologie variable.*

Dans ce chapitre nous donnons une idée de l'état de l'art en terme de méthodes spécifiques d'optimisation de structures mécaniques dans la littérature.

2.2 Optimisation de dimensionnement

L'optimisation des dimensions (cf. figure 2.1-(a)) ne permet de modifier que la section droite ou l'épaisseur transversale des composantes d'une structure dont la forme et la to-

pologie sont fixées. Elle est spécifique pour une classe de formes pré-choisie; il faut par exemple trouver l'épaisseur d'une coque, la taille d'une barre dans un treillis, ou encore le rayon d'une tige métallique dans un assemblage. Cette approche est alors limitée à la variation d'un certain nombre de paramètres de contrôle.

Évolution historique

A l'origine, l'optimisation de structures mécaniques était principalement limitée au dimensionnement de treillis ou de portique. Depuis les années 60 plusieurs méthodes ont été introduites et utilisées dans les problèmes de dimensionnement. Nous pouvons citer en premier lieu Schmit [146] qui a introduit une théorie moderne de l'optimisation de structures fondée sur le concept de synthèse structurale: il a indiqué la possibilité de coupler les méthodes d'analyse par éléments finis, pour une classe particulière de structures, aux méthodes de programmation mathématique. Depuis lors, plusieurs travaux ont permis de développer et généraliser ces concepts, citons notamment ceux de Gellatly et Gallagher [60], [61], Karnes et Tocher de Boeing [101]. Bien que le développement ait été rapide, il a fallu se rendre compte, au début des années 70, que la combinaison des analyses par éléments finis et des méthodes générales de la programmation non linéaire était inappliquable pour la résolution de problèmes réels. Plus tard, une approche basée sur la notion intuitive de critères d'optimalité, donnée par Berke [22], et les méthodes duales introduites par Fleury [52], a été proposée [54]. A la fin des années 70, les concepts d'approximation (qui consistent à remplacer le problème d'optimisation réel par une suite de sous-problèmes approchés) furent combinés à la formulation duale pour créer une méthode puissante pour la minimisation du poids des systèmes structuraux [54]. Le problème d'optimisation de dimensionnement fut également étendu au problème des éléments de flexion (Fleury et Sander [53]), à l'amélioration du comportement vibratoire et à la stabilité de l'équilibre.

2.3 Optimisation de domaine

Encore appelée *optimisation géométrique*, voir par exemple [122] [130] et [28], elle consiste à rajouter, par rapport aux problèmes de dimensionnement des structures, d'autres variables qui permettent des mouvements des frontières plus importants.

Méthode de variation de domaine

La méthode classiquement utilisée en optimisation géométrique de formes est la méthode de variation de domaine [28], appelée aussi *analyse de sensibilité*. Elle consiste en des

variations successives d'un domaine initial et est basée sur le calcul d'un gradient de la fonction-objectif par rapport aux variables définissant la forme (les variables d'optimisation sont les frontières du domaine, paramétrées par exemple par des segments ou des splines).

Le calcul des variations a été réalisé depuis 1908 par Hadamard [74] en déplaçant la frontière le long de sa normale et en calculant la variation induite de la fonctionnelle. Ensuite une théorie mathématique pour le contrôle par une forme a été développée à partir des années 70. On peut notamment citer les travaux de J. Céa [28, 29], F. Murat et J. Simon [122], O. Pironneau [130], J.P Zolésio et J. Sokolowski [179], B. Rousselet [140], M. Masmoudi [114],...

Cette approche est basée sur la définition ou représentation d'une forme et l'analyse de sensibilité (calcul des dérivées de la réponse structurale par rapport aux variables de conception).

Choix de la représentation

Différentes approches sont possibles pour la représentation géométrique. Il existe deux types de paramétrisation, chacune d'elles ayant des avantages et des limites. La première est une paramétrisation discrète à partir des points du maillage et la seconde est une paramétrisation par courbe (Bézier,spline).

Représentation discrète

Le but de cette représentation est d'utiliser les coordonnées des noeuds du maillage d'un modèle éléments finis comme variables de conception. Cette approche trouve une application naturelle pour l'optimisation de la configuration de treillis de barres. Une première application de ce mode de représentation géométrique aux structures continues semble avoir été donnée par Zienkiewicz et Campbell [177]. Cependant, le choix des coordonnées des noeuds en tant que variables de conception présente de nombreux désavantages [177] [131] :

- Lorsque les variables de conception sont les noeuds des éléments finis, les solutions obtenues sont rarement réalisables pratiquement. Les contours peuvent présenter des discontinuités dans leurs dérivées tangentielles, des points d'inflexion...
- une autre difficulté est liée à la définition et à la mise à jour du maillage au cours du processus itératif, ce qui alourdit le code de calcul.

Représentation polynomiale de la frontière

Cette technique se base sur la possibilité de ramener la représentation d'une forme à la connaissance d'une ou plusieurs courbes. En effet, seuls les points (généralement des points de contrôle sur les bords) de ces courbes sont nécessaires à la construction du modèle éléments finis. Bhavikatti et Ramakrishnan [23] utilisent des expressions polynomiales dont les coefficients servent de variables de conception pour caractériser une forme. Dans les années suivant cette publication, la plupart des chercheurs ont utilisé les polynômes pour décrire les frontières (par exemple Pedersen et Laursen [129] et les références dans [76]). L'idée se repose sur l'utilisation de points de contrôle pour définir des courbes paramétriques comme les courbes de Bézier, les B-splines ou les courbes rationnelles paramétriques (cf. [26] pour plus de détails sur ces courbes).

Analyse de sensibilité

L'analyse de sensibilité consiste à trouver des informations quantitatives sur la façon dont la réponse d'une structure est affectée par de petites modifications des variables qui définissent cette structure. Deux approches ont été proposées dans la littérature pour le calcul de sensibilité [76]. La première est basée sur la dérivation des équations d'équilibre continues [35], et la seconde consiste à calculer les dérivées des déplacements au moyen de l'équation suivante [26], résultant de la dérivation des équations d'équilibre discrétisées en élasticité linéaire :

$$K\frac{dU}{dx} = \frac{dF}{dx} - U\frac{dK}{dx}$$

où U est le vecteur déplacement, K est la matrice de raideur, F le vecteur force et x une variable de conception.

Discussion

Plusieurs problèmes apparaissent lors de la mise en oeuvre et de l'exploitation de la méthode de variation de domaine :

- elle nécessite une bonne intuition de la solution lors du choix de forme initiale : si la structure optimale est trop différente de la structure de départ il sera nécessaire durant le processus d'optimisation de remailler le domaine, afin de conserver une qualité de maillage suffisante, ce qui est coûteux et peut poser des problèmes techniques (des frontières qui vont se chevaucher, etc). De plus elle se montre instable pour de grandes variations du domaine.

- elle ne permet pas de modifier la topologie de la forme initiale (e.g. ajouter ou supprimer des trous). En effet elle présuppose que la forme de la structure finale est compatible avec une topologie fixée au départ et interdit donc les modifications de la configuration de la structure, c'est à dire que la structure gardera le même nombre de composantes et ne sera probablement pas optimale car dans un problème complexe, il est très difficile, voir impossible, de choisir a priori la topologie optimale.
- Elle ne permet pas toujours d'assurer l'existence de solution à moins d'imposer des conditions de régularité sur la frontière que l'on cherche à optimiser.

Toutefois, pour générer des structures de forme optimale sans aucun a priori sur la topologie, il est établi depuis longtemps qu'il faut sortir du cadre de l'optimisation de forme paramétrique et pouvoir se passer de fonction de forme pour décrire le domaine. Une alternative satisfaisante consiste à formuler le problème en fonction d'une distribution optimale de la matière disponible, car aucune hypothèse n'est nécessaire sur la solution.

Dans la section suivante nous allons considérer les méthodes utilisées en optimisation topologique de formes.

2.4 Optimisation topologique de formes : méthodes déterministes

2.4.1 Méthode d'homogénéisation

Nous présentons dans cette section une première méthode d'optimisation topologique qui est l'approche introduite en 1988 dans [18] et maintenant standard, utilisant l'homogénéisation. La forme optimale est recherchée comme une densité de matière en chaque point (comprise entre 0 et 1) et une micro-structure locale décrivant la forme du mélange matière/vide. L'introduction de telles micro-structures dans l'espace de recherche revient à relaxer le problème initial (au sens mathématique du terme) pour obtenir un problème bien posé [4]. Les algorithmes numériques issus de telles formulations convergent vers des solutions généralisées (non physiques) faites de matériaux composites. Elles peuvent être post-traitées pour obtenir une solution admissible avec une densité booléenne [2]. Dans ce qui suit, nous présentons brièvement les grandes lignes de la méthode d'homogénéisation décrite dans l'article de Allaire, Bonnetier, Francfort et Jouve [2]. Pour plus de détails sur les méthodes d'optimisation topologique de forme par les méthodes d'homogénéisation voir aussi Bendsoe et Kikuchi [18], Bendsoe [19],Allaire et Kohn [4].

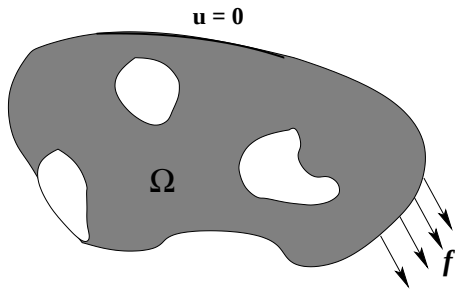


FIG. 2.2 – *Exemple de structure soumise à des forces et des conditions au bord.*

Formulation du problème d'optimisation de forme

Soit Ω un domaine de référence borné de $\mathbb{R}^n$, $n = 2{,}3$, constitué d'un matériau élastique, linéaire, homogène et isotrope, de loi de Hooke donnée par le tenseur d'ordre 4

$$A = (\kappa - \frac{2\mu}{n}) I_2 \otimes I_2 + 2\mu I_4$$

où κ et μ sont respectivement les modules de compression et de cisaillement du matériau, I_2 est l'identité dans l'espace des tenseurs d'ordre 2, noté S_2 (espace des applications linéaires symétriques de $\mathbb{R}^n$ dans lui même), et I_4 est l'identité dans l'espace des tenseurs d'ordre 4 (espace de toute les formes linéaires symétriques de S_2 dans S_2). Pour un matériau linéaire, la loi de Hooke établit une relation linéaire entre le tenseur des contraintes σ et le tenseur des déformations ε

$$\sigma = A\varepsilon$$

Le domaine Ω est soumis à une force surfacique f sur une partie de sa frontière $\partial\Omega$, qu'on note $\partial\Omega_F$, et des conditions aux limites homogènes sur les déplacements (bloquage) sont imposées sur l'autre partie $\partial\Omega_D$ (cf. figure 2.2). On demande à ces forces de respecter la condition d'équilibre

$$\int_{\partial\Omega} f.u = 0$$

pour tout champs de déplacement $u(x) = b + Mx$ correspondant à une rotation infinitésimale (avec $M = -M^t$ matrice antisymétrique).

Une forme admissible ω est un sous-ensemble du domaine de référence Ω obtenu en enlevant un sous ensemble de trous dans Ω. Les nouvelles frontières ainsi générées sont libres de toute traction. Pour que ce domaine ω soit admissible, il faut aussi que sa frontière $\partial\omega$ contienne la partie du bord $\partial\Omega$ où les forces surfaciques ne sont pas nulles. On a le

système d'équations d'élasticité linéaire dans ω

$$\begin{cases} \sigma_\omega & = & A\varepsilon(u_\omega) & \\ div\sigma_\omega & = & 0 & dans\ \omega \\ u_\omega & = & 0 & sur\ \partial\Omega_D \\ \sigma_\omega.\mathbf{n} & = & f & sur\ \partial\Omega_F \\ \sigma_\omega.\mathbf{n} & = & 0 & sur\ \partial\omega \setminus \partial\Omega \end{cases}$$

avec

$$\varepsilon(u_\omega) = \frac{1}{2}(\nabla u_\omega + \nabla^t u_\omega)$$

$u_\omega(x)$ le champ de déplacement (une fonction de ω dans $\mathbb{R}^n$), solution du système, σ_ω et $\varepsilon(u_\omega)$ sont les tenseurs symétriques respectivement de contrainte et déformation. On cherche à maximiser la rigidité d'une structure, que l'on peut envisager sous plusieurs critères pratiques. Le choix ici s'est porté sur la compliance (ou souplesse), définie comme le travail des forces extérieures ou l'énergie élastique de la structure. Elle est donnée par

$$c(\omega) = \int_{\partial\Omega} f.u_\omega$$

qui par intégration par parties s'écrit encore

$$c(\omega) = \int_\omega A\varepsilon(u_\omega).\varepsilon(u_\omega) = \int_\omega A^{-1}\sigma_\omega.\sigma_\omega$$

Le problème fondamental d'optimisation topologique consiste à minimiser la souplesse de la structure avec une borne supérieure sur le volume (le poids de la forme est supposé proportionnel à son volume), c'est à dire à trouver la forme admissible $\omega \subset \Omega$ qui minimise la compliance avec une contrainte sur le poids. En introduisant un paramètre de pénalisation $l > 0$ (multiplicateur de Lagrange), qui équilibre les deux objectifs contradictoires de maximisation de la rigidité et de minimisation du poids, on se ramène à un problème de minimisation sans contrainte

$$\inf_{\omega\subset\Omega}(J(\omega) = c(\omega) + l|\omega|) \tag{2.1}$$

où $|\omega|$ est le volume du domaine ω

De manière alternative, on peut également envisager la configuration optimale comme étant celle qui minimise le volume de matière utilisé avec une contrainte sur la compliance.

Comme σ réalise le minimum de l'énergie complémentaire, la compliance s'exprime également sous la forme (cf. [4])

$$c(\omega) = \min_{\substack{div\sigma=0 \ dans \ \omega \\ \sigma.\mathbf{n}=f \ sur \ \partial\Omega \\ \sigma.\mathbf{n}=0 \ sur \ \partial\omega\backslash\partial\Omega}} \int_\omega A^{-1}\sigma.\sigma \tag{2.2}$$

Formulation homogénéisée

En l'absence de contraintes supplémentaires sur les formes admissibles ω le problème (2.1) se révèle être théoriquement mal posé (voir les travaux de Murat et Tartar [123] [168]), car l'existence et l'unicité de la solution sont fausses en général. Ainsi il n'existe pas de forme optimale. Pour illustrer la raison physique de ce phénomène générique de non-existence, supposons qu'on a une forme qui respecte la contrainte sur le volume de matière. En général, une forme avec le même volume et une performance meilleure peut être obtenue en augmentant le nombre de trous (faire beaucoup de très petits trous plutôt que quelques grands trous). Nous pouvons continuer à améliorer la performance en produisant de nouvelles formes avec plus de trous. Par conséquent, atteindre le minimum peut faire appel à un processus de passage à la limite (lorsque les trous deviennent de plus en plus petits et de plus en plus nombreux) conduisant à une forme "généralisée" (ou homogénéisée) qui est un matériau composite obtenu par micro-perforation du matériau élastique d'origine.

L'introduction de matériaux composites comme conception généralisée est un processus de relaxation du problème de conception qui a été étudié par de nombreux groupes de recherches. Le concept du processus de relaxation est simple: on récrit le problème de conception sous une forme qui autorise les matériaux composites obtenus par micro-perforation, pour construire la forme optimale.

Une forme ou structure composite est décrite en chaque point par deux fonctions: $\theta(x)$, sa densité volumique locale de matériau prenant ses valeurs entre 0 et 1, et $A^*(x)$, son tenseur d'élasticité effectif correspondant à sa micro-structure. En utilisant la théorie d'homogénéisation (voir par exemple [123] pour plus de détails sur cette théorie) la formulation homogénéisée relaxée du problème (2.1) est donnée par

$$\min_{\substack{\theta\in[0,1] \\ A^*\in G_\theta}} \left\{ \tilde{J}(\theta,A^*) = \tilde{c}(\theta,A^*) + l\int_\Omega \theta(x) \right. \tag{2.3}$$

où G_θ est l'ensemble des lois de Hooke que l'on peut construire par homogénéisation en mélangeant du matériau A et du vide en proportions respectives θ et $1-\theta$ et $\tilde{c}$ est la compliance homogénéisée définie par

$$\tilde{c}(\theta,A^*) = \int_\Omega A^*\varepsilon(u).\varepsilon(u) = \int_\Omega A^{*-1}\sigma.\sigma$$

et u est maintenant la solution du problème homogénéisé suivant

$$\begin{cases} \sigma & = & A^*\varepsilon(u) & \\ div\sigma & = & 0 & dans\ \omega \\ u & = & 0 & sur\ \partial\Omega_D \\ \sigma.\mathbf{n} & = & f & sur\ \partial\Omega_F \\ \sigma.\mathbf{n} & = & 0 & sur\ \partial\omega \setminus \partial\Omega \end{cases}$$

Le théorème suivant nous donne l'existence des formes optimales généralisées et leurs relations avec les suites minimisantes de formes classiques. Pour les démonstrations voir [4] pour le cas 2-D, et [2] pour le cas 3-D.

Théorème 2.1

la formulation homogénéisée (2.3) est la relaxation du problème d'optimisation de formes (2.1) au sens où :
(i) Il existe, au moins, une forme optimale composite (θ,A^*) de (2.3).
(ii) Toute suite minimisante de formes classiques ω pour (2.1) converge, au sens de l'homogénéisation, vers un minimisateur (θ,A^*) de (2.3).
(iii) Les valeurs des minima de l'énergie originale et homogénéisée coïncident :

$$\inf_{\omega\subset\Omega} J(\omega) = \min_{\substack{\theta\in[0,1] \\ A^*\in G_\theta}} \tilde{J}(\theta,A^*)$$

L'utilisation de l'homogénéisation a conduit à une nouvelle classe d'algorithmes numériques qui optimisent la répartition de matière sans aucune contrainte topologique. Dans ce contexte, Bendsoe et Kikuchi [18] furent les premiers à exploiter l'homogénéisation pour résoudre numériquement le problème d'optimisation topologique des structures élastiques. La méthode qu'ils ont proposée pour résoudre le problème et qui est connue sous le nom de méthode d'homogénéisation se fonde sur les idées de distribution optimale de matière et de relaxation et sur l'introduction d'une micro-structure poreuse dont ils déterminent les propriétés macroscopiques en fonction de la densité grâce à la théorie de l'homogénéisation des micro-structures périodiques. La première étape de la procédure consiste à choisir une

micro-structure et à décrire ses propriétés en fonction de ses paramètres macroscopiques; la seconde étape consiste à déterminer leurs valeurs optimales afin de connaître la distribution de matière au sein du domaine de conception. Depuis, plusieurs auteurs ont contribué à l'application et l'amélioration de la méthode d'homogénéisation (cf. [2], [4], [19, 33, 92, 34, 161, 162] et [166]), et aussi à l'extension de la méthode à d'autres types de structures ainsi qu'à d'autres types d'analyse et de conditions de conception où de nouvelles difficultés apparaissent. Parmi les problèmes résolus à l'heure actuelle, on peut citer la conception de structures soumises à plusieurs conditions de chargement [1], le problème d'optimisation topologique pour les structures en vibrations libres, par exemple l'optimisation de fréquence propre d'une structure élastique ou bien la relocalisation du comportement fréquentiel afin d'exclure certaines bandes de fréquence (cf. [9] et [47]).

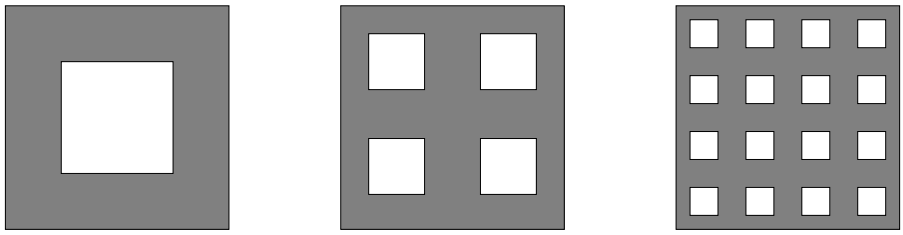
2.4.2 Méthode de contrainte sur le périmètre

Nous introduisons dans cette section une autre approche, appelée méthode de contrainte sur le périmètre, pour régulariser le problème fondamental de la topologie (minimum de la compliance pour un volume de matière donnée : problème (2.1)), et assurer l'existence de la solution, mais pas l'unicité. Contrairement aux méthodes d'homogénéisation, qui consistent à étendre l'espace des solutions en intégrant les solutions comportant des perforations microscopiques, la méthode de périmètre consiste à exclure de telles solutions. Ceci est réalisé en considérant une restriction supplémentaire portant sur le périmètre de la distribution de matière ([73] et [92]), c'est à dire qu'une contrainte supplémentaire est ajoutée au problème afin de s'assurer que le périmètre ne dépassera pas une borne supérieure donnée. L'introduction d'une telle contrainte aboutit à un problème d'optimisation bien posé, et assure également l'existence d'une solution (voir Ambrosio et Buttazo [5] pour la démonstration de ce résultat).

Présentation de la méthode

Le but de cette méthode est d'imposer une borne supérieure sur le périmètre. Ceci est défini comme la mesure du bord de la région pleine Ω_p et est noté $|\partial\Omega_p|$, c'est à dire la longueur totale des foncières de Ω_p. Afin de bien comprendre pourquoi une limite sur le périmètre empêche l'apparition de solutions avec des perforations microscopiques on peut se reporter à la figure 2.3. Cette figure présente trois solutions avec le même rapport vide-solide. Les perforations en nombre croissant présentent la même surface totale. Le périmètre total de la solution augmente avec le nombre de trous et, de manière équivalente, il décroît avec leur taille. cet exemple élémentaire montre que les solutions avec une faible valeur du périmètre possèdent des trous de grande taille tandis que des distributions de densité introduisent un grand périmètre. Si la distribution de matière donne lieu à l'apparition

d'une micro-structure au niveau du maillage macroscopique, le périmètre total augmente. A la limite, lorsque la taille des perforations devient de plus en plus fine, le périmètre de la solution devient infini. Il est alors clair qu'une borne finie sur le périmètre de la solution exclut les structures contenant une infinité de perforations microscopiques. En outre, la valeur du périmètre maximal autorise un contrôle sur le nombre et la taille des trous présents dans la topologie optimale.



Périmètre = 4 Périmètre = 8 Périmètre = 16

FIG. 2.3 – *Evolution du périmètre en fonction du nombre de perforations pour un volume de matière constant* $V = 4$.

Formulation du problème

Considérons le problème de minimisation de la compliance pour un volume de matière donné $\overline{V}$. En introduisant la fonction caractéristique χ définie par

$$\chi : \Omega \longrightarrow \{0,1\} \; avec \; \chi(x) = \begin{cases} 1 & pour \; x \in \Omega_p \; (partie \; pleine) \\ 0 & pour \; x \in \Omega \setminus \Omega_p \; (partie \; vide) \end{cases}$$

Le volume est donné par

$$V = \int_\Omega \chi(x) dx \; ,$$

et lorsque on adjoint la contrainte sur le périmètre, on obtient la formulation suivante du problème

$$\begin{array}{ll} \inf & c(\Omega) \\ \chi \in V_\chi & \\ V \leqslant \overline{V} & \\ |\partial\Omega_p| \leqslant \overline{P} & \end{array} \tag{2.4}$$

où $\overline{P}$ désigne la borne maximale admissible pour la conception, $c(\Omega)$ est la compliance de la structure, et V_χ est donné par

$$V_\chi = \{\chi : \chi(x) = 0 \; ou \; 1 \forall x \in \Omega\}$$

Une preuve d'existence a été donnée dans [5] pour un problème similaire. Plusieurs approches ont été proposées pour la recherche d'une approximation de la solution optimale (voir par exemple [17], [62] et [73]).

2.4.3 Méthode de sensibilité topologique

La première définition du gradient topologique (ou sensibilité topologique) à été introduite par Schumacher [156] pour le problème de minimisation de la compliance. Le problème consiste à minimiser la fonction coût $j(\Omega) = j(\Omega,u_\Omega)$, où le déplacement solution u_Ω est définit sur un domaine borné Ω de $\mathbb{R}^n$, $n = 2, 3$. Soit Ω_ρ le domaine Ω privé d'un petit trou sphérique de rayon ρ et de centre x_0. Un développement asymptotique de la fonctionnelle j, relativement à ρ, peut être donné sous la forme suivante

$$j(\Omega_\rho) = j(\Omega) + f(\rho)g(x_0) + o(f(\rho))$$

où $\lim_{\rho\to 0} f(\rho) = 0$ et $f(\rho) > 0$.

Le but de l'approche est de déterminer la fonction g, qui pourra être utilisée comme une direction de descente pour le processus d'optimisation. Plus précisément, cette méthode permet de localiser le lieu géométrique de la structure élastique où peut apparaître un trou.
En 1997, Sokolowski et Zochowski ont donné des justificatifs mathématiques de cette méthode, et ils l'ont généralisée pour d'autres fonctions coût. Récemment, Masmoudi [113] a étendu la définition du gradient topologique, en adaptant les techniques de la méthode d'état adjoint [29] et de celle de troncature de domaine, au cas où des conditions de Dirichlet homogènes sont imposées sur la frontière d'un trou sphérique. Elle est applicable uniquement dans le cadre de l'élasticité linéaire.

Nous présentons dans ce qui suit les résultats obtenus par J. Céa, S. Garreau, M. Masmoudi et P. Guillaume [30][59]

Construction du gradient topologique

On introduit un trou dans le domaine Ω de la façon suivante. Soit $x_0 \in \Omega$ et un réel $\rho > 0$ suffisamment petit pour que la boule $B(x_0,\rho)$ soit incluse dans Ω. Le domaine perforé Ω_ρ est alors

$$\Omega_\rho = \Omega \setminus \overline{B(x_0,\rho)},$$

et le bord de la boule est noté Γ_ρ.

Une méthode d'état adjoint généralisée

Soit ν un espace de Hilbert fixé. Pour $\rho \geqslant 0$, soit $a_\rho(.,.)$ une forme bilinéaire, uniformément continue et elliptique sur ν, c'est à dire il existe deux constantes strictement positives α et C, indépendantes de ρ, telles que pour tout $\rho \geqslant 0$

$$\begin{aligned} a_\rho(u,v) &\leqslant C\|u\|_\nu\|v\|_\nu \quad \forall u,\, v \in \nu \\ a_\rho(u,u) &\geqslant \alpha\|u\|_\nu^2 \qquad\quad \forall u \in \nu \end{aligned}$$

Supposons qu'il existe une forme bilinéaire et continue δ_a et une fonction réelle $f(\rho) > 0$ définie sur $\mathbb{R}_+$ telles que $\lim_{\rho\to 0} f(\rho) = 0$ et

$$\|a_\rho - a_0 - f(\rho)\delta_a\|_{L(\nu)} = o(f(\rho)). \tag{2.5}$$

où $L(\nu)$ désigne l'espace des formes bilinéaires continues sur ν.

Soit l une forme linéaire et continue sur ν. Pour simplifier, l est supposée indépendante de ρ. Pour $\rho \geqslant 0$, soit u_ρ la solution du problème sur ν

$$a_\rho(u_\rho,v) = l(v), \quad \forall v \in \nu. \tag{2.6}$$

Considérons maintenant une fonction coût $j(\rho) = J(u_\rho)$. Pour simplifier, la fonctionnelle continûment différentiable J est supposée indépendante de ρ. On note $DJ(u)$ la dérivée de J par rapport à u. Le théorème suivant nous donne le développement asymptotique de $j(\rho)$

Théorème 2.2

Si l'hypothèse (2.5) est satisfaite, alors

$$j(\rho) - j(0) = f(\rho)\delta_a(u_0,p_0) + o(f(\rho))$$

où u_0 est la solution de l'équation (2.6) avec $\rho = 0$ et p_0 est la solution du problème adjoint :

$$a_0(\omega,p_0) = -DJ(u_0)\omega\ , \quad \forall \omega \in \nu.$$

La solution u_{Ω_ρ} est définie sur le domaine Ω_ρ et par conséquent, elle appartient à un espace fonctionnel qui dépend de ρ. Le cadre mathématique du théorème 2.2 nécessite

que l'espace fonctionnel soit fixe. En optimisation géométrique, cette nécessité peut être satisfaite en utilisant une technique paramétrique, voir [29] et [71] : elle utilise une transformation bi-lipschitzienne entre un domaine de référence et le domaine réel. En optimisation topologique, une telle transformation n'existe pas entre Ω et Ω_ρ. Cependant, un espace fonctionnel indépendant de ρ va être construit par une méthode de troncature de domaine.

Troncature du domaine

Le but de cette méthode est définir :

- un espace de Hilbert ν_R indépendant de ρ
- une forme bilinéaire $a_\rho(.,.)$, continue et elliptique sur ν_R
- une forme linéaire et continue $l(.)$

telle que la solution u_ρ de l'équation

$$a_\rho(u_\rho,v) = l(v), \quad \forall v \in \nu_R$$

soit égale à la restriction de u_{Ω_ρ} à Ω_R, un sous-domaine de Ω (cf. figure 2.4).

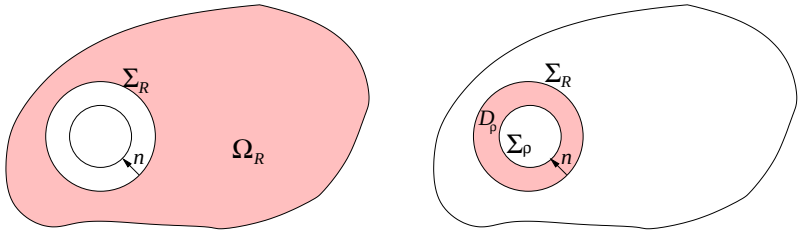


FIG. 2.4 – *Le domaine tronqué*

Le déplacement u_ρ sur le domaine initial Ω est solution de

$$\begin{cases} -div\sigma(u_\Omega) &= 0 \quad dans\ \Omega \\ \sigma(u_\Omega).\mathbf{n} &= f \quad sur\ \partial\Omega_F \\ u_\Omega &= 0 \quad sur\ \partial\Omega_D \end{cases} \tag{2.7}$$

$\partial\Omega_F$ désigne la partie de la frontière où s'exerce une force f et $\partial\Omega_D$ est la partie où le domaine est fixé.

Maintenant, considérons le domaine perforé Ω_ρ. Le déplacement u_{Ω_ρ} est solution de

$$\left\{\begin{array}{lcll} -div\sigma(u_{\Omega_\rho}) & = & 0 & dans\ \Omega_\rho \\ \sigma(u_{\Omega_\rho}).\mathbf{n} & = & f & sur\ \partial\Omega_F \\ u_{\Omega_\rho} & = & 0 & sur\ \partial\Omega_D \\ \sigma(u_{\Omega_\rho}).\mathbf{n} & = & 0 & sur\ \Sigma_\rho \end{array}\right. \tag{2.8}$$

Soit $R > \rho$ tel que la boule $B(x_0,R)$ soit incluse dans Ω. La frontière de $B(x_0,R)$ est notée Σ_R. Le domaine tronqué $\Omega \setminus B(x_0,R)$ est noté Ω_R et D_ρ désigne le "disque" $B(x_0,R) \setminus B(x_0,\rho)$ (cf. figure 2.4).
Le déplacement u_ρ est défini pour $\rho \geqslant 0$ comme la solution du problème tronqué

$$\left\{\begin{array}{lcll} -div\sigma(u_\rho) & = & 0 & dans\ \Omega_R \\ \sigma(u_\rho).\mathbf{n} & = & f & sur\ \partial\Omega_F \\ u_\rho & = & 0 & sur\ \partial\Omega_D \\ \sigma(u_\rho).\mathbf{n} & = & T_\rho u_\rho & sur\ \Sigma_R \end{array}\right. \tag{2.9}$$

avec T_ρ est l'opérateur pseudo-différentiable défini par

$$\begin{array}{rcl} T_\rho : H^{1/2}(\Sigma_R)^3 & \to & H^{-1/2}(\Sigma_R)^3 \\ \psi & \mapsto & T_\rho\psi = \sigma(u_\rho^\psi)n \end{array}$$

et, pour $\psi \in H^{1/2}(\Sigma_R)^3$ et $\rho > 0$, u_ρ^ψ est la solution du problème

$$\left\{\begin{array}{lcll} -div\sigma(u_\rho^\psi) & = & 0 & dans\ D_\rho \\ u_\rho^\psi & = & \psi & sur\ \Sigma_R \\ \sigma(u_\rho^\psi).\mathbf{n} & = & 0 & sur\ \Sigma_\rho \end{array}\right.$$

Nous avons alors le résultat classique suivant :

Proposition 2.1

Les problèmes (2.8) (respectivement (2.7)) pour $\rho = 0$ et (2.9) ont une unique solution. De plus, la restriction à Ω_R de la solution u_{Ω_ρ} (respectivement u_Ω) à (2.8) (respectivement (2.7)) est solution de (2.9).

En utilisant la formulation variationnelle associée au problème (2.9), dans un cadre mathématique adéquat, et les résultats sur les équations intégrales on obtient le résultat suivant :

Théorème 2.3

Le développement asymptotique de la forme bilinéaire a_ρ relativement à ρ est

$$\|a_\rho - a_0 - f(\rho)\delta_a\|_{L(\nu)} = o(f(\rho))$$

et

$$j(\rho) - j(0) = f(\rho)\delta_a(u_0,p_0) + o(f(\rho))$$

$\delta_a(u_0,p_0)$, dont l'expression est donnée dans [30], est appelé le gradient topologique et noté $g(x_0)$. On peut noter qu'il ne dépend pas de R et ne nécessite que deux calculs éléments finis pour être évalué.

L'algorithme numérique

Soit un domaine initial Ω_0 et une suite décroissante de contraintes de volume $(m_k)_{k\leqslant 0}$ telle que $m_0 = aire(\Omega_0)$. A une itération k donnée, le gradient topologique est noté $g_k(x)$ et c_{k+1} est choisi de telle sorte que

$$\begin{cases} \Omega_{k+1} & = \{x \in \Omega_k,\ g_k(x) \leqslant c_{k+1}\} \\ aire(\Omega_{k+1}) & = m_{k+1} \end{cases}$$

L'algorithme se déroule comme suit,

Algorithme :

- Un domaine Ω_0 est initialisé , $k = 0$
- Le processus suivant est itéré jusqu'à convergence
  - Résoudre le problème d'élasticité dans Ω_k
  - Calculer le gradient topologique g_k
  - Mettre à jour le domaine $\Omega_{k+1} = \{x \in \Omega_k,\ g_k(x) \leqslant c_{k+1}\}$ où c_{k+1} est choisi tel que $aire(\Omega_{k+1}) = m_{k+1}$
  - $k \leftarrow k + 1$

Le gradient topologique est calculé sur chaque élément et ceux pour lesquels il est le plus faible sont supprimés. Le nombre d'éléments supprimés est calculé de façon à supprimer un pourcentage du volume de matière à chaque étape.

2.4.4 Méthode des lignes de niveaux

Récemment proposée par Allaire et al. [3], la méthode des lignes de niveaux pour l'optimisation de formes consiste à combiner, autant que faire se peut, les avantages des méthodes

de variation de frontière et d'homogénéisation. Suivant une idée de la méthode d'homogénéisation, cette approche utilise un maillage fixe qui contient à la fois la forme et les trous (ou le vide) représentés par un matériau très faible. Le bord de la forme est paramétré par une fonction ligne de niveaux suivant le formalisme d'Osher et Sethian [127][160]. L'optimisation de formes consiste à transporter la fonction ligne de niveaux (c'est-à-dire le bord de la forme) avec une vitesse qui fasse décroître la fonction objectif. Le calcul de cette vitesse se fait par la méthode de variation de frontière, en dérivant la fonction objectif par rapport à la forme.

Discussion

Nous avons présenté dans cette section les différentes méthodes déterministes pour l'optimisation topologique de formes. Toute fois, ces approches sont principalement restreintes à l'élasticité linéarisée ; elles ne permettent de trouver qu'une seule solution (optimum local) ; et elles sont incapables de traiter les problèmes d'optimisation topologique multi-critères (par exemple optimiser à la fois la rigidité et les fréquences propres d'une structure, voir chapitre 6) autrement qu'en se ramenant à un seul objectif par agrégation des différents critères. Une approche possible pour dépasser ces limites est offerte par les méthodes d'optimisation stochastiques, comme les algorithmes évolutionnaires [153].

2.5 Optimisation topologique de formes : méthodes stochastiques

Les Méthodes d'optimisation stochastiques ont été appliquées avec succès dans les travaux antérieurs en mécanique des structures dont beaucoup ont concerné l'optimisation d'assemblage de barres. Lin et Hajela ([109] et [155]) et Schoenauer et Wu [155] ont résolus des problèmes de dimensionnement des sections des barres; Grierson et Pak [70] ont abordé le problème de l'optimisation topologique de treillis de barres. Des résultats intéressants ont également été obtenus pour l'optimisation des empilements de matériaux composites par Leriche et Haftka [107]. Dans le domaine de l'optimisation topologique de formes, les premiers travaux utilisant les algorithmes évolutionnaires sont dus à Jensen [91] et à Chapman, Saitou et Jakiela [27], et plus récemment à Kane et Schoenauer [99]. Citons enfin dans ce domaine Anagnostou, Ronquist et Patera [6], qui ont appliqué l'approche fondée sur le recuit simulé (cf. paragraphe 2.5.1 pour le principe de cette méthode) à l'optimisation de la section droite d'une barre en maximisant son moment d'inertie.

2.5.1 Le recuit simulé

Dans le début des années 80, Kirkpatrick, Gelatt et Vecchi [102] ont introduit les concepts du recuit pour l'optimisation combinatoire. Ces concepts sont inspirés d'une méthode employée en physique pour obtenir des corps dans leur état fondamental : la première étape de la procédure consiste à porter le solide à très haute température jusqu'à fusion (de ce dernier) ; la seconde consiste à le refroidir suivant un schéma de décroissance de température bien particulier afin d'atteindre un état solide d'énergie minimale. Celui-ci ne peut être obtenu que si la température initiale est suffisamment élevée et le refroidissement suffisamment lent. Sinon, on aboutira à un état méta stable à énergie non minimale (l'inverse du processus de recuit est la trempe qui consiste à refroidir brutalement un solide). Cette technique, qui peut être considérée comme une amélioration des méthodes d'optimisation locale, a l'avantage de ne pas s'arrêter au premier optimum local rencontré.

Pour simuler le processus de recuit, on introduit un paramètre de température qui est ajusté pendant la recherche. Si la température est élevée, alors la probabilité de transiter vers un niveau d'énergie plus élevée est importante. Si la température est basse alors les transitions à faible augmentation d'énergie sont plus probables. Initialement la température est haute mais elle diminue avec le temps.

Soit U la fonction à minimiser, les étapes du recuit simulé sont les suivantes :

1. Fixer une température initiale élevée, T.
2. Générer un point X.
3. Générer un point X' par une opération aléatoire sur X.
4. Si $y' = U(X')$ est meilleur que $y = U(X)$, alors $X = X'$. Sinon remplacer X par X' avec une probabilité P_a (loi de Boltzmann) dépendant de T et de l'écart d'énergie $\Delta E = y' - y$.
$$P_a = exp(\frac{\Delta E}{k_b T}), \tag{2.10}$$
avec k_b est la constante de Boltzmann.
5. Répéter 3 et 4 jusqu'à l'obtention de l'équilibre thermique. En pratique ce nombre d'itérations dépend de la température (équation 2.10).
6. Réduire la température selon le schéma suivant $T_{i+1} = T_i * \alpha$, avec $0 < \alpha < 1$.
7. Tant qu'un critère d'arrêt (dépendant de la température) n'est pas satisfait, retourner en 3.

Pour faire du recuit simulé, il faut une densité de probabilité de génération $g(X)$ per-

mettant de se déplacer dans l'espace d'état $\Omega = X^i : i = 1,D$, une densité de probabilité d'acceptation $h(X)$ d'un nouvel état après une transition et une loi de décroissance de la température $T(k)$ permettant de réduire progressivement les fluctuations des deux densités précédentes ($T(k)$ est homogène à la fonction objectif).

Le schéma de décroissance de la température est généralement lent et doit être ajusté à l'aide de la chaleur spécifique $C(T)$:

$$C(T) = \frac{d < E(T) >}{dT} = \frac{< E(T)^2 > - < E(T) >^2}{k_b T^2}$$

Une grande valeur de $C(T)$ indique que le système passe à l'état solide et donc que la décroissance de la température doit être plus lente. Pour la plage de décroissance rapide de la température, il est nécessaire de ménager un nombre important de transitions de Métropolis [116] afin de permettre au système d'atteindre son équilibre thermique. Par contre, dans la plage de décroissance de température faible (après augmentation de la chaleur spécifique), ce nombre d'itérations de Métropolis peut être réduit.

2.5.2 Les algorithmes évolutionnaires

Le point le plus crucial dans la construction d'un algorithme évolutionnaire (AE) est le choix de la représentation. Tous les travaux cités à l'introduction de la section 2.5 montrant des applications des algorithmes évolutionnaires à des problèmes d'optimisation topologique de formes utilisent la même représentation binaire "naturelle", appelée bitarray dans [99] : elle est associée à un maillage particulier du domaine – celui qui est utilisé pour calculer le comportement mécanique de la structure et déterminer la performance (cf. Chapitre 3). A chaque élément du maillage on attribue une valeur 1 si il contient de la matière, et 0 sinon. La figure 2.5 donne, pour un maillage 13×6, le tableau de bits et la forme correspondante. Notons que cette représentation binaire n'est pas équivalente à la représentation "bitstring" habituelle (cf. chapitre 1.2) : un opérateur de croisement spécifique, utilisant la géométrie du problème, a été développé [98] qui est similaire à celui décrit dans le chapitre 4 pour la représentation à base de diagrammes de Voronoï.

En 1996, cette approche a permis de lever quelques limitations des méthodes déterministes, cf. les travaux de C. Kane [97]:

- Obtention de plusieurs solutions de topologies différentes, solutions quasi-optimales du problème posé (cf figure 2.6), entre lesquelles l'expert pourra choisir selon des critères non modélisables.

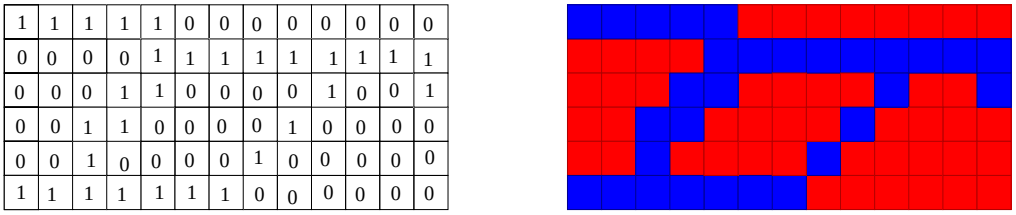


FIG. 2.5 – *Représentation d'une forme par tableau de bits "bitarray".*

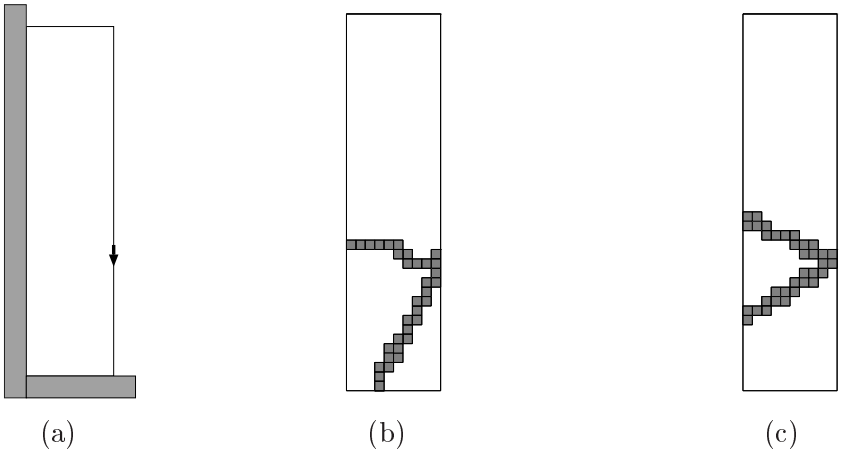


FIG. 2.6 – *Deux solutions optimales de topologies différentes (b) et (c) pour le problème de la plaque console de dimension* 1×4 *dont les contours gauche et bas sont fixes (a). Solutions obtenues par K. Kane [97].*

- Prise en compte des chargements multiples, éventuellement de natures différentes (forces, déplacements imposés, etc). Une simple modification de la fonction à optimiser est suffisante, l'algorithme restant par ailleurs inchangé.
- Résolution du problème avec chargement sur la frontière inconnue. L'exemple-type est le cas du dôme sous-marin, sur la surface supérieure duquel est appliquée une pression uniforme.
- Utilisation de tout type de modèle de comportement. Puisque seul un solveur du problème direct est requis, il est possible de traiter le problème de l'optimisation topologique de formes pour tout comportement dont on possède une simulation numérique robuste. Des résultats dans le cadre de l'élasticité en grands déplacements sont venus illustrer cette souplesse de l'approche évolutionnaire.

Malgré son succès dans la résolution de problèmes d'optimisation topologique de formes, voir [99], [97] et [100], la représentation "bitarray" souffre d'une profonde limitation liée à la dépendance de la complexité de l'algorithme par rapport à celle du maillage associé. En effet, la taille d'un individu (le nombre de bits nécessaire pour décrire un individu) est égale à la taille du maillage. Malheureusement, les résultats théoriques [32] comme les constatations empiriques [66] indiquent que la taille critique de population nécessaire pour atteindre la convergence augmente au moins linéairement avec la taille de chaque individu. De plus, des populations plus grandes nécessitent en général un plus grand nombre de générations pour converger. Il est donc clair que cette approche doit restreindre son domaine d'application à des maillages grossiers bidimensionnels, alors que les ingénieurs ont besoin de fins maillages tridimensionnels!

Ces considérations conduisent à la recherche de représentations plus compactes, dont la complexité ne dépend pas de celle de la discrétisation. Pour obtenir une représentation indépendante de la complexité une ultime étape consiste à la faire évoluer elle-même en l'ajustant par algorithmes évolutionnaires. C'est le but de cette thèse.

Toutefois, avant de passer en revue plusieurs propositions de représentations (chapitres 4 et 5) nous allons décrire en détail la fonction que nous optimiserons en particulier au niveau de la prise en compte des contraintes, sur un problème simple d'optimisation de formes.

Chapitre 3

La fonction performance

Sommaire

3.1 Introduction

Nous nous intéressons dans ce chapitre à la définition de la fonction performance (ou fonction objectif), utilisée dans cette thèse, pour le problème de l'optimisation topologique de formes.

Divers critères ont été utilisés dans les travaux antérieurs en optimisation de forme. La fonction objectif des méthodes déterministes, comme par exemple la méthode d'homogénéisation, est basée sur la compliance, définie par le travail des forces extérieures (cf. [4]) et section 2.4.1. Bien qu'elle ait l'avantage d'être différentiable, elle ne permet pas un contrôle précis du comportement mécanique de la solution (par exemple du déplacement maximal de la structure, qui est généralement une donnée importante du cahier des charges). Cependant, elle a été utilisée comme fonction objectif par les algorithmes évolutionnaires [97] afin de comparer l'approche évolutionnaire à la méthode d'homogénéisation. La compliance ne sera pas considérée ici.

Une autre approche, utilisée dans [27], consiste à maximiser la raideur de la structure en utilisant une définition basée sur le rapport entre la raideur, qui est inversement proportionnelle au déplacement maximal sous le chargement imposé, et le poids de la structure. Mais là encore, cette fonction ne permet pas un contrôle précis des déplacements ou des contraintes mécaniques [97]. De plus, des expériences comparatives [97] ont montré que l'utilisation de la raideur comme fonction performance conduisait à des structures plus lourdes sans amélioration notable du comportement mécanique.

Nous avons choisi de considérer le problème, comme dans [97], sous l'angle de la minimisation du poids d'une structure, définie dans un domaine de référence, en respectant un déplacement maximal limite pour chaque cas de charge considéré. Le choix de cette contrainte est justifié par le fait qu'elle est la plus utilisée dans l'industrie : un cahier des charges définissant une structure mécanique requiert le plus souvent que cette structure, sous un chargement donné, ait un déplacement maximal ne dépassant pas une valeur im-

posée.

Pour prendre en compte les contraintes dans les algorithmes évolutionnaires, plusieurs méthodes ont été mises au point. Toutefois, la méthode la plus générale reste la méthode de pénalisation et c'est aussi la plus simple à mettre en œuvre : elle ne demande que la modification de la fonction objectif en introduisant des termes de pénalisation. Cependant, la mise au point de ces termes influe sur l'efficacité de l'optimisation évolutionnaire. Plusieurs stratégies ont été proposées dans la littérature [93, 118, 20]. Nous introduisons ici deux nouvelles méthodes pour définir la fonction de pénalité. Ces deux approches sont basées sur une pénalité adaptative ajustée automatiquement selon l'état courant de la population afin d'assurer l'existence d'un taux d'individus faisable et infaisable dans la population. Cela mène à une exploration efficace du voisinage de la frontière du domaine faisable.

Ce chapitre est organisé de la façon suivante : la section 3.2 rappelle le cadre de travail théorique défini par Ghaddar et al. [62], dont le problème consiste à étudier la résistance d'une barre de section S. En section 3.3, nous introduisons brièvement le problème mécanique pour l'optimisation topologique de formes. Nous construisons ensuite, dans la section 3.4, la fonction performance utilisée par la suite. En section 3.5, nous présentons brièvement les différentes techniques de pénalisation pour les algorithmes évolutionnaires, et nous introduisons les deux nouvelles approches adaptatives proposées ici. Une étude expérimentale comparative, donnée dans la section 3.6, montre, pour le problème d'optimisation topologique de formes, la supériorité de la stratégie adaptative par rapport aux autres méthodes - statique et dynamique.

3.2 Optimisation du moment d'inertie

Nous présentons dans cette section le problème de l'optimisation de la section droite d'une barre, posé par Ghaddar et al. [62], ainsi que les résultats théoriques obtenus.

3.2.1 Position du problème

Soit une barre de longueur l (suivant z) et S sa section en (x,y). Le problème consiste à trouver la forme optimale de la section S de la barre soumise à des efforts de flexion ; ce qui revient à minimiser le poids (il est supposé proportionnel au volume de la barre), tout en maximisant le moment d'inertie de la barre I_{yy}, donné par :

$$I_{yy} = \int_S (y - y_G)^2 dxdy$$

où y_G est l'ordonnée du centre de gravité G de S, définie par :

$$y_G = \frac{\int_S y dx dy}{\int_S dx dy}$$

Par ailleurs, il faut respecter les contraintes suivantes :

- la section S doit être connexe et incluse dans un domaine $\Omega \subset \mathbb{R}^2$ donné.
- l'épaisseur de la section doit être supérieure à une épaisseur donnée, pour des raisons de faisabilité technologique.

3.2.2 Formulation mathématique

La fonction coût s'écrit de la manière suivante [62] :

$$\begin{aligned} C(S) &= \alpha_0 l Aire(S) \\ &+ \alpha_1 l Perimetre(S) + \alpha_2 V(S) \\ &+ \alpha_3 f(I_{yy}/I_0) \end{aligned} \tag{3.1}$$

où $Aire(S)$ est l'aire de la section S, $Perimetre(S)$ le périmètre de S, $V(S)$ le nombre de coins que peut avoir la pièce et f une application de $[0,\infty]$, décroissante, et qui tend vers 0 quand son argument tend vers ∞. Les α_i, $i = 0..3$, sont des constantes positives, I_0 est une valeur minimale donnée pour le moment I_{yy} et l est la longueur de la barre.

Pour tenir compte des contraintes technologiques (épaisseur minimum, nombre de coins) et numériques (discrétisation du domaine de travail) Ghaddar et al. [62] ont introduit de nouveaux espaces que nous allons rappeler maintenant.

On commence par introduire l'espace X_M^1 : espace des sous-domaines fermés S de $\mathbb{R}$ tels que

$$S \subset [0,L] \times [0,L],$$

et

$$\overline{\mathring{S}} = S$$

On suppose que la frontière ∂S, de $S \in X_M^1$, est composée de K courbes polygonales γ^k, $1 \leqslant k \leqslant K$. Chaque courbe fermée γ^k est définie par une suite ordonnée de m^k sommets distincts $\mathbf{a}_i^k = (a_i^k\ ,\ b_i^k)$, $1 \leqslant i \leqslant m^k$, vérifiant la condition de rectilinéarité

$$(b_{i+1}^k - b_i^k)(a_{i+1}^k - a_i^k) = 0,$$

et la condition des "vrais" coins

$$(b^k_{i+1} - b^k_{i-1})(a^k_{i+1} - a^k_{i-1}) \neq 0.$$

On suppose aussi que toute courbe $\gamma^k = \bigcup[\mathbf{a}^k_i\ ,\ \mathbf{a}^k_{i+1}]$, $1 \leqslant k \leqslant K$, est une courbe simple, qui, en plus de la condition des sommets distincts, vérifie la condition suivante

$$(\mathbf{a}^k_i,\mathbf{a}^k_{i+1}) \bigcap (\mathbf{a}^k_j,\mathbf{a}^k_{j+1}) = \emptyset,\ \ i \neq j\ ;$$

les γ^k sont également d'intersection vide

$$\gamma^k \bigcap \gamma^l = \emptyset,\ k \neq l.$$

Et enfin, on suppose que tout élément de X^1_M possède au plus un total de M coins

$$V(S) = \sum_{k=1}^{K} m^k \leqslant M.$$

Le contour de $S \in X^1_M$ est orienté de façon à ce que la normale unitaire $\mathbf{n}$, définie partout sauf aux coins, soit orientée vers l'intérieur du domaine. En prenant comme valeur de la normale au coin la moyenne des normales des cotés voisins, la normale peut être définie en tout point du contour . Le fait de choisir l'orientation de la normale à l'intérieur du domaine (dans le sens des aiguilles d'une montre ou dans le sens contraire des aiguilles d'une montre) rend le choix de γ^k unique.

Donnons maintenant les définitions d'épaisseur et de coépaisseur d'un élément S de X^1_M. Pour cela, on associe à chaque coté $[\mathbf{a}^k_i\ ,\ \mathbf{a}^k_{i+1}]$ de S une famille de rectangles de largeur τ décrite par les sommets ordonnés $(\mathbf{a}^k_i\ ,\ \mathbf{a}^k_{i+1}\ ,\ \mathbf{b}^k_{i+1}(\tau)\ ,\ \mathbf{b}^k_i(\tau))$, avec

$$\mathbf{b}^k_i(\tau) = \mathbf{a}^k_i + \tau\mathbf{n}(\frac{1}{2}(\mathbf{a}^k_i + \mathbf{a}^k_{i+1})).$$

De même, on associe à chaque coin $\mathbf{a}^k_i$ une famille de carrés de coté τ et de diagonale $[\mathbf{a}^k_i\ ,\ \mathbf{a}^k_i + \sqrt{2}\mathbf{n}]$

Définition 3.1

L'épaisseur $E(S)$ d'un élément S est le maximum de tous les τ tels que tout rectangle ou carré de coté τ soit dans S pour tout k, $1 \leqslant k \leqslant K$, pour tout i, $1 \leqslant i \leqslant m^k$. La coépaisseur $\overline{E}(S)$ est définie comme étant l'épaisseur de la fermeture du complémentaire de S dans $[O,L]^2$.

On introduit maintenant deux nouveaux espaces $X_M^2(t,\overline{t})$ et $X_M^3(t,\overline{t})$ définis pour tout t et $\overline{t}$ positifs par

$$X_M^2(t,\overline{t}) = \{S \in X_M^1 \mid E(S) \geqslant t,\ \overline{E}(S) \geqslant \overline{t},\ (t,\overline{t}) \leqslant t_{max}\}$$

$$X_M^3(t,\overline{t}) = \{S \in X_M^2(t,\overline{t}) \mid S\ connexe\}$$

Les espaces X_M^i, $i \in \{1,2,3\}$, munis de la distance δ définie par

$$\delta(A,B) = Aire(A\Delta B)$$

sont des espaces métriques. L'opérateur symétrique (ou le "OU exclusif"),Δ, est défini par

$$A\Delta B = (A \bigcup B) \setminus (A \bigcap B)$$

Le problème consiste alors à trouver la section $S^* \subset \Omega$ dans $X_M^3(t,\overline{t})$ qui minimise la fonctionnelle C suivante

$$\begin{aligned} C(S) &= \alpha_0 l[\sum_{k=1}^{K}\sum_{i=1}^{m^k}\frac{1}{2}(b_{i+1}^k + b_i^k)(a_{i+1}^k - a_i^k)] \\ &+ \alpha_1 l[\sum_{k=1}^{K}\sum_{i=1}^{m^k}\{(b_{i+1}^k - b_i^k)^2 + (a_{i+1}^k - a_i^k)^2\}^{\frac{1}{2}}] + \alpha_2 \sum_{k=1}^{K} m^k \\ &+ \alpha_3 f\left([\sum_{k=1}^{K}\sum_{i=1}^{m^k}\frac{1}{3}(b_i^k - y_G)^3(a_{i+1}^k - a_i^k)]/I_0\right) \end{aligned} \tag{3.2}$$

où

$$y_G = \frac{1}{Aire(S)}\sum_{k=1}^{K}\sum_{i=1}^{m^k}\frac{(b_i^k)^2}{2}(a_{i+1}^k - a_i^k)$$

est l'ordonnée du centre de gravité de S.

Rappelons maintenant le résultat obtenu dans [62] sur le problème de minimisation.

Théorème 3.1

Il existe au moins une solution S^* dans $\{S \in X_M^3 | S^* \subset \Omega\}$ telle que $C(S^*) \leqslant C(\tilde{S})$ pour tout sous-domaine $\tilde{S}$ dans $\{S \in X_M^3 | S^* \subset \Omega\}$.

Des résultats d'approximation dans le cas du problème discret et estimation d'erreur possible ont été obtenus aussi dans [62].

Remarque

Le résultat du théorème 1.1 a été étendu au cadre de l'élasticité linéaire par C. Kane [97], en utilisant la fonction coût que nous allons présenter et étudier dans la suite.

3.3 Le problème mécanique

Le contexte de cette thèse est l'Optimisation Topologique de Structures (Topological Optimum Design – TOD). Le problème consiste à trouver la forme optimale d'une structure contenue dans un domaine donné (i.e. la répartition de matière dans ce domaine) de telle sorte que le comportement mécanique de cette structure satisfasse certaines contraintes – ici par exemple une borne sur le déplacement maximal sous un chargement donné, mais on pourrait aussi imaginer une borne sur des fréquences propres ou une combinaison de critères mettant en jeu la rigidité et le comportement vibratoire. Le critère à optimiser est le poids de la structure mais on pourrait aussi optimiser d'autres critères : minimisation d'un état de tension, d'un déplacement ; maximisation d'une fréquence fondamentale ou d'une charge critique, conception pour un coût imposé.

3.3.1 Le modèle linéaire

Le modèle mécanique utilisé ici est celui de l'élasticité linéarisée bidimensionnelle en contraintes planes (à l'exception de la section 4.3.12 du chapitre 4 où les résultats sont tridimensionnels), et on ne considérera que des matériaux linéaires et isotropes (cf. e.g. [36]). Toutes les figures présentées par la suite sont adimensionnelles (e.g. le module d'Young vaut toujours 1) et les effets de gravité sont négligés. Le problème d'élasticité peut se formuler comme suit :

Soient Ω un ouvert borné de $\mathbb{R}^n$, $n = 2, 3$, et Ω_v l'ensemble des trous enlevés dans Ω (cf. figure 3.1). Les équations locales d'équilibre mécanique s'écrivent comme suit :

$$\left\{\begin{array}{lcll} \sigma & = & A\varepsilon(u) & \\ div\sigma & = & 0 & dans\ \Omega \setminus \Omega_v \\ \sigma.\mathbf{n} & = & 0 & sur\ \partial\Omega \setminus (\Gamma_f \cup \Gamma_u) \\ \sigma.\mathbf{n} & = & f & sur\ \Gamma_f \\ u & = & 0 & sur\ \Gamma_u \end{array}\right.$$

où

$$\varepsilon(u) = \frac{1}{2}(\nabla u + \nabla^t u)$$

est le tenseur des déformations de Green linéarisé, σ le tenseur des contraintes, A est le tenseur d'élasticité (loi de Hooke), $\mathbf{n}$ la normale sortante sur la frontière, f une force sufacique appliquée sur la frontière Γ_f, et u le champ de déplacement.

Pour une présentation détaillée et complète du modèle théorique nous renvoyons le lecteur à [36], et à [94] pour les modèles numériques.

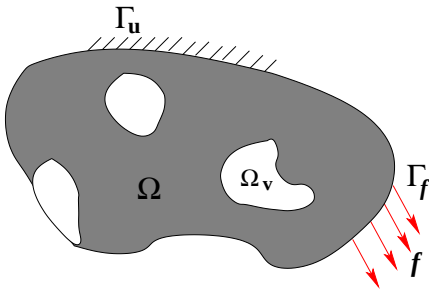


FIG. 3.1 – *Un domaine Ω soumit à des forces et des conditions au bord.*

3.3.2 Le problème test

Un benchmark très populaire pour l'optimum design est celui de la plaque-console (cantilever) : le domaine de calcul est un rectangle ; la plaque est encastrée sur la partie verticale gauche de la frontière (déplacement imposé égal à 0) et le chargement consiste à appliquer une force ponctuelle verticale au milieu de la frontière verticale de droite. La figure 3.2 montre le domaine de calcul pour le cantilever 2×1.

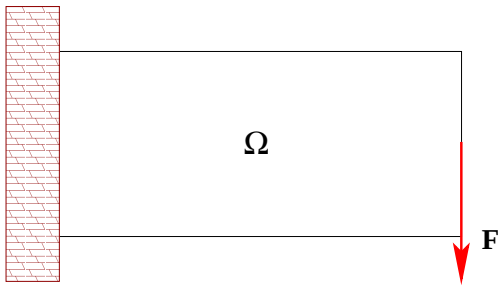


FIG. 3.2 – *Le problème de la plaque-console "cantilever" 2×1.*

3.4 Calcul de la fonction performance

Le problème traité dans ce travail consiste à minimiser le poids d'une structure soumise à un chargement donné (cf. figure 3.2 par exemple), avec une contrainte de type $D^i_{Max} \leqslant D^i_{lim}$ pour chaque cas de charge, où D^i_{Max} est le déplacement maximal calculé par simulation numérique pour le chargement i, et D^i_{lim} sa limite supérieure imposée (donnée par le cahier des charges). Le calcul du déplacement maximal est effectué en utilisant un programme classique d'éléments finis [94]. Le problème se formule alors de la façon

suivante

$$(P) \qquad \begin{cases} minimiser\ poids \\ avec\ D^i_{Max} \leqslant D^i_{lim}, \quad i = 1..n \end{cases}$$

C'est un problème d'optimisation avec contraintes. Pour tenir compte de ces contraintes dans la fonction performance (fonction à optimiser) nous avons adopté l'approche fondée sur la méthode de pénalisation, car c'est la plus simple à mettre en œuvre: elle ne demande que la modification de la fonction d'adaptation (cf. section 3.5). Le problème (P) se transforme ainsi en un problème d'optimisation sans contraintes, où la fonction-coût à minimiser est donnée par (dans le cas d'une seule contrainte, i.e d'un seul cas de chargement)

$$F = poids + \alpha(D_{Max} - D_{lim})^+ \tag{3.3}$$

où α est un paramètre positif (qui peut être vu comme un multiplicateur de Lagrange pour la contrainte associée) ; $(D_{Max} - D_{lim})^+$ mesure le degré de violation de la contrainte

$$(D_{Max} - D_{lim})^+ = max(0, D_{Max} - D_{lim})$$

La figure 3.3 résume les principales étapes du calcul de la fonction performance pour une structure connectée (cf. section 3.4.1) : le comportement mécanique est calculé numériquement par la méthode des éléments finis. Le poids de la structure, pénalisé par les violations des contraintes du cahier des charges déduites de la simulation, est une bonne mesure d'adaptation de la structure au problème.

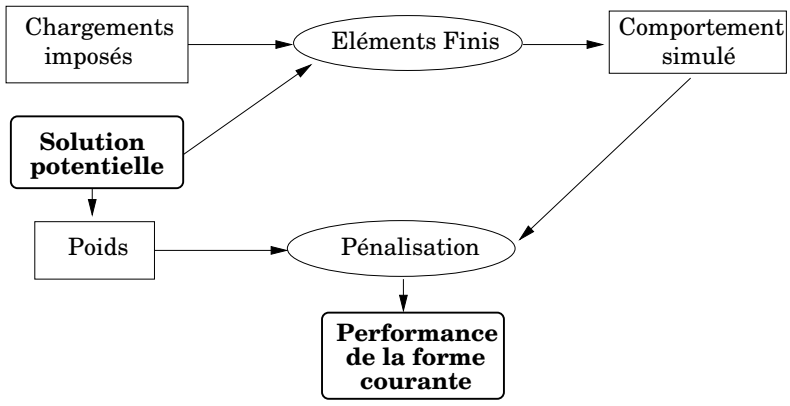


FIG. 3.3 – *Approche évolutionnaire de l'Optimum Design.*

3.4.1 Analyse géométrique

D'un point de vue mécanique, les structures qui ne relient pas le point d'application de la force et la frontière encastrée ne sont pas admissibles et on leur attribue une performance arbitrairement grande - ce qui devrait les éliminer lors de la sélection. De plus, toutes les parties de la structure qui ne sont pas connectées au point d'application de la force (cf. figure 3.4)– et ainsi n'affectent pas le comportement mécanique de la pièce – sont éliminées avant l'analyse par éléments finis mais pénalisent la performance du fait de leur poids additionnel inutile (cf. [99, 97] pour une discussion détaillée de ces sujets). Cette approche favorise la disparition progressive de la matière non connectée et évite la dégénérescence[1] dans la représentation. La fonction performance est donc donnée par

$$F = P_{utile} + \epsilon P_{inutile} + \alpha (D_{Max} - D_{lim})^{+} \tag{3.4}$$

où P_{utile} est la surface de la structure principale (la composante recevant le chargement); $P_{inutile}$ est la surface totale des composantes matérielles non connectées à la structure principale.
Le paramètre ϵ, comme α, est un paramètre de pénalisation positif, mais son importance est moins cruciale. On lui associe une petite valeur.

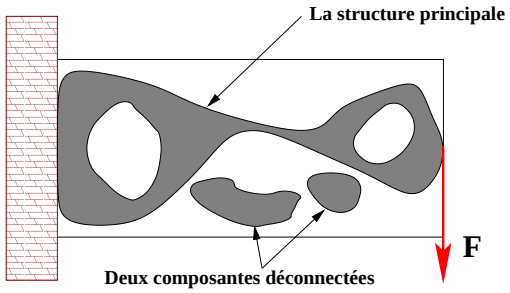


FIG. 3.4 – *Exemple de structure avec des composantes matérielles déconnectées.*

Nous présentons dans ce qui suit brièvement les différentes techniques de pénalisation pour les algorithmes évolutionnaires (cf. [86] [77] pour une description détaillée), et nous proposons deux nouvelles approches.

1. c'est-à-dire plusieurs génotypes correspondent a un même phénotype

3.5 Méthodes de pénalisation

3.5.1 Formulation du problème d'optimisation

Considérons le problème d'optimisation avec contraintes suivant

$$optimiser\ \mathcal{F}(x),\ x = (x_1, \ldots, x_n) \in F \subset \mathbb{R}^n, \tag{3.5}$$

sous les contraintes :

$$\begin{cases} g_j(x) \leqslant 0 & j = 1, \ldots, q \\ h_j(x) = 0 & j = q+1, \ldots, m \end{cases} \tag{3.6}$$

où $\mathcal{F}$, g_i et h_j sont des fonctions réelles dans $\mathbb{R}^n$; et F, appelé domaine faisable, désigne l'espace des solutions admissibles [2] (qui respectent l'ensemble des contraintes (3.6)).

Plusieurs stratégies ont été mises au point pour prendre en compte les contraintes avec les algorithmes évolutionnaires (pour une revue des différentes techniques utilisées dans les algorithmes évolutionnaires, cf. [86] et [77] ainsi que les références que ces articles contiennent). La plupart d'entre elles sont basées sur le concept de fonctions de pénalité, qui pénalisent les individus violant les contraintes dans la population en ajoutant à leur fonction de performance un terme de pénalité qui est nul si les contraintes sont respectées et positif sinon. Autrement dit, la méthode de pénalisation consiste à transformer le problème initial avec contraintes en un problème sans contraintes; le problème (3.5)-(3.6) s'écrit alors :

$$optimiser\ F(x) = \mathcal{F}(x) + p(x)\ , \tag{3.7}$$

où $p(x)$, appelée fonction de pénalité, dépend des violations de contraintes :

$$p(x) = \alpha_k \sum_{j=1}^{m} (v_j(x))^\beta \tag{3.8}$$

où α_k est le paramètre de pénalisation [3], β est un paramètre (souvent égal à 1 ou 2 [118]), et $v_j(x)$ est en général la mesure du degré de violation de la j^{eme} contrainte du problème posé, calculée comme suit :

$$v_j(x) = \begin{cases} g_j(x)^+, & 1 \leqslant j \leqslant q \\ |h_j(x)|, & q+1 \leqslant j \leqslant m \end{cases} \tag{3.9}$$

La difficulté majeure dans cette approche est la définition de la fonction de pénalité p,

2. On les appelle aussi solutions faisables ou réalisables

3. Ici on a considéré un seul paramètre pour toutes les contraintes. Notons qu'on peut associer à chaque contrainte j un paramètre de pénalisation α_j

ou plus précisément, l'ajustement du paramètre de pénalisation α_k. Trois approches sont généralement utilisées pour définir cette fonction : l'approche statique, l'approche dynamique et l'approche adaptative.

3.5.2 Pénalisation statique

Avec la méthode de pénalisation statique, le paramètre de pénalisation α_k es fixé par l'utilisateur (ajusté par essai-erreur), et reste constant au cours de l'évolution. Cette approche peut donner de très bons résultats mais nécessite un ajustement très fin de la valeur de α_k [118]. En effet,

- si α_k est trop petit, l'optimum du problème ne respectera pas strictement les contraintes;
- si α_k est trop grand, la recherche se fait essentiellement dans le domaine faisable de l'espace de recherche, interdisant tout passage par l'espace infaisable, ce qui pénalise l'exploration de tout le domaine et augmente le risque de la convergence prématurée.

Pour remédier à cette difficulté, une idée naturelle est alors de faire varier le paramètre de pénalisation dynamiquement au cours de l'évolution.

3.5.3 Pénalisation dynamique

Contrairement à la méthode statique, avec la stratégie dynamique, le paramètre de pénalisation est modifié au long de l'évolution. L'idée de base de l'approche dynamique est de donner initialement une petite valeur au paramètre de pénalisation pour favoriser l'exploration, puis l'augmenter au fur et à mesure de l'évolution afin d'augmenter la pression sur l'algorithme pour trouver des individus faisables.

Plusieurs stratégies ont été proposées dans la littérature. On cite par exemple la méthode de Joines et Houk [93] ; à une génération k donnée la fonction de pénalité est calculée comme suit :

$$p(x) = (\alpha_0 \times k)^\gamma \sum_{j=1}^{m} (v_j(x))^\beta \tag{3.10}$$

ou α_0, γ et β sont des constantes positives.

Dans le cas particulier $\gamma = 1$ l'approche est dite linéaire : le paramètre de pénalisation croît linéairement avec le nombre de générations.

Une deuxième approche similaire à la première a été proposée pour l'optimisation de

formes dans [99] qui consiste à augmenter le paramètre de pénalisation toutes les M générations. La fonction de pénalité, à la génération k, est donnée par

$$p(x) = (\alpha_0 \times \beta^{[k/M]}) \sum_{j=1}^{m} (v_j(x)) \tag{3.11}$$

où α_0 et β sont des constantes et $[k/M]$ désigne la partie entière de k/M

L'approche dynamique, dans laquelle le paramètre de pénalisation est modifié suivant une stratégie spécifiée par l'utilisateur, requiert une bonne intuition initiale pour déterminer une stratégie efficace. Pour remédier à ce problème, l'idée est de faire adapter le paramètre de pénalisation à chaque génération : il peut être augmenté ou diminué selon l'importance des violations des contraintes et l'état de la recherche. C'est le principe des techniques de pénalisation dites adaptatives, que nous présentons maintenant.

3.5.4 Pénalisation adaptative

La plupart des algorithmes évolutionnaires utilisent de nombreux paramètres que l'utilisateur doit ajuster pour chaque problème particulier, et la technique systématique par essai-erreur est évidemment très coûteuse. Différentes méthodes, regroupées sous la terminologie générale de techniques adaptatives ont été proposées pour remédier à ce problème : les valeurs des paramètres peuvent être déduits de statistiques sur les itérations précédentes – comme dans la règle du 1/5ème pour la taille du pas de mutation proposée par [139] (cf. section 1.7.2) – ou bien évoluer suivant les opérateurs d'évolution [158] (on parle alors de techniques auto-adaptatives). Un petit nombre de paramètres doivent encore être ajustés par l'utilisateur (e.g. les valeurs initiales et les schémas de mise à jour), mais on peut les considérer comme des paramètres du second ordre : dans une large plage de valeurs, l'algorithme restera robuste (pour plus de détails, nous renvoyons le lecteur à [118]).

Des paramètres de pénalisation adaptatifs ont été utilisés avec succès pour des problèmes de satisfaction de contraintes discrètes [4] (*Constraint Satisfaction Problems*) [50], où l'objectif est de trouver au moins un individu admissible.

Dans le contexte de l'optimisation de paramètres, deux schémas adaptatifs ont été proposés, le premier en 1992 par Hadj-Alouane et Bean [75], le second en 1993 par Smith et Tate [163]. Le paramètre de pénalisation est mis à jour en fonction de la faisabilité du meilleur individu dans la population des générations précédentes.

4. Pour ces problèmes, les contraintes sont des fonctions booléennes, et la satisfaction d'une contrainte est vérifiée si sa fonction est égale à 1

Méthode de Hadj-Alouane et Bean

Avec cette méthode le paramètre de pénalisation est mis à jour en fonction de la faisabilité du meilleur individu dans la population des générations précédentes. La fonction de pénalité est définie comme suit :

$$p(x) = \alpha_k \sum_{j=1}^{m} (v_j(x))^2$$

où α_k est mise à jour chaque génération k comme suit :

$$\alpha_{k+1} = \begin{cases} (1/\beta_1) \cdot \alpha_k & si\ x_b^i \in F\ \forall i,\ k-t+1 \leqslant i \leqslant k \\ \beta_2 \cdot \alpha_k & si\ x_b^i\ \notin F\ \forall i,\ k-t+1 \leqslant i \leqslant k \\ \alpha_k & sinon, \end{cases}$$

où x_b^i est le meilleur individu de la population, à la génération i, β_1,$\beta_2 > 1$ et $\beta_1 \neq \beta_2$ (pour éviter les cycles). Autrement dit, la valeur du paramètre de pénalisation α_{k+1} à la génération $k+1$ diminue, avec un facteur multiplicatif $1/\beta_1$, si tous les meilleurs individus pendant les t dernières générations étaient faisables et augmente, avec un facteur multiplicatif β_2, si tous les meilleurs individus pendant les t dernières générations étaient infaisables. S'il y a eu pendant ces générations à la fois des meilleurs solutions faisables et d'autres infaisables, la valeur de α_k ne change pas.

Hadj-alouane et Bean ont introduit cette approche adaptative pour augmenter l'efficacité du processus de recherche. Ainsi, si les contraintes ne posent pas de problème pour l'algorithme, les pénalités diminuent, sinon elles augmentent. On remarque que cette méthode introduit trois paramètres, β_1,β_2 et t, qui doivent être ajustés par l'utilisateur.

Méthode de Smith et Tate

La fonction de pénalisation adaptative proposée par Smith et Tate dépend de la qualité de la meilleure solution pendant l'évolution (jusqu'à l'itération courante), ainsi que du degré de violation des contraintes (déterminé par la distance des meilleurs solutions infaisables au domaine faisable). La fonction de pénalité est définie comme suit :

$$p(x) = \mathcal{F}_{feas}(t) - \mathcal{F}(t) \sum_{j=1}^{m} (v_j(x)/q_j(t))^k,$$

où $\mathcal{F}(t)$ est la valeur de la fonction objectif (sans pénalisation) de la meilleure solution trouvée pendant l'évolution(jusqu'à la génération t), $\mathcal{F}_{feas}(t)$ est l'évaluation de la meilleure solution faisable trouvée pendant l'évolution, k est une constante et $q_j(t)$ désigne l'estimation du seuil d'extension de faisabilité pour chaque contrainte j $(1 \leqslant j \leqslant m)$; un tel seuil détermine les solutions infaisables intéressantes (proches du domaine faisable).

Discussion

Les deux techniques de pénalisation adaptatives présentées dans cette section ne tiennent compte que de l'état des meilleurs individus pendant l'évolution, sans se soucier de l'état général de la population; ce qui conduit l'algorithme dans certains cas à la convergence prématurée. En effet, considérons la première approche. Si le meilleur est réalisable pendant plusieurs générations mais tous les autres individus ne le sont pas, le paramètre de pénalisation est ainsi diminué, alors qu'il devrait être augmenté pour générer d'autres solutions réalisables. Pour cette raison, nous proposons deux nouvelles approches adaptatives qui utilisent des informations globales sur la population, afin d'ajuster le paramètre de pénalisation, que nous introduisons dans la suite.

3.5.5 Pénalisation adaptative basée sur la population

Les deux méthodes de pénalisation adaptative que nous allons introduire partent d'un même principe qui consiste à effectuer la mise à jour des paramètres de pénalisation en utilisant les statistiques globales de faisabilité dans la population. Le but de cette démarche est d'explorer les environs de la frontière de la région admissible en essayant de conserver dans la population les individus qui se situent "de part et d'autre" de cette frontière (cette même idée a mené aux algorithmes "ségrégatifs" – Segregated GA – [108] qui utilisent deux différents paramètres de pénalisation pour atteindre ce but).
En effet, dans de nombreux problèmes d'optimisation, on sait que la solution se situe sur la frontière de la région admissible. Des techniques spécifiques de traitement des contraintes ont d'ailleurs été proposées pour explorer uniquement cette frontière [152, 151] lorsque ceci est possible .

Toutefois, pour l'optimisation topologique de formes, avec la rigidité comme fonction-objectif, bien qu'il n'a pas été établi si la solution est sur la frontière du domaine admissible ou pas pour le problème continu, le bon sens suggère qu'il se trouve "proche" de cette frontière: intuitivement, si on est loin de cette frontière (i.e. le déplacement maximal est bien plus petit que la limite autorisée), on peut sans doute enlever de la matière sans violer la contrainte. De plus, cette frontière est difficilement caractérisable pour ce problème, mais on peut explorer la zone admissible voisine de la frontière grâce à la méthode de pénalisation adaptative.

La première approche

Nous proposons dans ce paragraphe une nouvelle approche de pénalisation adaptative basée sur le degré de faisabilité de la population. Le but de cette méthode est d'ajuster le paramètre de pénalisation selon la proportion d'individus satisfaisant les contraintes à la

génération courante k, notée $\Theta^k_{feasible}$. La stratégie d'ajustement est donnée par l'équation suivante :

$$\alpha_{k+1} = \begin{cases} \beta \cdot \alpha_k & si\ \Theta^k_{feasible} < \Theta_{lim} \\ (1/\beta) \cdot \alpha_k & si\ \Theta^k_{feasible} > \Theta_{lim} \end{cases} \tag{3.12}$$

où $\beta > 1$ est un paramètre défini préalablement par l'utilisateur, $\Theta^k_{feasible}$ désigne la proportion d'individus faisables à la génération courante k, et Θ_{lim} est le taux de faisabilité limite permettant de décider l'augmentation ou la diminution du paramètre de pénalisation. Notons que pour cette approche trois paramètres sont à ajuster par l'utilisateur, β, Θ_{lim} et la valeur initiale de α_0.

La deuxième approche

Dans ce paragraphe nous proposons une deuxième approche adaptative basée, comme la première, sur la proportion de faisabilité de la population. Le but est de garder dans la population une proportion minimale d'individus satisfaisant les contraintes ainsi qu'une proportion minimale qui les violent. Notons $\Theta^k_{feasible}$ la proportion d'individus qui satisfont les contraintes à la génération k, et Θ_{inf} et Θ_{sup} deux paramètres donnés par l'utilisateur. Les faibles valeurs des paramètres de pénalisation favorisent les individus qui violent les contraintes (et inversement). Pour maintenir $\Theta^k_{feasible} \in [\Theta_{inf}, \Theta_{sup}]$, nous proposons la règle de mise à jour suivante :

$$\alpha_{k+1} = \begin{cases} \beta \cdot \alpha_k & \text{si } \Theta^k_{feasible} < \Theta_{inf} \\ (1/\beta) \cdot \alpha_k & \text{si } \Theta^k_{feasible} > \Theta_{sup} \\ \alpha_k & \text{sinon} \end{cases} \tag{3.13}$$

avec $\beta > 1$. Les paramètres choisis par l'utilisateur dans cette méthode sont Θ_{inf}, Θ_{sup}, β et la valeur initiale α_0. Si $\Theta_{inf} = \Theta_{sup}$ on est dans le cas 1.

Remarque

Notons que, pour les deux approches proposées, les variations de α_k ne sont pas monotones, et il n'y a donc pas de garantie a priori *que le meilleur individu de la population satisfasse les contraintes. Il peut même arriver que la population ne contienne aucun individu admissible – même si dans ce cas, l'augmentation régulière de la valeur de α_k doit favoriser les individus violant le moins les contraintes, devant mener à l'émergence d'individus admissibles.*

3.6 Résultats comparatifs

Cette section récapitule et discute les résultats comparatifs obtenus pour les différentes approches de pénalisation. Les expériences ont été effectuées sur deux problèmes test

simples de l'optimisation topologique de formes (problèmes de la plaque console encastrée 1×2 et 2×1), introduit dans la section 3.3.2. Signalons que pour les tests de ce chapitre nous avons utilisé la représentation par diagrammes de Voronoï (cf. section 4.2, chapitre 4).

3.6.1 Conditions expérimentales

Pour permettre les comparaisons entre les résultats donnés pour les différents schémas de pénalisation nous avons utilisé les mêmes conditions expérimentales suivantes : l'algorithme d'évolution artificielle utilisé est similaire à un algorithme génétique "classique" (sélection linéaire basée sur le rang et remplacement générationnel) avec des populations de 80 individus ; les taux de croisement et de mutation sont respectivement 0.6 et 0.3 ; le nombre de générations maximum est de 2000 et le critère d'arrêt de l'algorithme est de 300 itérations successives sans amélioration du meilleur individu (cf. chapitre 4, section 4.3.1, pour plus de détails sur les conditions expérimentales).

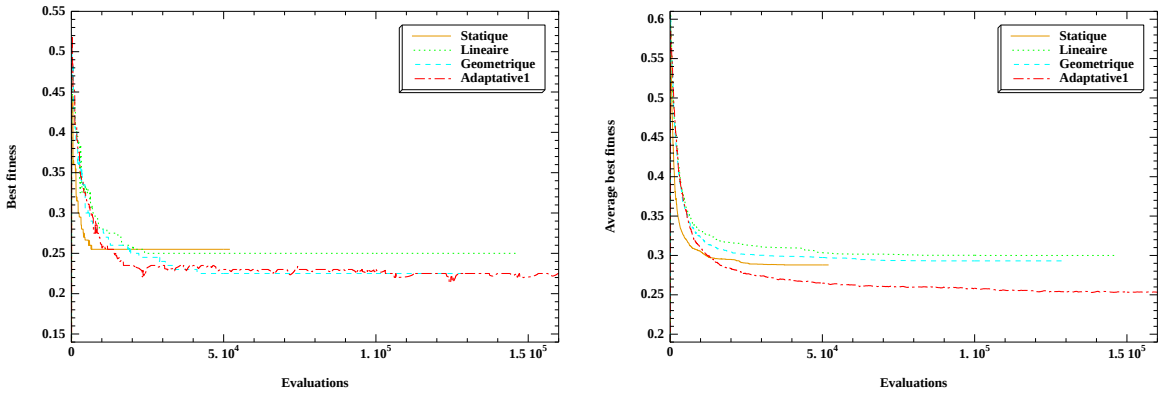
Nous commençons par étudier expérimentalement l'influence du paramètre de pénalisation statique sur la solution. Nous verrons dans le paragraphe suivant que cette influence n'est pas négligeable pour un problème d'optimisation topologique de formes.

Pénalité statique

Pour trouver la pénalisation adéquate, pour le cas statique, on est obligé de tester plusieurs valeurs jusqu'à ce qu'on trouve celle qui permet de donner des résultats satisfaisants. Des séries d'essais numériques avec différentes valeurs de α ont été effectuées sur les deux cas tests simples de l'optimisation topologique de formes. Les valeurs sont comprises entre 0.1 et 1000, afin de visualiser l'effet du choix des petites et des grandes valeurs du paramètre α. Les résultats obtenus démontrent que l'influence de α sur le problème d'optimisation n'est pas négligeable : pour des petites valeurs de α (par exemple 0.1), les structures obtenues sont très légères mais la contrainte de déplacement est violée, alors que pour des grandes valeurs de α (par exemple 1000), les structures obtenues sont faisables mais leur poids est assez important. Une bonne valeur intermédiaire a été choisie, $\alpha = 30$, donnant des résultats raisonnables pour la plupart des tests. Cette valeur sera utilisée comme la valeur initiale pour les autres approches.

3.6.2 Premiers résultats

Nous présentons dans ce paragraphe les résultats comparatifs obtenus pour différentes stratégies de pénalisation sur le problème de la plaque console 1×2. Deux méthodes de pénalisation dynamique ont été incluses dans la comparaison, à savoir le schéma linéaire et le schéma géométrique (cf. 3.5.3). Pour cette première étude expérimentale nous avons



(a) (b)

FIG. 3.5 – *Meilleure des meilleures performances (a) et moyenne des meilleures (b) au cours de l'évolution (moyenne sur 21 calculs indépendants) sur le problème du cantilever* 1×2. $D_{lim} = 20$.

considéré la première approche adaptative proposée ici.

Après plusieurs tests de robustesse, les valeurs (des paramètres spécifiques à chaque approche) suivants ont été choisies et seront utilisées dans toutes les simulations présentées dans cette thèse :

- Approche linéaire (eq. 3.10): $\beta = 1$.
- Approche géométrique (eq. 3.11) : comme dans [99], $\beta = 1.01$ et $M = 10$; ce qui revient à multiplier le paramètre α par le facteur β toutes les 10 générations.
- Approche adaptative 1 (eq. 3.12) : $\beta = 1.1$,$\Theta_{lim} = 0.6$,.
- Approche adaptative 2 (eq. 3.13) : $\beta = 1.1$,$\Theta_{inf} = 0.4$, et $\Theta_{sup} = 0.8$.

Plaque 1×2**,** $D_{lim} = 20$

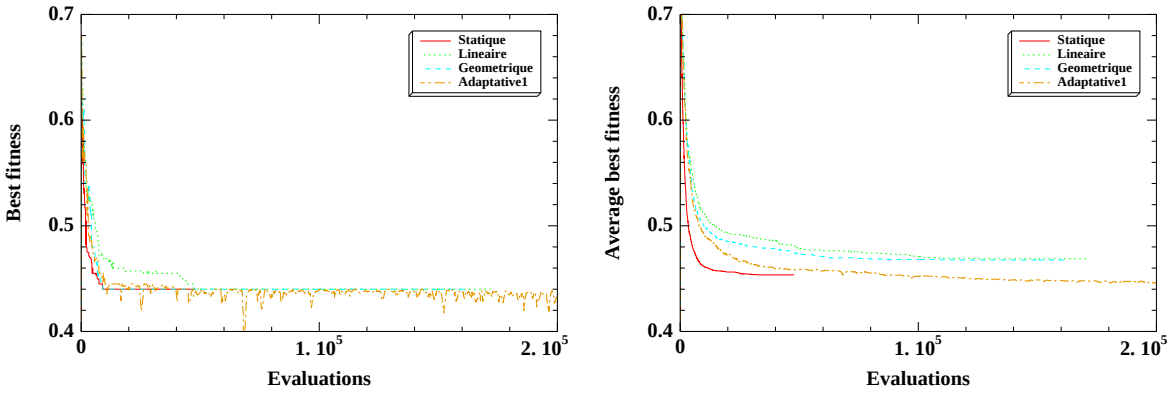
Pour chaque cas, 21 essais indépendants ont été effectués dans les mêmes conditions expérimentales décrites ci-dessus. Les résultats enregistrés sur le premier cas test, avec les 4 différentes approches, statique, linéaire, géométrique et adaptative 1 sont illustrés sur la figure 3.5. Les courbes de la figure 3.5 montrent l'évolution de la valeur minimale et moyenne des meilleures solutions trouvées sur 21 tests indépendants. La première observation qu'on déduit de la figure (b) est que la courbe correspondant à la méthode adaptative montre clairement que la meilleure performance dans la population correspond parfois à

un individu infaisable, ce qui explique la présence des oscillations dans la courbe pour la meilleure performance.

En ce qui concerne la comparaison, la figure 3.5 montre bien que la stratégie adaptative est plus performante que les autres méthodes en moyenne, et égale ou dépasse le meilleur résultat obtenu par les autres méthodes. D'autre part, l'approche géométrique peut donner des résultats qui sont aussi bons que les meilleurs de l'approche adaptative, mais la variance des résultats obtenus par cette approche est très élevée : certains résultats sont mauvais. Ainsi sa performance est moins bonne en moyenne. La pénalité statique peut donner rapidement de bons résultats à condition que le paramètre soit bien choisi, mais il se trouve qu'elle est incapable d'améliorer l'optimum après un (petit) nombre de générations.

Plaque 1×2, $D_{lim} = 10$

En vue de montrer la performance et la robustesse de l'approche adaptative nous avons comparé ces différentes approches dans un premier temps sur le même cas test mais avec une contrainte plus forte. Les résultats obtenus sont illustrés sur la figure 3.6. Ces résultats confirment la performance de la stratégie adaptative par rapport aux autres approches.



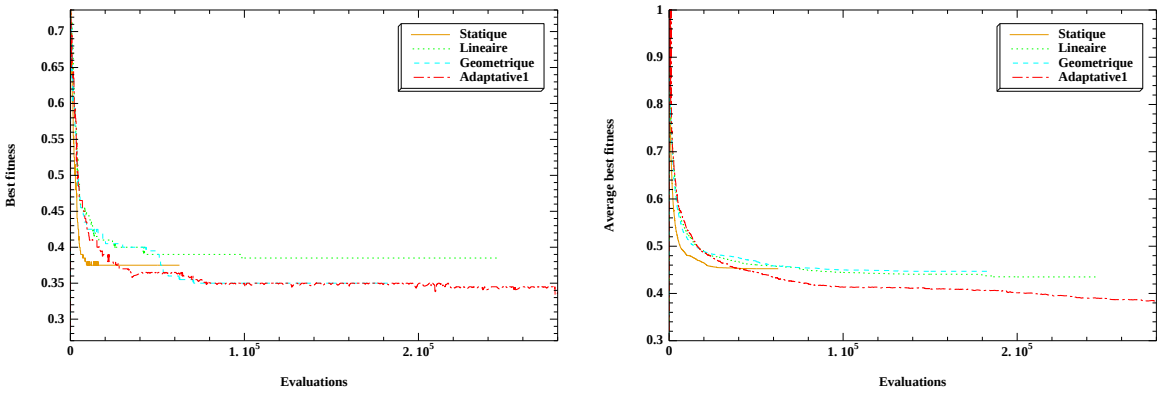
(a) (b)

FIG. 3.6 – *Meilleure des meilleures performances (a) et moyenne des meilleures (b) au cours de l'évolution (moyenne sur 21 calculs indépendants) sur le problème du cantilever* 1×2. $D_{lim} = 10$.

Plaque 2×1, $D_{lim} = 220$

Comme pour le cas précédent, les résultats obtenus sur le problème de la plaque console 2×1 (cf. figure 3.7) confirment la performance de la stratégie adaptative par rapport aux

autres approches.



(a) (b)

FIG. 3.7 – *Meilleure des meilleures performances (a) et moyenne des meilleures (b) au cours de l'évolution (moyenne sur 21 calculs indépendants) sur le problème du cantilever* 2×1. $D_{lim} = 220$

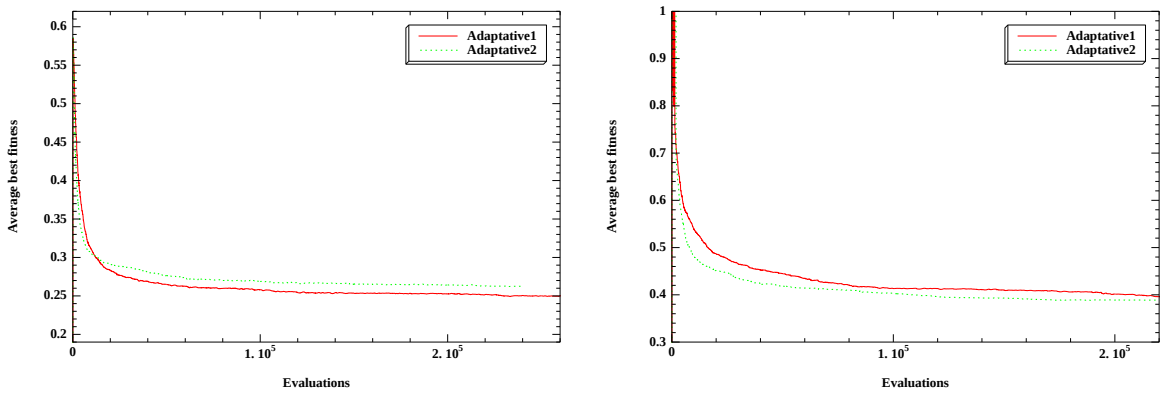
Remarque

D'autres résultats comparatifs montrant la validité de cette approche adaptative basée sur la population, sur d'autres problèmes sous contraintes, se trouvent dans [77] et [20].

3.6.3 Comparaison des deux approches adaptatives

Comme pour la première approche adaptative, une série de 21 tests indépendants ont été effectués, dans les mêmes conditions expérimentales (à part bien entendu les paramètres spécifiques de l'approche), en utilisant la deuxième approche adaptative. Les courbes de la figure 3.8 illustrent l'évolution de la valeur moyenne des meilleures performances sur 21 tests pour les deux approches adaptatives, appliquées sur les deux cas test du cantilever 1×2 et 2×1.
La première observation qu'on déduit de ces courbes est que la première approche dépasse légèrement en moyenne la deuxième sur le cantilever 1×2, mais cette observation n'est plus valable sur le cantilever 2×1 (la deuxième approche dépasse la première). On ne peut donc pas dire laquelle des deux méthodes adaptatives est la meilleure.

Par ailleurs, en regardant de près les meilleures solutions (au sens de la fitness pénalisée) obtenues par les deux approches, on remarque que la première technique arrive pour la plupart des expériences à des solutions infaisables violant légèrement la contrainte de déplacement (des solutions proches de la frontière de l'espace faisable). Alors que la



(a) cantilever 1×2 , $D_{lim} = 20$ (b) cantilever 2×1 , $D_{lim} = 220$

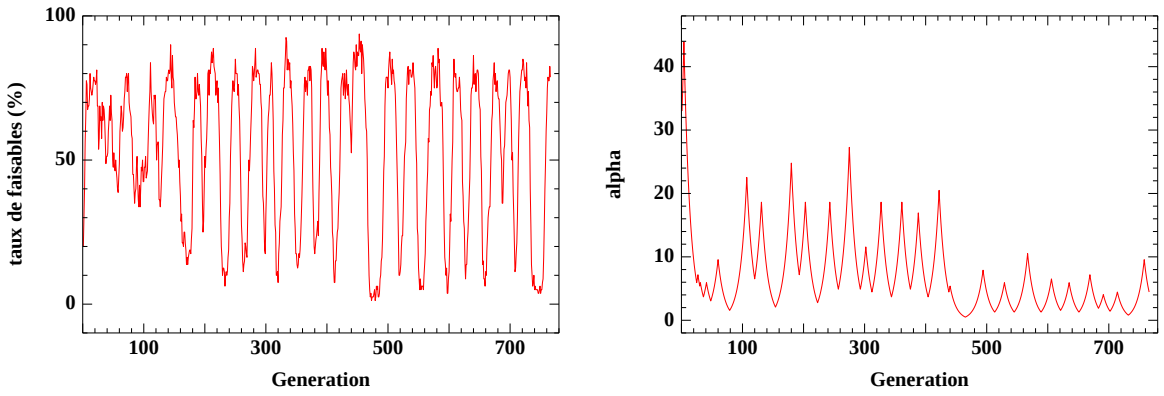
FIG. 3.8 – *Moyenne des meilleures performances au cours de l'évolution (moyenne sur 21 calculs indépendants)*

deuxième stratégie trouve pour la plupart des tests des solutions faisables.

Dans la figure 3.9 nous avons tracé l'allure de l'évolution du nombre de faisables dans la population et l'évolution du paramètre de pénalisation α au cours des générations, pour un essai de résolution du problème de la cantilever 1×2. La stratégie de pénalisation utilisée est celle de la première approche adaptative avec un taux limite 0.6 et $\alpha_0 = 30$. On remarque que le taux de faisables dans la population est très variable tout au long de l'évolution. Ces variations sont mises en évidence dans la figure 3.9-a par des oscillations dans un intervalle de taille variable et centré en 60% (le taux limite). Ces augmentations et diminutions de degré de faisabilité engendrent en parallèle des augmentations et diminutions du coefficient de pénalité α, qui oscille aussi sur un intervalle variable (cf. figure 3.9-b). En effet, d'après l'équation 3.12, l'augmentation de la valeur du coefficient α permet de favoriser les individus faisables pendant les procédures de sélection ultérieures, alors que sa diminution favorise les infaisables.

3.7 Conclusion

La puissance de l'adaptativité dans le calcul évolutionnaire est maintenant bien connue et les résultats présentés dans ce chapitre le confirment au niveau de la fonction de performance. En effet, deux approches de prise en compte des contraintes, basées sur la pénalité adaptative ajustée automatiquement selon le taux d'individus faisables dans la population, ont été proposées. Les expérimentations réalisées sur deux problèmes test classiques de l'optimisation topologique de formes ont montré la performance et la robustesse des



(a) (b)

FIG. 3.9 – *Evolution du degré de faisabilité (a) et de α (b) au cours des générations. Problème du cantilever* 1×2. $D_{lim} = 20$. *Adaptative 1,* $\Theta_{lim} = 0.6$, $\alpha_0 = 30$

approches adaptatives par rapport aux approches statiques et dynamiques utilisées dans des travaux antérieurs.

Mais une nouvelle direction concernant la fonction performance, qui pourrait apporter une amélioration significative pour les ingénieurs mécaniciens, pourrait être l'utilisation des techniques multi-critères plutôt que des méthodes traitant les contraintes : les contraintes peuvent être considérées comme autant d'objectifs. Nous étudions les avantages de ces approches dans le chapitre 6, ainsi que les résultats numériques comparatifs.

Nous pouvons maintenant revenir sereinement au problème crucial de la représentation pour les structures. C'est l'objet du chapitre suivant.

Chapitre 4

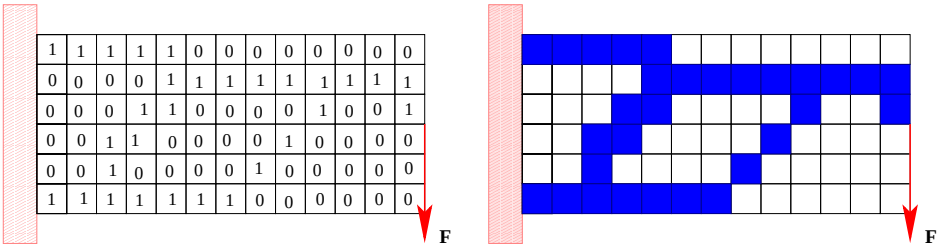
Représentations basées sur les diagrammes de Voronoï

Sommaire

4.1 Introduction

Nous nous intéressons dans ce chapitre aux problèmes de représentations pour les structures. Une des difficultés principales de l'optimisation topologique de formes par algorithmes évolutionnaires est la représentation d'une forme, c'est à dire le choix de l'espace de recherche. L'approche la plus "naturelle", utilisée dans tous les travaux antérieurs [91, 27, 99], consiste à partir d'un maillage donné (utilisé pour le calcul du comportement mécanique de la structure qui détermine sa performance), et à attribuer à chaque élément de ce maillage une valeur 1 ou 0, i.e matière ou vide (cf. figure 4.1) : l'espace de recherche est alors celui de tableaux de bits (*bit-array*), qui se trouve être le cadre traditionnel des algorithmes génétiques, avec des opérateurs de variations plus adaptés au problème de l'optimisation de formes. Bien que cette technique ait donné de bons résultats, elle est limitée par la dépendance de la complexité de la représentation (le nombre de variables de design) à celle du maillage. En effet, si la taille du maillage augmente, la taille du problème suit : l'augmentation de la taille des individus (tableaux de bits) imposera de choisir une taille de population plus importante [32, 66], nécessitant ainsi un plus grand nombre de générations pour la convergence de l'algorithme. Il est donc impensable d'appliquer cette technique aujourd'hui à des maillages très fins - sans parler des problèmes 3D. Pour remédier à cet inconvénient, nous proposons dans ce chapitre un ensemble de représentations compactes dont la complexité est indépendante de celle de tout maillage, et auto-adaptative (i.e la complexité des solutions est ajustée par l'algorithme lui-même). Dans ces nouveaux espaces de représentations, un individu est une liste non ordonnée et de longueur variable : on dit que ces représentations sont non structurées.
La représentation par diagrammes de Voronoï est un premier pas vers les représentations non structurées pour l'optimisation topologique de formes. Elle a été proposée pour la première fois dans [147], et a été utilisée essentiellement pour des problèmes d'identification [150, 149, 112].

Ces représentations se basent à l'origine sur la géométrie algorithmique [132, 25]. De plus, cette approche se distingue de manière fondamentale de la représentation par tableaux



Un individu: tableau de bits — La structure correspondante

FIG. 4.1 – *Représentation d'une forme par un tableau de bits*

de bits par son adaptativité: chacun des individus peut avoir un nombre de paramètres qui diffère de celui de ses congénères, et en outre ce nombre peut changer au cours de l'évolution (c'est une représentation de longueur variable). Nous verrons ainsi dans la section 4.2 que deux opérateurs de mutation et l'opérateur de croisement utilisent la caractéristique de longueur variable de la représentation durant l'évolution.

Ce chapitre est organisé de la façon suivante: En section 4.2, nous introduisons la représentation de Voronoï et les opérateurs de variation associés. Nous présentons ensuite, dans la section 4.3, les résultats numériques obtenus sur le problème de l'optimisation du moment d'inertie, et sur plusieurs problèmes de l'optimisation topologique de formes: le problème test classique de la plaque console encastrée; le modèle en "L"; un problème possédant plusieurs solutions quasi-optimales; un problème de cantilever 10×1, difficile à résoudre par l'approche évolutionnaire bit-array; divers problèmes de chargements multiples - i.e devant résister à différents chargements - (cadre de vélo, le modèle du pont 2D et le modèle "L"); enfin un problème tridimensionnel sur lequel nous avons obtenu des résultats originaux (les premiers du genre). De plus, nous discutons les avantages de cette représentation par rapport à la représentation standard bit-array. Ensuite, nous proposons dans les sections 4.4 et 4.5 deux nouvelles représentations qui ont des propriétés analogues à la représentation par diagrammes de Voronoï. Dans la section 4.6, des résultats comparatifs sur le benchmark du cantilever justifient l'introduction des autres représentations de type Voronoï. Enfin, nous esquissons, dans la section 4.7, ce que pourrait être une représentation modulaire permettant la réutilisation d'une partie de la représentation.

4.2 Représentation par diagrammes de Voronoï

Utilisée avec succès dans le problème connexe de l'identification d'inclusions élastiques [150] et l'identification des modèles géologiques [149], cette représentation se base sur un parti-

tionnement du domaine par des polyèdres convexes.

Diagrammes de Voronoï :

Considérons un nombre fini de points $V_0, \ldots, V_N$ (les *sites de Voronoï*) dans un domaine Ω donné de $\mathbb{R}^n$ (le domaine de travail). A chaque site V_i on associe l'ensemble de tous les points du domaine de travail pour lesquels le site de Voronoï le plus proche est V_i. On nomme cet ensemble *cellule de Voronoï*:

$$Cell(V_i) = \{M \in \Omega,\, d(M,V_i) = \min_{j=1\cdots N} d(M,V_j)\}$$

où $d(.,.)$ dénote la distance euclidienne.

Le *diagramme de Voronoï* est la partition du domaine défini par les cellules de Voronoï. Chaque cellule est un sous-ensemble polyédral du domaine de travail ; réciproquement, toute partition d'un domaine de $\mathbb{R}^n$ en sous-ensembles polyédraux est le diagramme de Voronoï d'au moins un ensemble de sites de Voronoï (voir [132] pour une introduction détaillée aux diagrammes de Voronoï et une présentation générale de la géométrie algorithmique). Dans toute la suite nous ne considérons (sauf indication contraire) que les domaines de $\mathbb{R}^2$.

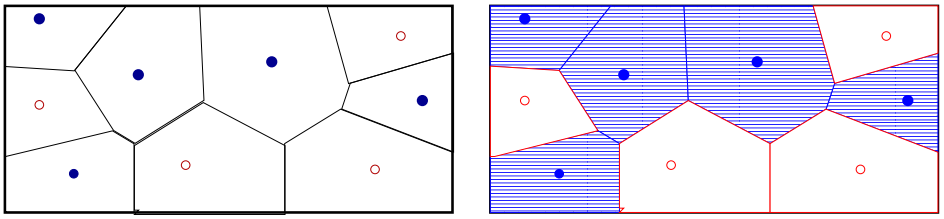
4.2.1 Le génotype

Considérons maintenant une liste – de longueur variable – de sites de Voronoï, chaque site étant étiqueté 0 ou 1. Si chaque cellule est à son tour étequitée par l'étiquette de son site, le diagramme de Voronoï correspondant représente une partition du domaine de travail en deux sous-ensembles. La figure 4.2 montre l'exemple d'un partitionnement d'un domaine à l'aide d'un diagramme de Voronoï. Ainsi une forme est représentée par une liste de N paires (site de Voronoï, label). Les variables du problème sont donc les N triplets $(X_{V_i},Y_{V_i},C_{V_i})$, où X_{V_i},Y_{V_i} représentent les coordonnées de V_i évoluant dans des intervalles réels bornés, $C_{V_i} \in \{0,1\}$ et $N \leqslant N_{max}$ (nombre maximal de sites) [1].

4.2.2 Décodage

Comme pour l'utilisation d'un codage binaire des individus (cf. section 1.6), le choix de la représentation par diagrammes de Voronoï nous contraint donc à définir une fonction de décodage. En effet, le calcul de la fonction de performance d'une structure (cf. chapitre 3)

1. En théorie le nombre de sites n'est pas limité mais pour des raisons techniques on fixe un nombre maximal.

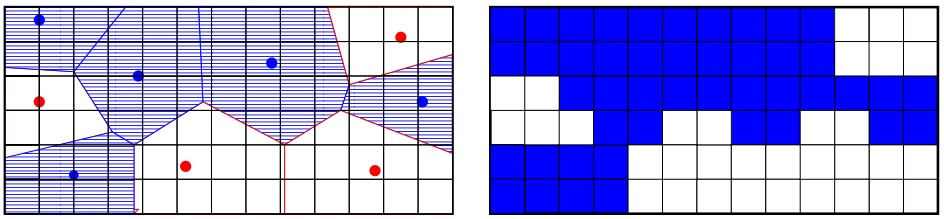


(a) (b)

FIG. 4.2 – *La représentation de Voronoï sur un domaine rectangulaire : Une liste de points définit un pavage en polygones convexes - les cellules de Voronoï (a). À chaque site est de plus attachée une variable booléenne, et la cellule correspondante est alors considérée comme "matière" ou "vide" en fonction de la valeur de cette variable, définissant ainsi une forme (b).*

est obtenu par simulation numérique de son comportement mécanique, en utilisant la méthode des éléments finis, qui nécessite l'utilisation de maillages plus fins que le maillage sous forme de diagrammes de Voronoï.

Par ailleurs, l'utilisation d'un maillage différent pour chaque forme considérée introduirait un bruit numérique dû au remaillage: le calcul du comportement de deux structures voisines avec deux maillages différents pourrait donner des résultats non significatifs. Il est donc nécessaire d'utiliser le même maillage à l'intérieur d'une même génération. Ainsi, pour un maillage régulier donné, une partition décrite par un diagramme de Voronoï peut facilement se projeter sur ce maillage : un élément appartient à l'une ou l'autre des catégories (matériau ou vide) en fonction du label de la cellule de Voronoï dans laquelle se trouve son centre de gravité. La valeur au centre d'une maille est alors donnée par la valeur de la cellule de Voronoï dans laquelle elle se situe (cf. figure 4.3).



(a) (b)

FIG. 4.3 – *Projection d'un diagramme de Voronoï sur un maillage régulier* 13×6 *: la fonction de performance est calculée sur la forme projetée (b).*

4.2.3 Initialisation

La procédure d'initialisation consiste en un tirage aléatoire uniforme du nombre de sites de Voronoï (compris entre 1 et un nombre maximum donné par l'utilisateur) et des sites de Voronoï dans la structure. À chaque site est associé un label booléen 0 ou 1 (vide ou matériau), choisi avec une probabilité $p = 0.5$; et c'est cette procédure qui a été employée dans la plupart des essais numériques. Toutefois, dans le cas de cantilever "élancée" 10×1 (cf. section 4.3.10) la plupart des structures de la population initiale ne connectent pas le point d'application de la force avec la frontière encastrée. Pour remédier à ce problème, une autre procédure d'initialisation, dite de densité uniforme, proposée dans [96], est utilisée.

Procédure de densité uniforme (96)

Pour chaque individu,

1. choisir r la densité des labels 1, uniformément dans [0,1],
2. pour chaque site, choisir le label 1 avec la probabilité r.

Avec cette procédure, la probabilité pour qu'un site ait le label 1 est 0.5, et la distribution du nombre total de labels 1 dans la population est uniforme, tout en assurant une diversité maximale en terme de poids des structures. Dans la procédure standard, la distribution du poids des structures suit une loi gaussienne centrée sur 1/2.

4.2.4 Opérateurs de variations

Les opérateurs standards comme le croisement réel ou arithmétique, tels qu'ils sont définis dans le chapitre 1, n'ont plus de sens pour la représentation par diagrammes de Voronoï. Nous devons alors, soit les redéfinir pour qu'ils s'adaptent à la représentation choisie, soit utiliser d'autres opérateurs. Nous optons pour un croisement géométrique similaire à celui utilisé pour la représentation par tableaux de bits [98]. Pour la mutation nous utilisons différents opérateurs de mutation. La suite de cette section est dédiée à la présentation de ces opérateurs.

Opérateur de croisement

Comme son nom l'indique, l'opérateur de croisement géométrique échange les sites de Voronoï sur une base géométrique. De ce point de vue, il est similaire à l'opérateur de croisement spécifique décrit dans [98]. Il consiste à choisir deux parents P_1 et P_2 dans la population, puis à choisir aléatoirement deux points distincts sur la frontière du domaine qui seront situés au même endroit sur les deux parents. Ces deux points vont générer une droite coupant à l'identique chacun des deux parents en deux régions. Les enfants sont

obtenus en échangeant les sites répartis de part et d'autre de cette ligne. La figure 4.4 est un exemple d'application de cet opérateur.

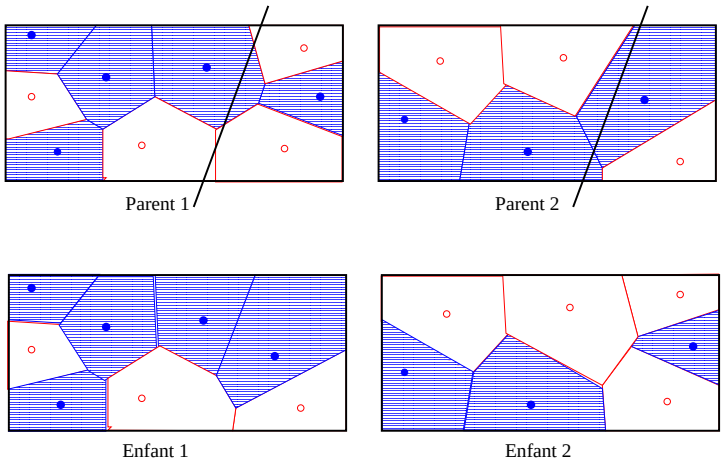


FIG. 4.4 – *La représentation de Voronoï et son opérateurs de croisement : une droite aléatoire est tracée dans chaque diagramme et les sites sont échangés de part et d'autre de cette droite.*

Opérateurs de mutation

L'opérateur de mutation est choisi grâce à une sélection par tirage de roulette, avec des poids définis par l'utilisateur, parmi les opérateurs suivants (cf. figure 4.5) :

• **la mutation de *déplacement***

L'utilisation d'un codage réel des variables d'espaces du problème (coordonnées des sites) amène à l'emploi de mutations réelles telles que nous avons présentées dans la section 1.7.2 du chapitre 1. La mutation de déplacement consiste ainsi à ajouter un bruit Gaussien à chacune des coordonnées d'un site choisi aléatoirement. Comme dans les stratégies d'évolution [158], la mutation log-normale auto-adaptative anisotrope (section 1.7.2 du chapitre 1) est utilisée : une déviation standard est associée à chaque coordonnée de chaque site de Voronoï, et elle subit une mutation log-normale avant d'être utilisée pour la mutation Gaussienne des coordonnées correspondantes. Si (x_i,y_i) désignent les coordonnées d'un site, la relation qui lie les variables avant et après l'opérateur de mutation est de la forme :

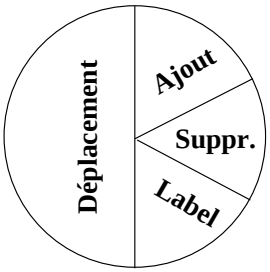


FIG. 4.5 – *Choix entre les mutations (poids à définir).*

$$\begin{cases} \sigma_x^i := \sigma_x^i exp(\tau'_x N(0,1) + \tau_x N_i(0,1)) \\ x_i := x_i + \sigma_x^i N(0,1) \\ \sigma_y^i := \sigma_y^i exp(\tau'_y N(0,1) + \tau_y N_i(0,1)) \\ y_i := y_i + \sigma_y^i N(0,1) \end{cases} \tag{4.1}$$

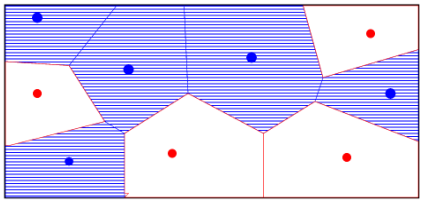
où $N(0,1)$ est une loi de distribution normale centrée d'écart type 1, σ_x^i et σ_y^i sont les écarts types (déviations standards). Dans l'expression (4.1), les coefficients $\tau'_x N(0,1)$ et $\tau'_y N(0,1)$ sont des facteurs globaux qui entraînent une même perturbation de toutes les coordonnées des sites, et les coefficients $\tau_x N_i(0,1)$ et $\tau_y N_i(0,1)$ sont des facteurs locaux : ils autorisent chaque site à avoir sa propre variation. Les divers coefficients τ et τ' sont choisis par extrapolation des résultats des travaux de Schwefel [158] réalisés sur la fonction sphère (cf. section 1.7.2).

• Les mutations *ajoute* et *supprime*

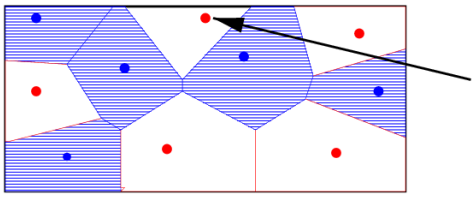
Ces deux opérateurs de mutation agissent sur la longueur variable et la complexité de la représentation. Ils consistent respectivement à ajouter et à supprimer un site choisi aléatoirement dans le domaine. Pour l'ajout, les coordonnées du point seront choisies selon une loi uniforme dans $[0,Xmax] \times [0,Ymax]$, et le label qui lui est associé sera choisi d'une manière aléatoire dans $\{0,1\}$.

• La mutation de *label*

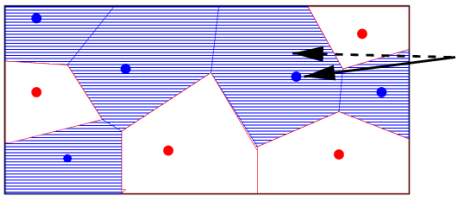
Cet opérateur change aléatoirement l'attribut booléen d'un site, ce qui entraîne un passage du vide à la matière (ou de la matière au vide) de la cellule de Voronoï correspondante.



(a) Le parent



(b) La structure obtenue après l'ajout d'un site dans le diagramme du parent



(c) Un petit déplacement d'un site modifie légèrement la structure du parent.

FIG. 4.6 – *Deux mutations pour la représentation de Voronoï.*

4.3 Résultats numériques

Nous présentons ici les résultats obtenus avec la représentation par diagrammes de Voronoï proposée ci-dessus. Des tests avec différents maillages ont été pratiqués pour s'assurer que la représentation non structurée joue bien le rôle qu'on attend d'elle dans la dépendance de l'algorithme par rapport à la complexité de la discrétisation. Des résultats nouveaux sur un cantilever "allongé" (10×1) et quelques cantilevers tridimensionnels montrent que de telles représentations compactes permettent de franchir une étape dans la résolution de problèmes d'optimisation topologique par algorithmes évolutionnaires.

4.3.1 Paramètres numériques pour les algorithmes évolutionnaires

Sauf indication contraire, les expériences numériques présentées dans la suite utilisent les paramètres suivants : évolution standard de type Algorithme Génétique Générationnel (sélection linéaire basée sur le rang et remplacement de tous les parents par leur descendance) avec des populations de 80 individus ; au plus 40 sites de Voronoï par individu ; taux de croisement de 0.6 et taux de mutation de 0.3 par individu ; les poids relatifs de chaque type de mutation sont de 0.5 pour la mutation de déplacement, les autres types de mutations se partageant en parts égales les 0.5 restants ; le nombre maximum de génération est de 2000 et le critère d'arrêt de l'algorithme est de 300 itérations successives sans amélioration du

meilleur individu ; tous les graphiques sont les résultats de 21 calculs indépendants avec les mêmes paramètres ; les temps CPU sont donnés pour des processeurs Pentium III à 800MHz. Par exemple, le coût d'une génération de 80 individus pour le cantilever 1×2 discrétisé avec 200 éléments finis est de moins d'une seconde. Les valeurs de ces paramètres ont été mises au point après de nombreuses expériences numériques.

Les simulations numériques du comportement des structures ont été faits avec un module d'Young $E = 1$ et un coefficient de Poisson $\mu = 0.3$. Afin de pouvoir effectuer ces calculs sur le maillage fixe du domaine, le vide qui entoure la forme est considéré comme un autre matériau élastique linéaire isotrope très peu rigide. Typiquement, son module d'Young est égal à 10^{-5}. Il faudra cependant être très prudent avec cette astuce technique si l'on se place dans le cadre de l'optimisation modale : en effet, des modes propres apparaissent, qui ne font intervenir que le "vide" et il faut donc les repérer et les éliminer.

L'approche a été validée dans un premier temps sur le problème test simple de l'optimisation de moment d'inertie, brièvement introduit dans le chapitre précédent (section 3.2), puis sur plusieurs problèmes de l'optimisation topologique de formes.

Dans la section suivante, nous reprenons le problème de l'optimisation du moment d'inertie et nous présentons les résultats obtenus.

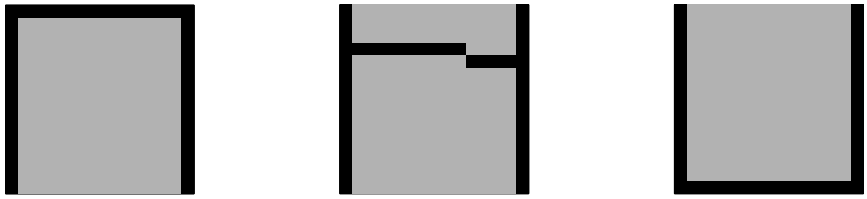
4.3.2 Optimisation du moment d'inertie

Le problème consiste à optimiser le moment d'inertie d'une barre soumise à des efforts de flexion. En supposant que la section droite est constante pour toute la barre, le problème est équivalent à maximiser le moment d'inertie tout en minimisant le poids de cette section (cf. section 3.2).

Considérons le cas test simple où le domaine de travail est le carré $[0,1] \times [0,1]$. La solution attendue a la forme en H : plusieurs solutions sont possibles suivant la position de la barre horizontale puisque on ne spécifie pas comment les moments sont appliqués à la barre. La figure 4.7 montre trois résultats numériques possibles obtenus pour ce cas test, en utilisant la représentation par diagrammes de Vorornoï. Le coût total en temps de calcul pour obtenir une solution, pour ce problème, est en moyenne 27s.

4.3.3 Problème test de cantilever

Le problème test le plus populaire en optimisation topologique de formes est celui de la plaque console "cantilever". Considérons un domaine rectangulaire de dimension 1×2



(a) (b) (c)

FIG. 4.7 – *Trois formes possibles de la section droite avec un poids = 19% et moment d'inertie = 0.033. Temps CPU = 0.03s/génération (900 générations en moyenne). Les éléments de la structure (b) sont considérés comme connectés.*

discrétisé selon un maillage régulier. On suppose que la plaque est encastrée à son côté gauche (le vecteur déplacement est nul) et soumise à une force ponctuelle appliquée au milieu de sa frontière verticale droite (cf. figure 4.8).

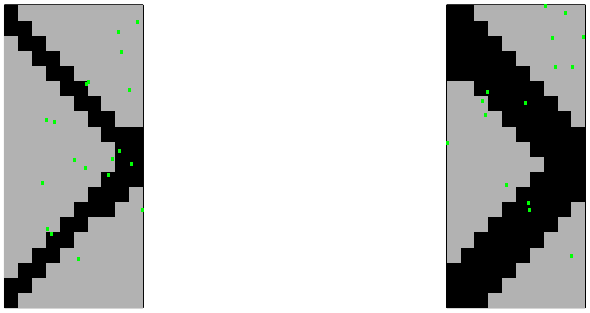
FIG. 4.8 – *Problème de la plaque cantilever standard* 1×2 *et la solution attendue*

Ce cas test est l'un des plus classiques et aussi des plus simples. En effet, toutes les approches (déterministes ou stochastiques) utilisées dans les travaux antérieurs convergent aisément vers la structure optimale bien connue : il s'agit de deux barres qui ont entre elles un angle de 90° (cf. figure 4.8).

Les tracés de la figure 4.9 montrent deux résultats, parmi les plus significatifs, obtenus pour deux contraintes (limites sur le déplacement maximal) respectives $D_{lim} = 20$ et $D_{lim} = 10$, et pour un maillage régulier 10×20. Ces résultats ont été obtenues après plusieurs calculs indépendants utilisant des populations initiales différentes. Les déplacements

limites imposés sont respectés, et les structures obtenues ont la forme en **V** attendue. Le poids (volume occupé par la matière) est exprimé en pourcentage par rapport au volume total du domaine de référence.

Le temps de calcul pour un essai, qui requiert en moyenne 1500 générations pour une population de 80 individus (environ 130000 analyses éléments finis), est de 27 minutes sur un processeur pentium III à 800 MHz.



(a) $D_{lim} = 20$ (b) $D_{lim} = 10$

FIG. 4.9 – *Les deux meilleurs individus pour la représentation de Voronoï, maillage* 10×20. *(a)* $D_{max} = 19.77$*, poids = 21%, 19 sites. (b)* $D_{max} = 9.98$ *, poids = 44%, 15 sites. Les points verts sont les sites*

Les tracés de la figure 4.10 montrent les résultats les plus significatifs pour le même problème avec une discrétisation plus fine 20×40. Pour ce maillage il a fallu en moyenne 1h10min pour chaque essai.
Remarquons que les meilleurs solutions obtenues (pour $D_{lim} = 10$) pour les deux maillages 10×20 et 20×40 (figure 4.9-b et figure 4.10-b) ont des poids différents. On discutera la dépendance par rapport au maillage dans la section 4.3.7.

4.3.4 Élimination des sites inutiles

En analysant les solutions obtenues pour le problème de la plaque cantilever standard (voir par exemple figure 4.9), on remarque qu'il y a des regroupements de sites dans un même voisinage ayant le même label. Ces sites, dits "inutiles", pourraient être supprimés sans entraîner aucune modification du phénotype (la structure). Une étape de suppression des sites inutiles est donc souhaitable afin de diminuer la complexité de la solution. Plusieurs méthodes peuvent être utilisées pour supprimer ces sites. Nous avons testé une approche déterministe associée à une suppression "manuelle", et une méthode qui consiste à augmenter le poids pour la mutation "supprime" (cf. paragraphe 4.2.4).

(a) $D_{lim} = 20$ (b) $D_{lim} = 10$

FIG. 4.10 – *Les deux meilleurs individus pour la représentation de Voronoï, maillage* 20×40. *(a)* $D_{max} = 20$, *poids = 21%, 30 sites. (b)* $D_{max} = 9.99$, *poids = 47%, 12 sites.*

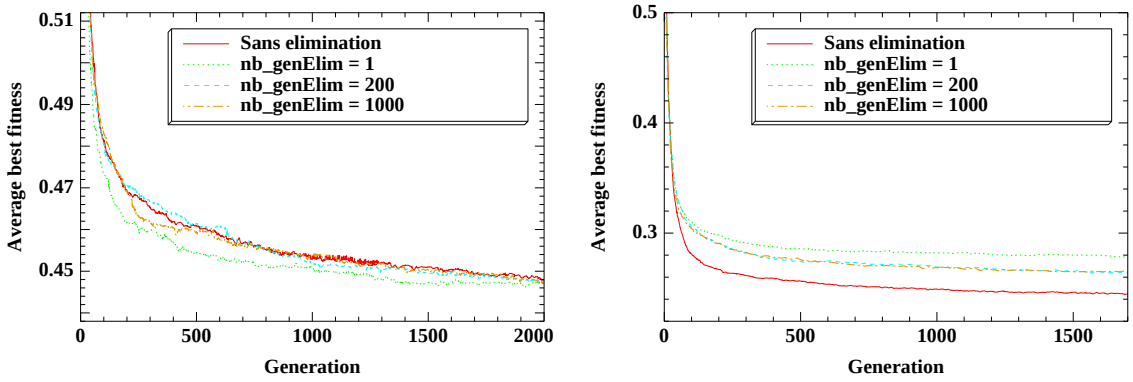
Élimination déterministe

Cette première méthode de suppression consiste à éliminer les sites inutiles, à partir d'un certain nombre de générations $nb\_genElim$, pour toute la population et à chaque génération. En vue d'examiner l'influence de cette approche sur l'évolution de la population au cours des générations, une étude expérimentale sur le cas test simple du cantilever 1×2 a été réalisée. Différentes valeurs de $nb\_genElim$ ont été testées : $nb\_genElim = 1$, $nb\_genElim = 200$, $nb\_genElim = 1000$.

Les courbes de la figure 4.11 illustrent l'évolution de la valeur moyenne des meilleures performances sur 21 tests. Bien que la figure 4.11-a (pour $D_{lim} = 10$) montre, comme on l'attendait, un léger avantage en utilisant la procédure d'élimination, la figure 4.11-b ($D_{lim} = 20$) contredit cette hypothèse. Il est difficile de conclure sur ces résultats, mais on peut dire que les sites inutiles sont en fait utiles pour l'évolution.

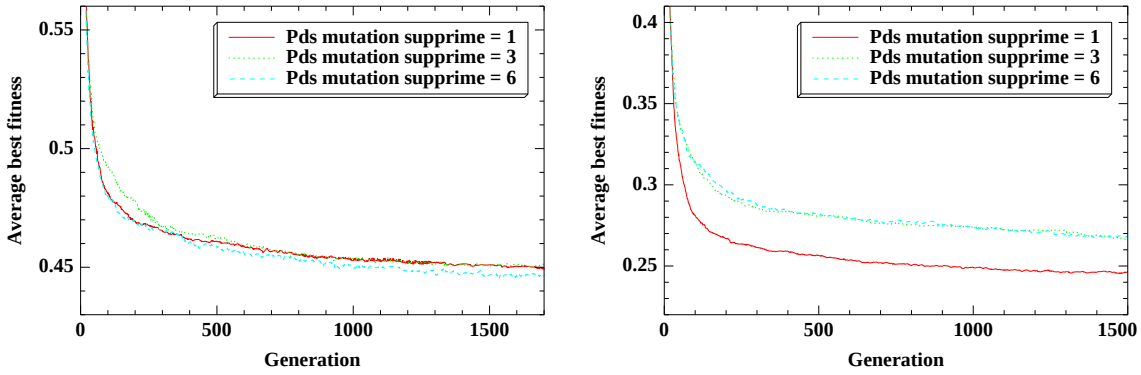
Différents poids pour la mutation "supprime"

Il s'agit d'augmenter le poids de mutation par destruction. Une série d'expériences a été effectuée pour différentes valeurs de poids de mutation "supprime". L'ensemble des résultats est résumé dans la figure 4.12. On remarque que l'augmentation du poids de mutation par destruction améliore légèrement la convergence, pour la contrainte $D_{lim} = 10$. Par contre cette constatation n'est plus valable pour une contrainte plus relaxée : $D_{lim} = 20$.



(a) $D_{lim} = 10$ (b) $D_{lim} = 20$

FIG. 4.11 – *Évolution de la moyenne des meilleures performances (moyenne sur 21 calculs indépendants).*



(a) $D_{lim} = 10$ (b) $D_{lim} = 20$

FIG. 4.12 – *Évolution de la moyenne des meilleures performances (moyenne sur 21 calculs indépendants).*

La conclusion qu'on peut déduire à partir de ces résultats expérimentaux est que, contrairement à ce qui était attendu, les sites inutiles sont en fait des sites utiles pour l'évolution dans la plupart des cas.

4.3.5 Un autre problème de cantilever

On considère toujours une plaque fixée sur son bord gauche et sur lequel on applique une force ponctuelle au milieu du bord opposé dirigée vers le bas, mais on modifie les dimensions du domaine par rapport au modèle précédent. Le domaine de référence est un

rectangle de dimension 2×1 (Figure 4.13-a). La figure 4.13-b montre une structure optimale obtenue, sur ce cas test, par C. Kane et M. Schoenauer [99] en utilisant la représentation par tableaux de bits.

(a) (b)

FIG. 4.13 – *(a) Problème de la plaque cantilever standard* 2×1. *(b) Une solution obtenue en utilisant la représentation bit-array pour un maillage* 32×22*(d'après [99])*

Différentes expériences numériques, faites sur le problème de la plaque console 2×1 discrétisée selon un maillage régulier 20×10, ont donné des structures de poids et de comportements mécaniques semblables, mais montrant de petites variations de formes. Les tracés de la figure 4.14 montrent deux de ces meilleures structures obtenues. Chacune de ces structures correspond à une valeur donnée de la contrainte sur le déplacement maximal ($D_{lim} = 220$ et $D_{lim} = 110$).

(a) $D_{lim} = 220$ (b) $D_{lim} = 110$

FIG. 4.14 – *Les deux meilleurs individus pour la représentation de Voronoï, maillage* 20×10. *Temps CPU* $\propto$ *1s/génération. (a) poids = 35%, 32 sites. (b) poids = 57%, 37 sites.*

La figure 4.15 illustre les solutions les plus significatives pour le même problème avec une discrétisation plus fine 32×22.

(a) $D_{lim} = 220$ (b) $D_{lim} = 110$

FIG. 4.15 – *Les deux meilleurs individus pour la représentation de Voronoï, maillage* 32×22. *Temps CPU = 3s/génération. (a) poids = 33%, 12 sites. (b) poids = 55%, 32 sites.*

4.3.6 Tableaux de bits et représentation de Voronoï

Ces premières expériences lancées avec la représentation par diagrammes de Voronoï nous ont servi à valider cette approche et à la comparer avec la représentation par tableaux de bits, en terme de qualité des solutions et aussi de coût en temps de calcul.

La figure 4.13-b (d'après [99]) montre un exemple typique d'un résultat obtenue sur le problème de cantilever 2×1 discrétisé selon un maillage 32×22, en utilisant la représentation par tableaux de bits. Un résultat sur un cas très semblable est présenté par la figure 4.15-a (quoique pour des raisons techniques, les conditions mécaniques exactes ne puissent pas être reproduites).

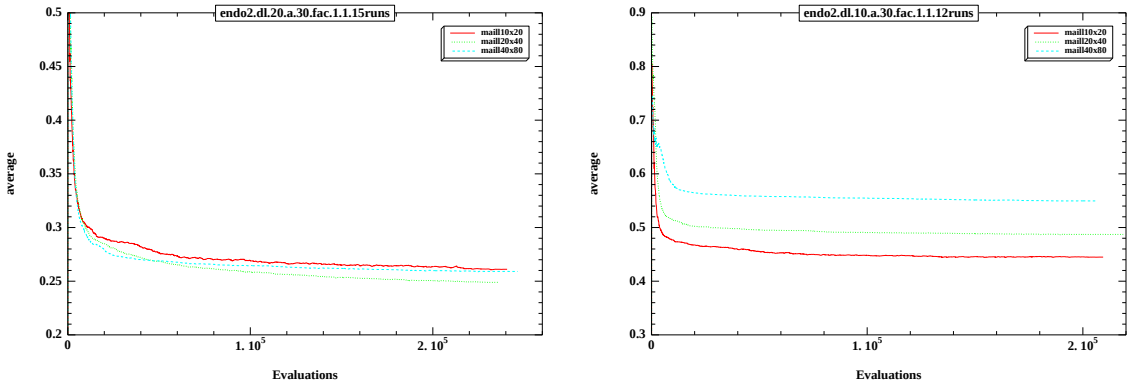
La différence principale entre les deux solutions est que la structure obtenue par notre approche ne présente pas les petits trous qui apparaissent dans la solution obtenue par la représentation par tableaux de bits. Cependant, nous savons que la solution optimale d'un problème d'optimisation topologique de formes posé dans l'espace relaxé est composé d'une infinité de trous de tailles infiniment petites (cf. chapitre 2). De ce point, il semble que l'approche bit-array essaye d'aller vers cette solution, limitée par le maillage utilisé. Par ailleurs, la représentation par diagrammes de Voronoï produit des formes beaucoup plus lisses, plus satisfaisantes de point de vue technologique : l'approche par tableaux de bits, manipulant chaque élément indépendamment, a besoin de très nombreuses générations simplement pour affiner la forme en fin d'évolution, et obtenir une frontière un peu régulière, alors que l'approche par diagrammes de Voronoï agit sur les interfaces entre les cellules, ajustant ainsi directement des frontières polygonales par morceaux. C'est ainsi que notre approche nécessite des temps de calcul beaucoup moins importants que l'approche booléenne pour obtenir des résultats analogues : d'après [99] un essai, en uti-

lisant la représentation par tableaux de bits, requiert environ 150000 analyses éléments finis, alors que notre approche nécessite en moyenne 130000 analyses éléments finis.

4.3.7 Dépendance par rapport au maillage

Pour vérifier l'indépendance des résultats de la représentation par diagrammes de Voronoï par rapport à la complexité du maillage, nous avons testé différentes discrétisations régulières pour le cantilever 1×2, avec les mêmes paramètres mécaniques et d'évolution.

Nous avons regardé les variations du nombre d'évaluations de la performance afin de déterminer si les raffinements de maillages modifiaient la vitesse de convergence, c'est-à-dire le nombre de calculs par éléments finis nécessaires pour atteindre une solution donnée (bien entendu le temps de calcul de chaque évaluation de la performance augmente avec la finesse de la discrétisation sans qu'il soit possible de l'éviter).



(a) $D_{lim} = 20$ (b) $D_{lim} = 10$

FIG. 4.16 – *Poids de la meilleure structure au cours de l'évolution (moyenne sur 21 calculs indépendants) pour trois finesses de maillages différentes du cantilever* 1×2.

Sur la figure 4.16 nous avons tracé l'évolution de la performance du meilleur individu (moyennée sur 21 calculs indépendants) pour des maillages 10×20, 20×40 et 40×80 et deux valeurs différentes de la contrainte sur le déplacement maximum ($D_{lim} = 10$ et $D_{lim} = 20$), le nombre d'individus par génération restant constant (80). Bien que la figure 4.16-a montre comme on l'attendait une parfaite indépendance de la vitesse de convergence par rapport à la discrétisation, la figure 4.16-b semble contredire cette hypothèse. Toutefois, un examen plus attentif permet d'avancer une explication : la meilleure solution obtenue pour le maillage 10×20 (poids = 44%, déplacement maximal = 9.97), une fois projetée sur le maillage plus fin 20×40 donne un poids de 43.12% et un déplacement maxi-

mal de 11.265 ! Ainsi, la grossièreté du maillage est responsable des différences entre les courbes – et comme la différence relative est plus importante pour les structures légères, cela explique en grande partie le premier graphique. Notons enfin que, même sur la figure 4.16-b, le comportement général de l'algorithme évolutionnaire est identique pour les 3 maillages, à la différence près de la différence entre les solutions finales. De plus, le nombre de sites de Voronoï pour décrire la meilleure solution sur les trois maillages est approximativement le même.

En conclusion, on peut dire qu'on a bien indépendance de la complexité de la représentation par diagrammes de Voronoï par rapport à celle de tout maillage.

4.3.8 La demi roue

Dans cette section nous considérons un autre exemple d'application dans le même cadre, avec des conditions aux limites différentes, dont la solution devrait être une roue. Le problème est représenté par la figure 4.17 (la symétrie du problème nous permet de travailler sur un quart du domaine réel). La structure est simplement supportée au point en bas à gauche (figure 4.17) et une force surfacique verticale est appliquée au coin droite de sa base. On suppose que tout point du bord droite est fixé en x et libre en y (conditions de symétrie par rapport à (oy)), et on impose que la structure soit connectée au point en haut à droite (le point C dans la figure).

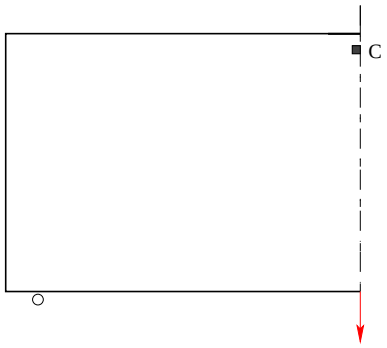


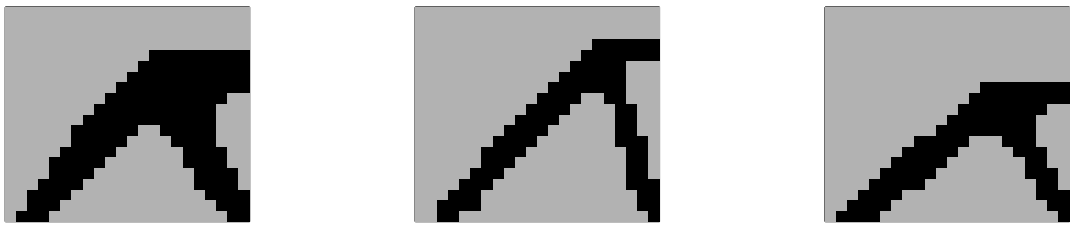
FIG. 4.17 – *Un quart du domaine et conditions aux limites*

La figure 4.18 montre les tracés de deux solutions optimales, obtenues en utilisant un maillage régulier 22×20, correspondant respectivement à deux valeurs différentes du déplacement limite imposé. Les deux structures respectent les contraintes de déplacement, et ressemblent à peu près à un quart de roue.

(a) $D_{lim} = 40$, poids = 35% (b) $D_{lim} = 60$, poids = 25%

FIG. 4.18 – *Deux solutions optimales quand on impose la connection au point C*

Les tracés de la figure 4.19 montrent les formes optimales obtenues pour les mêmes conditions que précédemment, mais sans imposer la contrainte de connectivité au point C (figure 4.17). On constate que l'algorithme arrive à trouver des structures optimales plus légères que celles de la figure 4.18, et qui respectent la contrainte de déplacement. Par contre, si on complète les solutions par symétrie on obtient des structures plus ovales que rondes. Mais rien dans le problème ne privilégie le rond, ce qui illustre la souplesse de l'approche évolutionnaire de trouver des solutions sans imposer la contrainte de connectivité.



(a) $D_{lim} = 40$, poids = 34% (b) $D_{lim} = 60$, poids = 22% (c) $D_{lim} = 60$, poids = 22%

FIG. 4.19 – *Résultats sans imposer la contrainte de connectivité. (b) et (c) sont deux solutions alternatives pour la même contrainte de déplacement.*

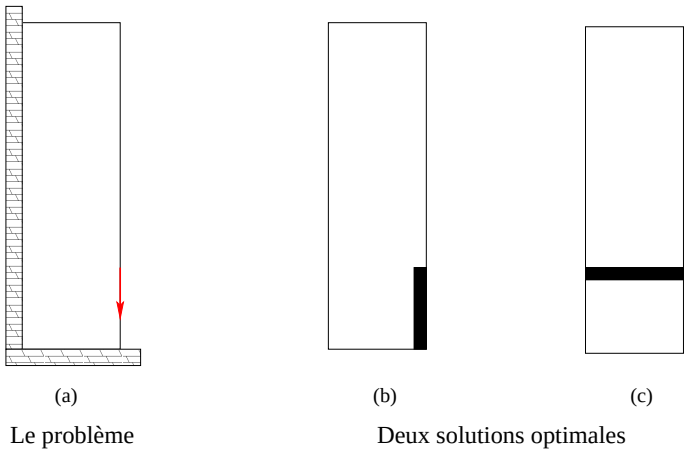


FIG. 4.20 – *Le problème de cantilever modifié* 1×4.

4.3.9 Solutions multiples

Le problème considéré dans cette section est une modification du problème de la plaque console standard afin d'assurer l'existence de plusieurs solutions quasi-optimales. Ce problème a été déjà testé par les algorithmes génétiques en utilisant la représentation par tableaux de bits, et les résultats obtenues illustrent la capacité des algorithmes évolutionnaires à trouver des solutions multiples [99] (voir figure 2.6 dans le chapitre 2).

Nous reprenons ce problème dans cette section pour valider notre approche. Le domaine de référence est un rectangle de dimension 1×4, discrétisé selon un maillage régulier 10×40, pour lequel les deux bords gauches et bas sont fixes (cf. figure 4.20-a). Selon la hauteur du point où la force est appliquée et suivant la contrainte de déplacement, le même problème peut avoir plusieurs solutions.

Un exemple simple de tels cas est donné dans la figure 4.20 (b) et (c) : si la force est appliquée au point de hauteur 1 et à condition que le déplacement limite est assez grand, les deux structures (b) et (c) sont deux solutions optimales. De plus, si la hauteur du point d'application de la force est fixée et que la contrainte de déplacement D_{lim} est graduellement relaxée, il existe différentes solutions quasi-optimales pour certaines valeurs de D_{lim}. Les tracés de la figure 4.21 montrent les résultats les plus significatifs obtenus, pour différentes valeurs de D_{lim}, avec une force appliquée au point de hauteur 1.5.

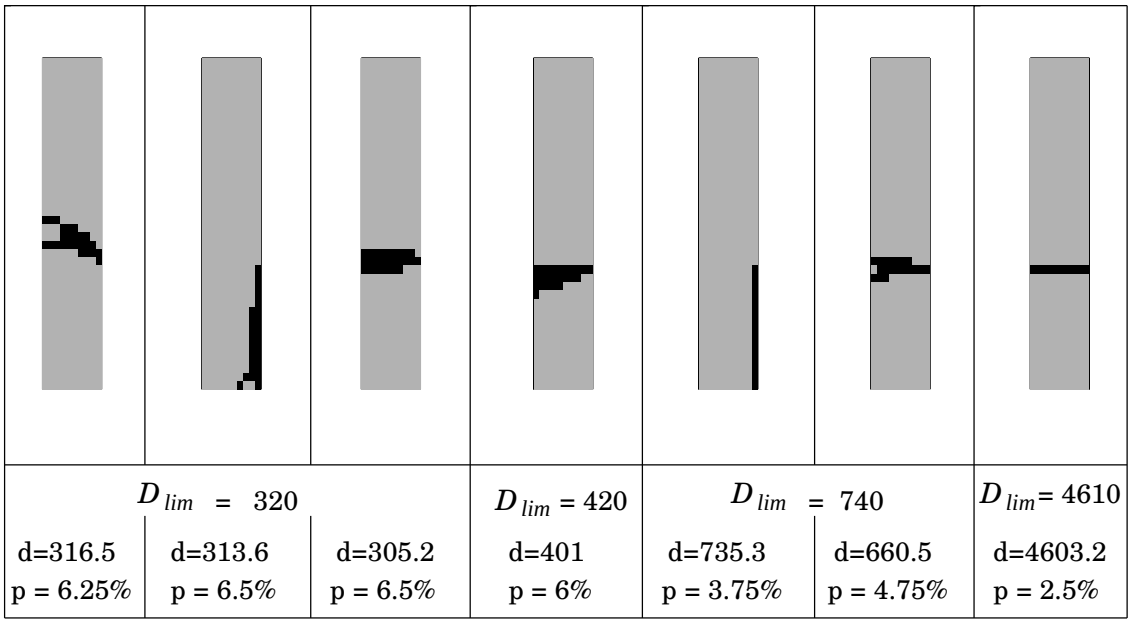


FIG. 4.21 – *Structures optimales pour le problème avec solutions multiples. La force est appliquée au point de hauteur 1.5. p désigne le poids et d désigne le déplacement maximal de la structure.*

4.3.10 Le cantilever 10×1

Le problème du cantilever 10×1 (figure 4.22) est délicat à traiter avec une représentation de type bit-array à cause d'une difficulté supplémentaire par rapport au cas précédent : la plupart des structures générées aléatoirement au cours du processus d'initialisation ne connectent pas le point d'application de la force avec la frontière encastrée. Une procédure d'initialisation particulière est utilisée, dans laquelle le poids moyen des structures aléatoires peut être contrôlé (cf. section 4.2.3 pour le principe de cette procédure). De plus, le nombre maximal de sites par individus a été augmenté à 120 et les meilleurs résultats sont obtenus pour des populations de 120 individus. La figure 4.22 montre un des résultats les plus significatifs pour une contrainte $D_{lim} = 12$, et un maillage régulier 100×10.

Les tracés de la figure 4.24 nous donnent l'évolution du meilleur individu (structure) obtenu au cours des générations pour l'essai qui a abouti à la figure 4.22 : pour les premières générations, la plupart des structures sont non connectées (le point d'application de la force n'est pas relié au bord bloqué); de la génération 10 et jusqu'à la génération 50 la meilleure structure trouvée est connectée mais la contrainte est violée; la génération 75

présente une structure respectant la contrainte ; à partir de cette génération les structures obtenues sont de plus en plus légères et la contrainte est toujours respectée jusqu'à la génération 1500 où la solution est optimale (pour ce test). Notons bien que le nombre de sites varie d'une structure à l'autre. À signaler que l'algorithme a continué à tourner jusqu'à la génération 1800 sans améliorer la solution.

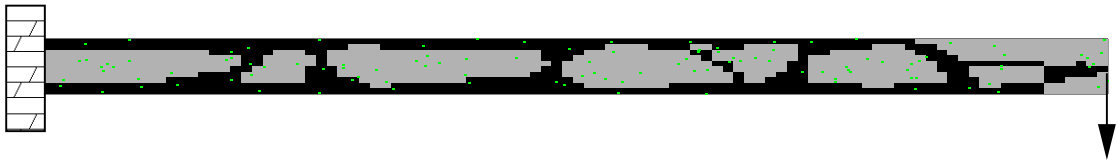


FIG. 4.22 – *Structure optimale sur le maillage* 100×10 *pour le cantilever* 10×1. $D_{lim} = 12$, $D_{max} = 11.98$, *poids* $= 44\%$. *Temps CPU = 5 à 6s/génération (120 individus).*

Une discrétisation plus fine 200×20 pour le même problème avec la même contrainte de déplacement nous donne, par exemple, comme résultat la figure 4.23.

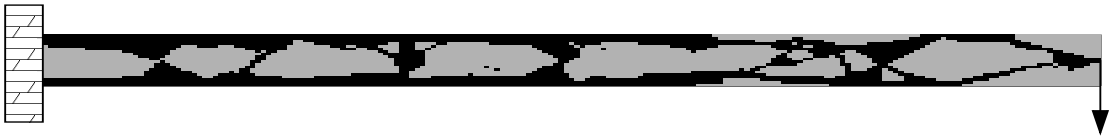


FIG. 4.23 – *Structure optimale sur le maillage* 200×20 *pour le cantilever* 10×1. $D_{lim} = 12$, $D_{max} = 11.99$, *poids* $= 44.5\%$. *Temps CPU = 28s/génération.*

Ces résultats montrent la robustesse de l'approche par diagrammes de Voronoï dans la résolution d'un problème hors de portée de l'approche par tableaux de bits.

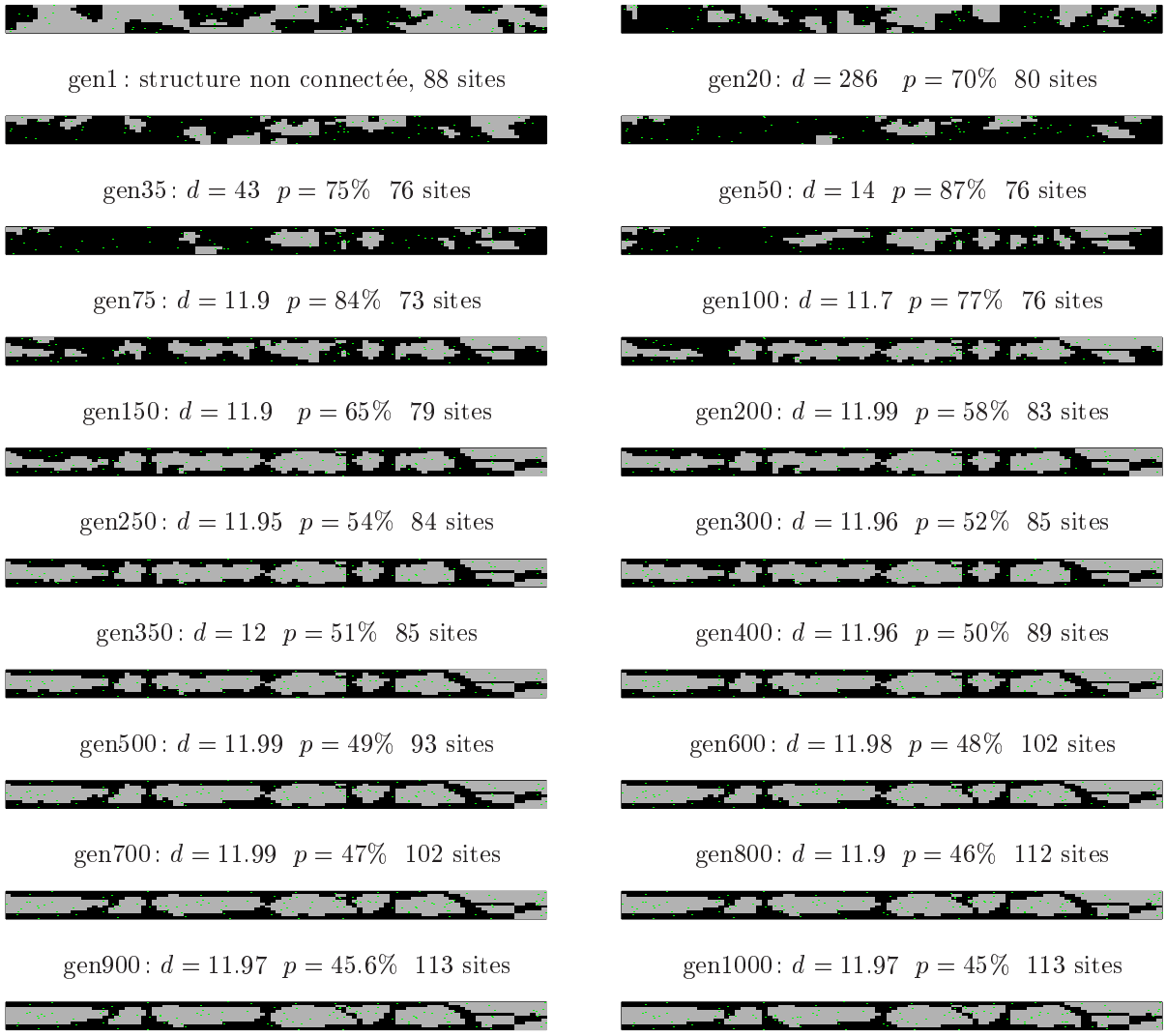


FIG. 4.24 – *Evolution de la meilleure structure au cours des générations. maillage* 100×10, $D_{lim} = 12$. *temps CPU = 5 à 6s/génération (120 individus). p désigne le poids et d désigne le déplacement maximal de la structure.*

4.3.11 Problèmes multi-chargements

Les problèmes considérés dans les parties précédentes ne mettaient en jeu qu'un chargement unique. Or il est clair que les structures sont en général construites pour plusieurs utilisations possibles, et doivent par conséquent être optimisées pour plusieurs cas de charges : il est fréquent, dans la pratique de l'ingénieur, de devoir effectuer la conception d'un composant mécanique sollicité de plusieurs manières fondamentalement différentes.

Modification de la fonction d'adaptation

La prise en compte de cette nouvelle situation est très simple dans l'approche évolutionnaire : comme il a été indiqué dans le chapitre 3, pour chaque structure, une simulation numérique est effectuée pour chacun des cas de chargement, et la fonction de performance est pénalisée par la somme des violations de contraintes

$$F = P_{util} + \epsilon P_{inutile} + \sum_{j=1}^{m} \alpha^j \max(0, D^j_{max} - D^j_{lim})$$

où P_{util} est la surface de la structure principale (la composante recevant le chargement) et $P_{inutile}$ est la surface totale des composantes matérielles non connectées à la structure principale, et ϵ est un (petit) paramètre positif. D^j_{lim} est la valeur limite du déplacement maximal pour le cas de charge j, D^j_{Max} est le déplacement maximal de la structure soumise au chargement j, et α^j est le facteur de pénalisation de la j^{eme} contrainte. Les paramètres α^j varient au cours des générations suivant la méthode adaptative introduite dans la section 3.5.5 du chapitre précédent. Dans la suite, nous considérons un seul paramètre de pénalisation pour toutes les contraintes.

Bien entendu, l'évaluation de la fonction performance d'une structure requiert m analyses par éléments finis (1 décomposition LU + m descentes/remontées), et augmente donc le coût de calcul de la méthode.

Le cadre de vélo

Une situation où l'optimisation d'une structure doit certainement s'effectuer en considérant plusieurs cas de chargement est le cas du cadre de vélo : le cycliste pédalant sur son vélo sollicite le cadre de celui-ci de plusieurs manières très différentes suivant les conditions de terrain (plat, montée, descente), la vitesse à laquelle il souhaite se déplacer, le nombre de passagers qu'il transporte, ...

Ce problème de vélo a été étudié par Rasmussen et Olhoff (1992) [136], avec la méthode

d'homogénéisation et a été repris par Kikuchi et al. [34], par Allaire et Jouve [1], et plus récemment, en utilisant les algorithmes évolutionnaire, par Kane et Schoenauer [100].

Comme dans [100], pour des raisons de simplicité, seuls trois cas ont été envisagés : sur route plate, la plus grande force est appliquée sur la selle; en montée modérée, l'effort sur les pédales est plus important, et le cycliste tire sur le guidon; et enfin, en position de "danseur" le cycliste ne touche plus la selle.

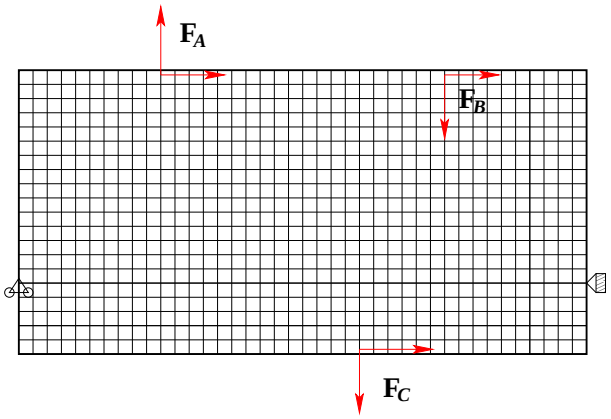


FIG. 4.25 – *domaine de travail et chargement.*

Domaine de travail et chargements

Une formalisation possible du problème est donné par la figure 4.25 : le domaine est un rectangle 2×1 discrétisé suivant un maillage régulier 40×20. les conditions de chargement sont indiquées par des flèches : nous avons considéré trois cas de chargements appliqués au points A, B et C de coordonnées respectives (0.5,1), (1.5,1) *et* (1.2,0). A désigne la position du guidon, B la position de la selle, et C la position du pédalier. Les trois cas de charge sont donnés par :

`chargement`	F_A	F_B	F_C
1	$(-0.05,1)$	$(0.15,-1)$	$(0,-1)$
2	$(-0.2,1)$	$(0,0)$	$(0.2,-2)$
3	$(-0.6,0.7)$	$(0.3,-0.5)$	$(0.5,-1)$

Le chargement 1 correspond à la position classique, le chargement 2 est la position du "danseur", le troisième chargement représente la position "en montée".

Résultats obtenus

On considère une population de taille 120, et le nombre maximal de sites par individu est fixé à 80. La figure 4.26 montre les solutions obtenues pour chaque position du cycliste. Les différences entre les trois structures correspondant respectivement à chaque position du cycliste nous amène à considérer tous les cas de chargement auxquels la structure risque d'être soumise. La figure 4.27 montre la structure optimale du cadre de vélo, en appliquant successivement les 3 cas de chargement.

(a) (b) (c)

FIG. 4.26 – *Les trois structures optimales pour les trois cas correspondant : (a) la position classique. (b) la position du danseur. (c) la position de la montée*

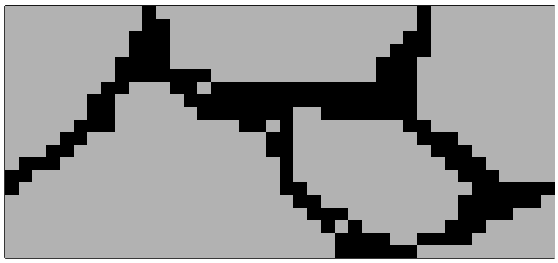


FIG. 4.27 – *Le cadre de vélo optimisé pour les trois cas appliqués successivement.*

Le modèle "L"

Le domaine de travail a la forme d'un "L" inversé contenant 300 mailles. Son bord gauche est fixé par des conditions de blocages et on exerce une charge ponctuelle au milieu du bord supérieur (cf. figure 4.28). Deux différents cas de charges ont été considérés : $F = (-1,1)$ pour le premier cas, et $F = (1,1)$ pour le deuxième cas.
Pour ce problème le nombre maximal de sites par individu a été fixé à 60, et la taille de

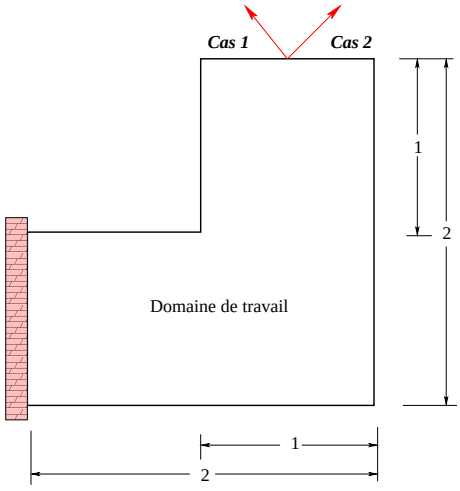
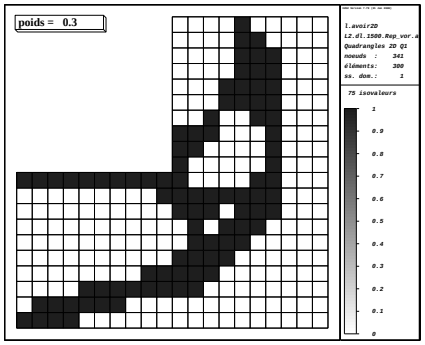


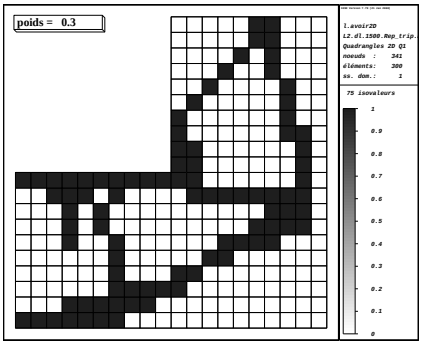
FIG. 4.28 – *Domaine de travail et chargement.*

la population est égale à 100 individus.

Les figures 4.29 et 4.30 montrent les structures optimales obtenues, en appliquant le premier cas de charge, pour deux valeurs différentes de déplacement limite D_{lim}. En considérant le deuxième cas de charge la solution optimale obtenue, pour une valeur limite donnée de D_{lim}, est illustrée dans la figure 4.31.



(a) poids = 30%, 52 sites.



(b) poids = 30%, 36 sites.

FIG. 4.29 – *Deux structures optimales pour le premier cas de charge.*

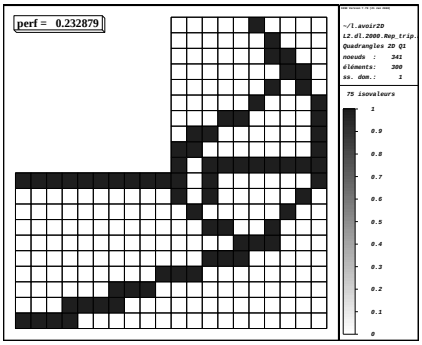


FIG. 4.30 – *Une structure optimale pour le premier cas de charge, avec une contrainte plus relaxée sur le déplacement que celle imposée pour les structures de la figure 4.29. poids = 23%, 23 sites*

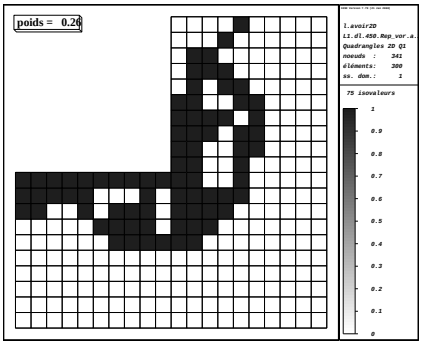


FIG. 4.31 – *Une structure optimale pour le premier cas de charge. poids = 26%, 46 sites*

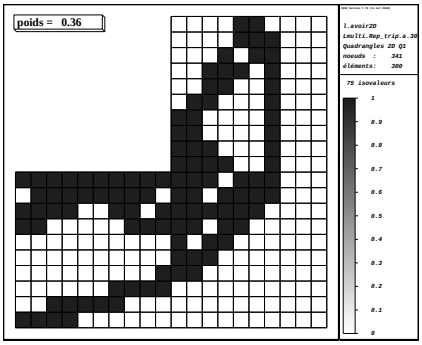


FIG. 4.32 – *Modèle "L" multi-chargements. Une structures optimale. poids = 36%, 24 sites*

Pour le problème multi-chargements (en appliquant les deux cas de charges successivement), une solution optimale respectant les contraintes de déplacement est représentée par la figure 4.32.

Les structures obtenues pour le modèle "L" présentent de petits trous éparpillés, ce qui indique que notre algorithme tente en quelque sorte de se rapprocher de la solution dans l'espace homogénéisé!

Le pont 2D

Les résultats suivants montrent quelques solutions obtenues dans les cas mono-chargement et multi-chargements pour le même exemple numérique 2D.

L'exemple proposé présente une symétrie permettant de ne considérer que la moitié du domaine et la solution finale sera obtenue par symétrisation du domaine. Le domaine de travail est un rectangle formé de 41×21 points. La structure est supposée bloquée au coin A et soumise à trois cas de charges, montrés sur la figure 4.33, correspondant à trois forces ponctuelles appliquées successivement.

Pour le problème mono-chargement, nous avons considéré le chargement constitué par les trois forces, de la figure 4.33, d'intensités égales appliquées en même temps.

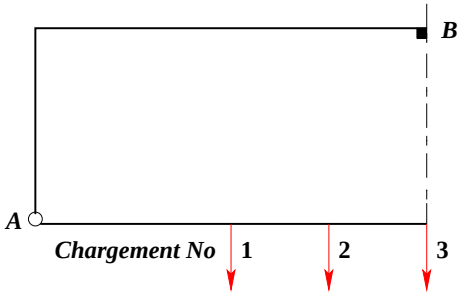
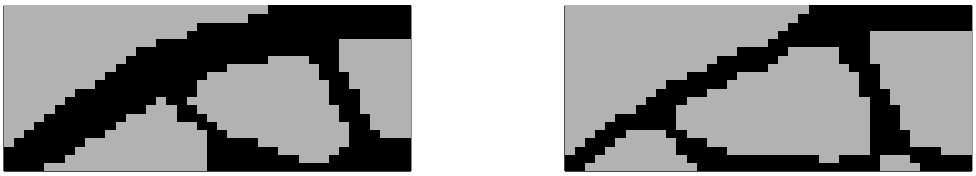


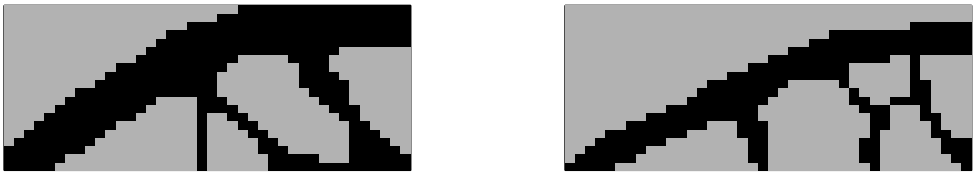
FIG. 4.33 – *Pont2D - Les cas de charges pour l'optimisation en multi-chargements.*

Plusieurs tests ont été effectués pour les deux problèmes mono-chargement et multi-chargements. Les paramètres suivants ont été utilisés : une population de taille 120, le nombre de sites maximal est fixé à 80. Deux valeurs de D_{lim} ont été prises 200 et 350. Les structures présentées ici respectent toutes la contrainte de déplacement.



(a) $D_{lim} = 200 \quad poids = 42\%$ (b) $D_{lim} = 350 \quad poids = 30\%$

FIG. 4.34 – *Pont2D mono-chargement - Deux structures optimales pour deux contraintes de déplacement : en imposant une contrainte de connectivité*



(a) $D_{lim} = 200 \quad poids = 43\%$ (b) $D_{lim} = 350 \quad poids = 29\%$

FIG. 4.35 – *Pont2D mono-chargement - Deux structures optimales pour deux contraintes de déplacement : sans imposer la contrainte de connectivité*

Dans les premières expériences numériques nous avons imposé la contrainte de connectivité suivante : la structure doit être connectée au point B (figure 4.33). Les solutions optimales (les plus significatives) obtenues pour les deux valeurs de D_{lim} sont données par la figure 4.34.

La figure 4.35 montre les structures optimales (toujours les plus significatives) obtenues pour les mêmes contraintes de déplacement, mais sans imposer la contrainte de connectivité. On remarque que la première structure est bien connectée au coin droite en haut, alors que la deuxième structure est connectée au bord droit mais pas au coin. C'est le processus d'optimisation lui-même qui "choisit" où fixer la structure le long de la frontière verticale droite. Les résultats obtenus montrent une autre fois la flexibilité de l'approche de trouver plusieurs solutions pour la même contrainte.

Pour le problème multi-chargements la solution optimale obtenue est représentée par la figure 4.36.

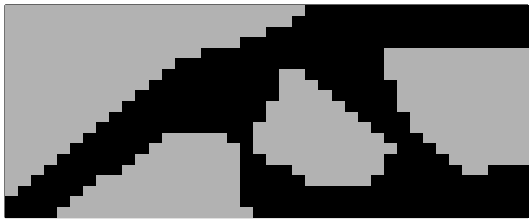


FIG. 4.36 – *Pont2D multi-chargements - Une solution optimale.* $poids = 44\%$

Conclusion

Dans cette section, nous avons présenté des résultats mettant en jeu l'optimisation des formes soumises à plusieurs chargements sur 3 problèmes différents. A noter qu'une autre approche possible pour ces mêmes problèmes est l'optimisation multi-objectifs (cf. chapitre 6) : au lieu d'agréger tous les cas de charge dans une seule fonction objectif, l'idée consiste à considérer chaque cas comme un objectif et d'essayer d'identifier l'ensemble des meilleurs compromis possible entre les objectifs. Ce qui permettra beaucoup plus de flexibilité dans la conception finale. A notre connaissance, une telle approche n'a pas encore été utilisée dans le cadre de l'optimisation topologique multi-chargements des formes.

4.3.12 Un problème tridimensionnel

Nous présentons dans cette section les premiers (à notre connaissance) résultats 3D en optimisation topologique de formes par algorithmes évolutionnaires.

Le domaine de travail est un parallélépipède rectangle et le problème présente une symétrie permettant de ne discrétiser que la moitié du domaine par un maillage à $16\times7\times10$ éléments. Le plan en $x = 0$ est encastré, et une force verticale est appliquée au milieu de la face opposée (cf. figure 4.37).

Pour ce problème plus complexe que les cas précédents, quelques paramètres doivent être modifiés : la taille de la population est de 120 individus et le nombre maximum de sites de Voronoï par individu vaut 120.

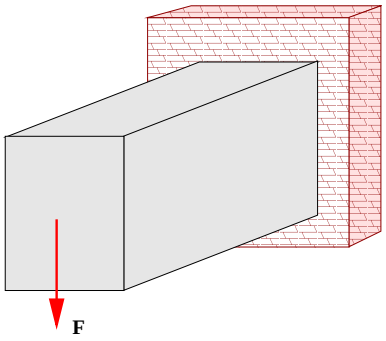
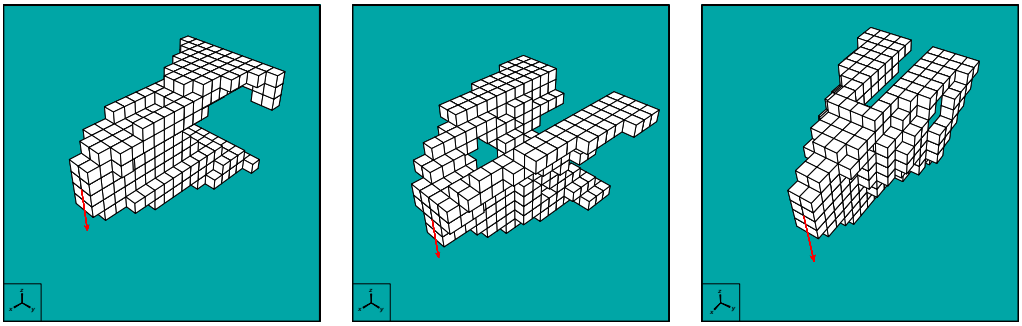


FIG. 4.37 – *Le cantilever tridimensionnel*

La figure 4.38 montre que l'algorithme a été capable de trouver quelques solutions acceptables . . . en quelques jours de calcul (les analyses par éléments finis de problèmes 3D sont beaucoup plus coûteuses que pour les cas 2D à nombre de mailles égal). De plus, elle démontre la capacité connue des algorithmes évolutionnaires de trouver des solutions quasi-optimales multiples pour un même problème, certaines étant assez originales par rapport à celle obtenue par la méthode d'homogénéisation [2].



(a) (b) (c)

FIG. 4.38 – *Trois résultats pour le cantilever tridimensionnel, avec la même contrainte sur le déplacement maximal. Temps CPU = 4 à 5mn/génération (120 individus). (a) poids = 15.2%, 103 sites (b) poids = 16%, 109 sites (c) poids = 15.7%, 112 sites*

4.3.13 Conclusion

Les résultats obtenus ont permis de repousser les limites de l'optimisation topologique de formes évolutionnaire: la représentation par diagrammes de Voronoï a montrée sa puissance et sa robustesse par rapport à l'approche par tableaux de bits.

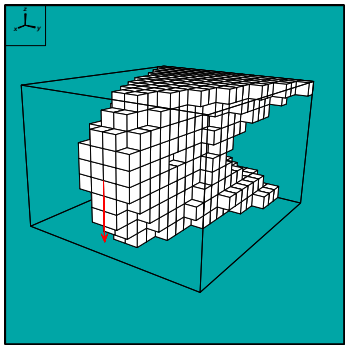


FIG. 4.39 – *Une Solution 3D pour le même problème avec une contrainte plus forte,* $D_{lim} = 10$. *poids = 32.4%, 75 sites*

Cependant, un point important en optimisation topologique est l'ajustement fin des frontières de la solution. Une forme optimale ne peut être atteinte en un temps de calcul raisonnable que si l'algorithme est capable de contrôler précisément les frontières des individus d'une population. Or, la représentation Voronoï ne permet qu'un contrôle indirect de la frontière d'un individu. De plus, par sa structure même, cette représentation rend difficile la modification d'une portion de la frontière sans toucher aux portions adjacentes. Pour palier cette difficulté, nous proposons dans la suite deux autres représentations.

4.4 La représentation par dipôles

Nous introduisons dans cette section une deuxième représentation basée sur les diagrammes de Voronoï, que nous avons nommée représentation par *dipôles*, ainsi que les opérateurs de variation associés.

4.4.1 Dipôles

Un dipôle est un ensemble de deux sites de Voronoï, l'un étiqueté 0 et l'autre 1, situés au même point du domaine de calcul, avec une médiatrice associée repérée par son angle directeur. Un dipôle est donc défini en 2 dimensions d'espace par trois variables réelles : ses coordonnées (x,y) et l'angle θ de sa médiatrice avec l'axe des abscisses. La figure 4.40-a est un exemple de dipôle.

4.4.2 Le génotype

Un individu dans la représentation dipolaire est une liste – de longueur variable – de dipôles. Comme dans la représentation Voronoï, le diagramme de Voronoï correspondant

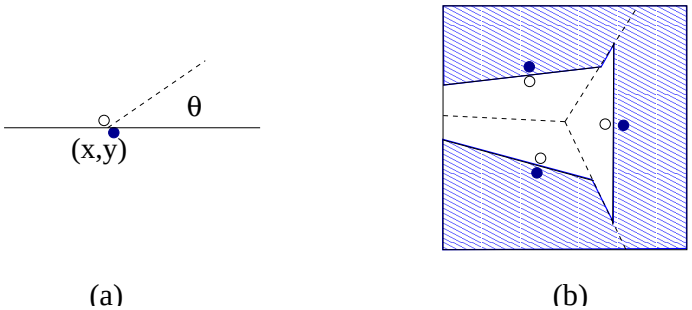


FIG. 4.40 – *La représentation par dipôles. Un seul dipôle (a) et le diagramme de Voronoï construit à l'aide de trois dipôles (b) : des coins indésirables apparaissent aux croisements des médiatrices.*

donne une partition du domaine de travail en deux sous-ensembles.

4.4.3 Décodage

Comme dans la représentation Voronoï, la performance des individus est évaluée sur un maillage fixe, et la projection sur ce maillage s'effectue comme dans la section 4.2.

Toutefois, comme on peut le voir sur la figure 4.40-b, le décodage de deux dipôles adjacents montre que la structure qui en résulte possède deux sortes de frontières : la médiatrice des dipôles, qui peut être contrôlée directement par l'algorithme; et la médiatrice entre deux dipôles, dont le contrôle est aussi difficile que dans la représentation Voronoï. Il peut même arriver qu'interviennent des situations compliquées, comme celles de la figure 4.40-b, où le contrôle est encore plus délicat.

4.4.4 Opérateurs de variations

Ils sont dérivés de ceux de la représentation de Voronoï: la procédure d'initialisation choisit un nombre de dipôles et initialise leurs coordonnées et angles uniformément dans le domaine de travail $\times[0,2\pi]$. L'opérateur de croisement échange les dipôles exactement comme les sites dans le cas de la représentation de Voronoï (cf. figure 4.4). Les opérateurs de mutation sont également similaires avec une possibilité supplémentaire de mutation Gaussienne sur l'angle d'un dipôle qui consiste à ajouter une quantité aléatoire selon une distribution normale $N(0,1)$:

$$\begin{cases} \sigma_i := \sigma_i exp(\tau' N(0,1) + \tau N_i(0,1)) \\ \theta_i := \theta_i + \sigma_i N(0,1) \end{cases}$$

où σ_i est la déviation standard attachée à l'angle θ_i.

4.5 La représentation par barres de Voronoï

Nous introduisons dans cette section une troisième représentation basée aussi sur les diagrammes de Voronoï, nommée représentation par *barres* de Voronoï, ainsi que les opérateurs de variation associés.

4.5.1 Structures en treillis

Pour les problèmes de cantilever, en 2D, il est connu que les formes optimales ont une structure en treillis [119]. L'obtention de telles structures par des diagrammes de Voronoï ou des dipôles requière l'émergence de sous-ensembles de sites ou de dipôles couplés. L'évolution vers des situations de ce type peut demander de très nombreuses générations. L'idée de la représentation par barres de Voronoï est d'aider à l'apparition de structures de ce genre en introduisant des formes élémentaires qui sont déjà des barres.

4.5.2 Les barres de Voronoï

Une barre de Voronoï est définie par quatre variables réelles : ses coordonnées (x,y), l'angle θ de la barre avec l'axe x et sa largeur. La figure 4.41-a montre un exemple d'une barre de Voronoï.

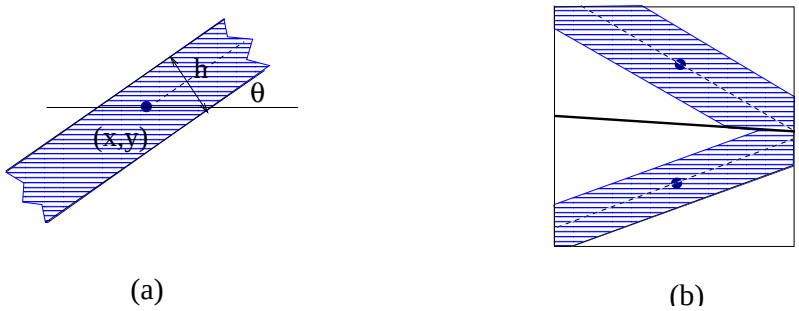


FIG. 4.41 – *La représentation par barres de Voronoï. Une seule barre (a) et la structure générée par deux barres(b) : le trait plus épais est la frontière entre les deux cellules de Voronoï. Elle ne fait partie de la structure qu'à la jonction entre les deux barres.*

4.5.3 Le génotype

Un individu dans la représentation par barres de Voronoï est une liste de taille variable de barres. Quand toutes les barres sont simplement vues comme des sites de Voronoï, le diagramme correspondant donne une partition du domaine en polygones convexes. Chaque polygone est alors séparé en deux sous-domaines : la partie centrale – définie par l'angle et la largeur – contient de la matière, et son complémentaire contient du vide (cf. figure 4.41).

Lorsque la largeur est assez grande, la totalité de la cellule est remplie de matière, tandis qu'une valeur nulle de la largeur donne une cellule vide.

4.5.4 Décodage

Comme pour la représentation Voronoï, la performance des individus est évaluée après projection sur un maillage fixe telle qu'elle est décrite dans la section 4.2 : un élément du maillage est considéré comme plein de matière si et seulement si son centre de gravité est dans la partie pleine d'une barre de Voronoï.

Comme on peut le voir sur la figure 4.41-b, le décodage de barres de Voronoï adjacentes permet le contrôle direct de presque toute la frontière de la structure, à l'exception des quelques portions limitées à la jonction de deux barres.

4.5.5 Opérateurs de variations

Ils sont de nouveau dérivés de ceux de la représentation de Voronoï : la procédure d'initialisation choisit un nombre de barres et initialise leurs coordonnées, angles et largeurs uniformément. L'opérateur de croisement échange les barres exactement comme les sites dans le cas de la représentation de Voronoï (cf. figure 4.4). Les opérateurs de mutation sont également similaires avec des possibilités supplémentaires de mutation Gaussienne sur l'angle et la largeur de la barre :

$$\begin{cases} \sigma_\theta^i := \sigma_\theta^i exp(\tau'_\theta N(0,1) + \tau_\theta N_i(0,1)) \\ \theta_i := \theta_i + \sigma_\theta^i N(0,1) \\ \sigma_h^i := \sigma_h^i exp(\tau'_h N(0,1) + \tau_h N_i(0,1)) \\ h_i := h_i + \sigma_h^i N(0,1) \end{cases}$$

où σ_θ^i et σ_h^i sont les déviations standards attachées respectivement à l'angle θ_i et la largeur h de la barre.

4.6 Résultats comparatifs

Nous présentons dans cette section les résultats obtenus avec les deux représentations non structurées (dipôles et barres) décrites plus haut sur des benchmarks classiques : nous avons considéré des cantilevers de dimensions 1×2 et 2×1 avec des limites respectives sur le déplacement maximal de 20 et 220. Les paramètres numériques sont ceux indiqués dans la section 4.3.1.

Les figures 4.42 et 4.43 montrent les meilleures structures obtenues avec les deux représentations. Nous rappelons que pour les mêmes cas test les solutions obtenues en utilisant la représentation de Voronoï sont données respectivement par les figures 4.9-a de la section 4.3.3 et 4.14-a de la section 4.3.5.

Les figures 4.44 et 4.45 montrent des statistiques sur 21 calculs indépendants, sur les deux cas test, pour les trois représentations à base de diagrammes de Voronoï (Voronoï, dipôles et barres).

(a) : poids=215%, 15 dipôles (b) : poids=32.5%, 36 dipôles

FIG. 4.42 – *Les deux meilleurs individus pour la représentation par dipôles.*

(a) : poids=20%, 4 barres (b) : poids=33%, 20 barres

FIG. 4.43 – *Les deux meilleurs individus pour la représentation par barres de Voronoï.*

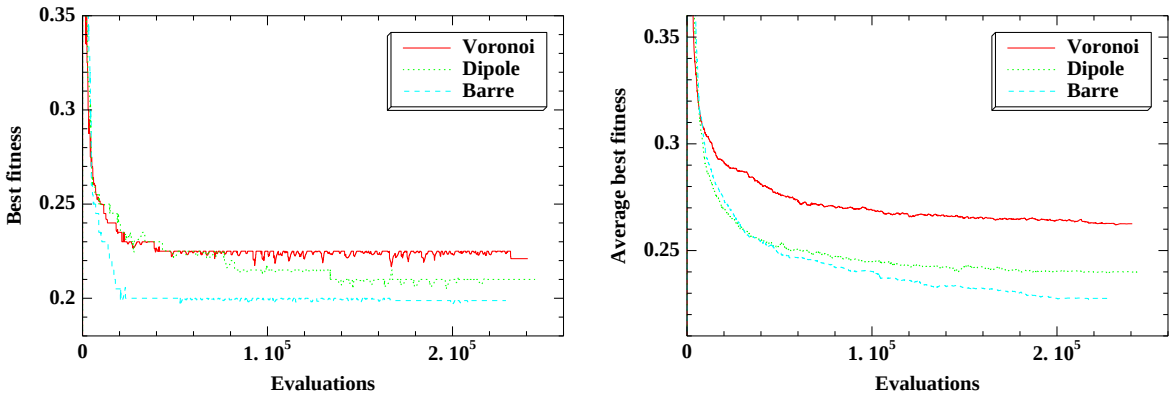
La première conclusion de ces expériences numériques est que les deux représentations,

comme pour la représentation de Vorornoï, permettent toutes de trouver des solutions aussi bonnes sur les 21 calculs. Toutefois, pour ces deux cas test, la meilleure représentation est sans conteste la représentation par barres de Voronoï : avec elle, presque toutes les solutions parmi les 21 calculs indépendants sont identiques à celle de la figure 4.43, tandis que de nombreuses solutions obtenues par la représentation par dipôles sont plus mauvaises, et encore plus avec la représentation Voronoï. Ces tendances sont visibles sur les comparaisons des figures 4.44 et 4.45.

Notons que les deux autres représentations permettent parfois de trouver des solutions comparables à celles obtenues avec les barres de Voronoï, mais cette dernière s'avère beaucoup plus robuste.

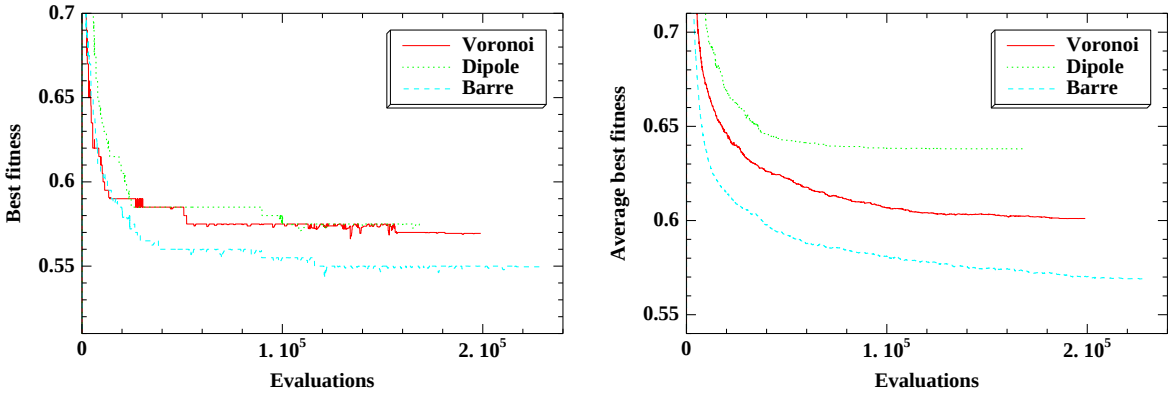
Un autre critère est la complexité des solutions. Les solutions attendues pour ces cas test sont très simples et la représentation devrait refléter cette simplicité. Sur ce point aussi, la représentation par barres de Voronoï trouve des représentations très compactes comparées aux autres. Un des calculs a même obtenu la structure parfaite formée de deux barres en $\mathbf{V}$ pour le cantilever 1×2.

Il semble donc que l'information de haut niveau supplémentaire introduite dans les "gènes" élémentaires de la représentation par barres de Voronoï soit efficace, au moins sur ces deux benchmarks.



(a) (b)

FIG. 4.44 – *Représentations à base de diagrammes de Voronoï sur le cantilever* 1×2 *pour Dlim = 20.*



Meilleur Moyenne

FIG. 4.45 – *Représentations à base de diagrammes de Voronoï sur le cantilever* 2×1 *pour Dlim = 220.*

Remarque

Il ne faut pas perdre de vue que la représentation par barres de Voronoï semble taillée sur mesure pour représenter des structure en treillis de barres – et que nous n'avons comparé les représentations que sur des problèmes oú la solution est précisément un treillis. Il convient donc de ne pas généraliser hâtivement ce résultat !

4.7 Vers une représentation modulaire

Certains des résultats les plus impressionnants de calcul évolutionnaire sont basés sur les représentations avancées, qui incluent l'identification de sous-domaine et la réutilisation [105].

Dans le contexte de l'optimisation topologique de formes, quand on regarde les résultats obtenus pour le problème du cantilever 10×1 (voir par exemple les figure 4.22 et 4.23), ou quand on considère les solutions technologiques existantes (la structure en treillis pour les longs bras de grue par exemple), il est clair que certains degrés de modularité pourraient être bénéfiques aussi bien pour la qualité des résultats que pour l'efficacité de la recherche, puisque cela supprimerait le besoin de l'algorithme de découvrir la même composante utile plusieurs fois et permettrait ainsi un gain supplémentaire en complexité.

Avant d'aller vers des représentations totalement modulaires, cette section présente les premiers résultats obtenus en utilisant une représentation permettant la réutilisation d'une

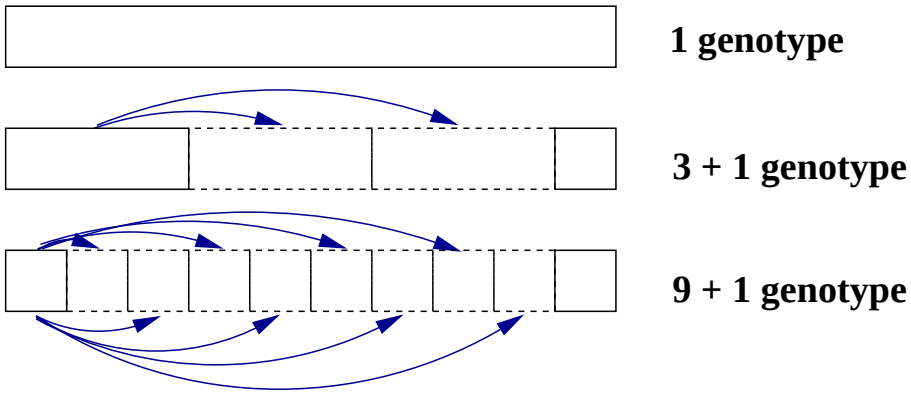


FIG. 4.46 – *représentations modulaires*

partie de la structure : plutôt que d'essayer de trouver toute la structure comme un diagramme de Voronoï, ce qui a été fait en utilisant la représentation à base de diagrammes de Voronoï pour tous les résultats discutés dans les précédentes sections, le domaine 10×1 est divisé en 4 ou 10 sous-domaines, et deux diagrammes de Voronoï sont utilisés pour représenter toute la structure : une première composante interne, qui sera réutilisée 3 ou 9 fois, et une composante de terminaison qui sera placée à l'extrême droite du domaine (cf. figure 4.46). Rappelons que la structure est fixée à sa frontière gauche. La réutilisation de la composante interne est forcée ici dans la représentation, alors que, bien entendu, un nouveau pas devra développer la modularité.

Les premiers résultats sont très encourageants. En effet, comme en témoignent les figures 4.47 et 4.48 (respectivement les figures 4.49 et 4.50), à comparer avec le résultat de la figure 4.22 (respectivement la figure 4.23) obtenu avec la représentation de Voronoï "ordinaire" : les structures résultantes sont de qualités équivalentes (même comportement mécanique et même poids), pour un temps de calcul total presque identique. Mais plus important, le nombre de sites nécessaires pour représenter les solutions modulaires des figures 4.47 et 4.48 est beaucoup plus petit que les 113 sites nécessaires pour représenter la structure de la figure 4.22 : entre 30 et 40 sites seulement sont nécessaires pour chaque composante des solutions $(3 + 1)$ (moins de 80 en total), et entre 20 et 30 pour celles de la représentation $(9 + 1)$ (moins de 60 sites en total). Ceci est un signe certain que des améliorations sont possibles, puisque moins on aura de sites à prendre en compte, plus facile sera l'ajustement fin de la solution.

Les courbes de la figure 4.51 montrent des statistiques sur 21 calculs indépendants pour les différentes représentations (voronoï, barres et modulaires)

FIG. 4.47 – *Meilleure solution pour la représentation* $(3+1)$, *avec un maillage* 100×10, *poids = 43.7%,* D_{max} *= 11.91.*

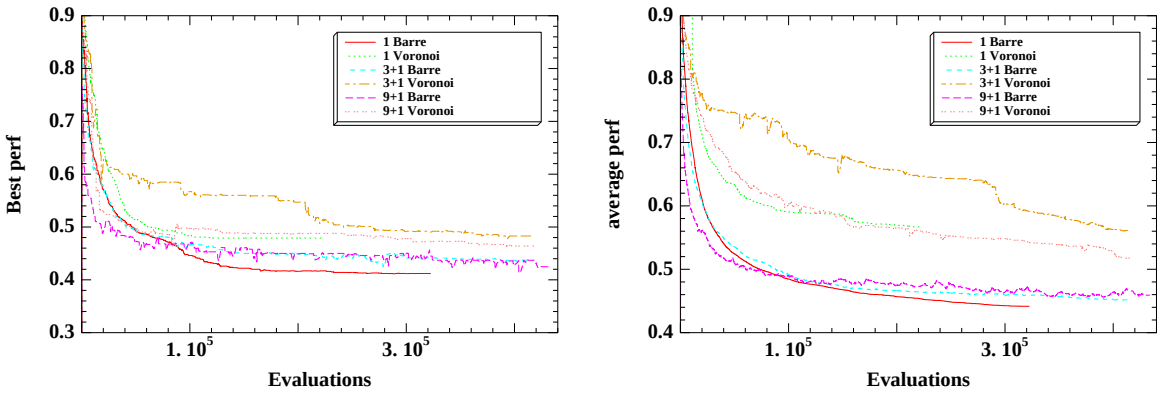
FIG. 4.48 – *Meilleure solution pour la représentation* $(9+1)$, *avec un maillage* 100×10, *poids = 44.5%,* D_{max} *= 11.92.*

FIG. 4.49 – *Meilleure solution pour la représentation* $(3+1)$, *avec un maillage* 200×20, *poids = 42.8%,* D_{max} *= 11.98.*

FIG. 4.50 – *Meilleure solution pour la représentation* $(9+1)$, *avec un maillage* 200×20, *poids = 43.2%,* D_{max} *= 11.99.*

4.8 Conclusion

Nous avons introduit de nouvelles représentations pour le traitement de problèmes d'optimisation topologique de formes par algorithmes évolutionnaires. En partant de la représentation binaire liée à un maillage donné du domaine de calcul, des représentations basées sur la théorie des diagrammes de Voronoï ont été proposées, depuis la représentation Voronoï simple jusqu'aux représentations plus complexes par dipôles ou barres de Voronoï. Ces trois représentations sont non structurées, c'est-à-dire qu'un individu est caractérisé par une liste de "gènes" non ordonnée et de longueur variable. La complexité de la structure d'un gène élémentaire augmente de la représentation Voronoï jusqu'à la représentation



Meilleur Moyenne

FIG. 4.51 – *Comparaison des différentes représentations. Meilleure des meilleures performances (a) et moyenne des meilleures (b) au cours de l'évolution (moyenne sur 21 calculs indépendants) sur le problème* 10×1

par barres de Voronoï. Toutefois, ces trois types de représentations permettent d'obtenir des méthodes où la complexité de la solution est auto-adaptative, i.e. pour lesquelles le nombre de variables effective décrivant un individu évolue à travers l'algorithme.

Les résultats obtenus ont permis de repousser les limites de l'optimisation topologique de formes évolutionnaire. En effet, l'approche basée sur les diagrammes de Voronoï a montré sa puissance et sa robustesse par rapport à l'approche par tableaux de bits : elle nous a permis de résoudre des cas de figure hors de portée de l'approche par tableaux de bits (par exemple le problème de cantilever élancé 10×1); de plus, des résultats originaux ont été obtenus sur un problème tridimensionnel. Ces derniers confirment la capacité des algorithmes évolutionnaires à trouver plusieurs solutions quasi-optimales au problème posé.

Par ailleurs, nous avons testé ces représentations sur des problèmes test simples d'optimisation topologique de formes. Les résultats semblent montrer que les trois représentations sont adaptées à la résolution de ce type de problèmes. Elles demandent grossièrement le même temps de calcul pour atteindre des solutions comparables, avec un léger avantage pour la représentation par barres de Voronoï. Pourtant, après examen de la complexité du codage des solutions obtenues, on trouve un avantage net pour la représentation par barres de Voronoï, dont les solutions utilisent moins de "gènes" que les deux autres. Cela explique sans doute le très léger avantage observé pour ce codage en terme de rapport qualité/coût de calcul. En effet, il est plus facile d'ajuster finement la solution lorsque peu de gènes sont en jeu.

Enfin, nous avons proposé des représentations modulaires, permettant la réutilisation d'une partie de la représentation, sur un problème de cantilever 10×1. Les premiers résultats montrent la performance de ces approches surtout au niveau complexité des individus : elles nécessitent un nombre de sites beaucoup plus petit que les autres représentations. Cependant, cette approche présente des inconvénients : en effet, cette façon de faire "manuelle" est imprécise et nécessite la présence d'un expert pour guider la conception. Il serait donc intéressant de développer des représentations hiérarchiques adaptatives. Une représentation qui vient immédiatement à l'esprit est l'emploi des structures d'arbres, considérées comme des programmes, de la Programmation Génétique (cf. chapitre 1), qui ont été utilisées avec succès sur d'autres problèmes de conception [21, 106].

Chapitre 5

Représentation à base de systèmes de fonctions itérées (IFS)

Sommaire

5.1 Introduction

Les représentations à base de diagrammes de Voronoï sont des techniques utilisées pour découpler l'encodage des structures et la discrétisation qui sert au calcul de la performance. Toutefois, les blocs élémentaires servant à construire la structure doivent être définis, et des choix inadéquats peuvent biaiser la recherche dans une mauvaise direction.

La représentation suivante, à base de fractales, est une tentative pour aller plus loin dans la direction morphogénétique : aucune hypothèse n'est faite sur ce que doit être la forme des blocs élémentaires, mais l'espace de recherche pour le génotype est supposé assez riche pour que n'importe quel type de structure puisse apparaître.

Un IFS (Système de Fonctions Itérées) est défini par la donnée d'un espace métrique complet et un ensemble de transformations contractantes (cf. section 5.2 pour une définition détaillée). Le principal intérêt d'un tel objet mathématique est qu'il définit de façon unique un ensemble particulier, appelé son "attracteur" qui peut être construit simplement grâce à un algorithme itératif. Les attracteurs d'IFS furent utilisés à l'origine pour définir des ensembles fractals (tapis de Sierpinski, fougère de Barnsley, etc.). Mais la propriété des IFS, qui motive leur utilisation pour représenter des formes est leur capacité à décrire des formes complexes avec un petit nombre de paramètres. Elle est à l'origine du succès de la théorie des IFS dans le domaine de la compression du signal, mais elle a de nombreuses autres applications en analyse d'image ou de signal, en particulier en tant qu'outil de représentation très souple (cf. par exemple [171]).

Dans ce chapitre nous étudions l'emploi de ces représentations fractales de formes, à base d'IFS mixtes, ainsi que des résultats préliminaires montrant ses avantages potentiels.

Ce travail a été réalisé en collaboration avec **Evelyne Lutton** [1] [79][78].

1. Responsable scientifique du projet FRACTALES à l'INRIA Rocquencourt.

5.2 Introduction à la théorie des IFS

5.2.1 Notations et définitions

Cette section rappelle brièvement les bases de la théorie des IFS et les algorithmes numériques les plus connus, pour calculer les attracteurs d'IFS.

Définition 5.1

Soit (F,d) un espace métrique complet. Un IFS $\Omega = \{F,(w_n)_{n=1,..,N}\}$ *est un ensemble de N fonctions* w_n *définies de F dans F.*

Définition 5.2

Une fonction w, définie sur un espace métrique (F,d), est dite contractante s'il existe un réel positif $s < 1$ *tel que*

$$\forall\ (x,y)\ \in\ F^2,\ \ d(w(x),w(y))\ \leqslant\ s.d(x,y)$$

s est appelé facteur de contractance de ω.

Un résultat crucial pour les fonctions contractantes est donné par le théorème de point fixe suivant :

Théorème 5.1

Soient (F,d) un espace métrique complet, et $w : F \longrightarrow F$, *une fonction contractante, alors w possède un unique point fixe.*

Définition 5.3

Soient $\Omega = \{F,(w_n)_{n=1,..,N}\}$ *un IFS, et W l'opérateur de Hutchinson défini sur l'espace des sous-ensembles de F par :*

$$\forall\ K\ \subset\ F,\ \ W(K) = \bigcup_{n\in[0,N]} w_n(K)$$

Si les fonctions w_n *sont contractantes, l'IFS est dite hyperbolique (ou contractant). Son facteur de contractance est le plus petit des facteurs de contractance des* w_n.

Une propriété importante des IFS est donnée par :

Proposition 5.1

Si un IFS Ω *est contractant, alors il existe un unique ensemble* $A \subset F$, *appelé l'attracteur de l'IFS, tel que*

$$W(A)\ =\ A$$

L'unicité de l'attracteur est le résultat du théorème du point fixe pour W, qui est contractante relativement à la distance de Hausdorff (notée d_H) sur l'espace des sous-ensembles de F :

Définition 5.4

La distance de Hausdorff entre deux sous-ensembles A et B de F est définie par :

$$d_H(A,B) = \max[\max_{x\in A}(\min_{y\in B} d(x,y)), \max_{y\in B}(\min_{x\in A} d(x,y))]$$

5.2.2 Calcul de l'attracteur

D'un point de vue algorithmique, deux méthodes principales permettent de calculer l'attracteur d'un IFS par des approches différentes :

La méthode stochastique (toss-coin)

Soit x_0 un point fixe de l'une des fonctions w_i. On construit la suite de points x_n par $x_{n+1} = w_i(x_n)$, i étant choisi aléatoirement, uniformément dans $\{1..N\}$. Alors $\bigcup_n x_n$ est une approximation de l'attracteur de Ω (La précision de l'approximation augmente avec n).

La méthode déterministe

A partir de n'importe quel noyau A_0 de F, On construit la suite de sous-ensembles $\{A_n\}$ de F :

$$A_{n+1} \;=\; W(A_n) \;=\; \bigcup_{i=1}^{N} w_i(A_n)$$

Lorsque $n \longrightarrow \infty$, A_n est une approximation de l'attracteur de Ω.

La première approche est la plus utilisée en pratique, car elle nécessite beaucoup moins de calculs. De plus, à condition de partir d'un point situé dans l'attracteur [2], le parcours ne sort alors jamais de l'attracteur. Au contraire, la deuxième méthode impose d'estimer chaque ω_i en tout point de A_n.

5.3 Identification d'IFS par algorithmes évolutionnaires

5.3.1 Problème inverse

La première tentative pour faire évoluer des IFS par algoritme évolutionnaire concernait le problème inverse suivant [14] : pour une forme donnée $A \subset F$, trouver un IFS dont

2. On débute généralement du point fixe d'une des fonctions ω_i.

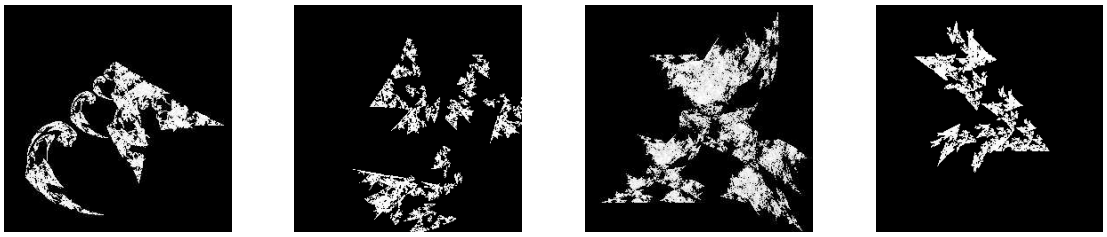


FIG. 5.1 – *Exemples d'attracteurs d'IFS. pris de [37]*

l'attracteur est A.

Ce problème a été étudié par plusieurs auteurs [14][110][38]. Une solution exacte de ce problème n'est calculable que dans des circonstances particulières. Généralement, on doit se contenter d'une approximation, obtenue par des méthodes d'optimisation. La stratégie de résolution directe utilisable lorsque l'on a très peu de connaissance a priori sur le problème consiste à transformer le problème inverse en un problème d'optimisation : trouver l'IFS dont l'attracteur minimise la distance à l'objectif A.

Le théorème du collage suivant [14] fournit une justification de cette transformation :

Théorème 5.2

Soient A l'attracteur de l'IFS $\Omega = \{F, (w_i)_{i=1,..,N}\}$, et λ son facteur de contractance. Alors :

$$\forall\, K \subset F, \quad d_H(K,W(K)) < \epsilon \implies d_H(K,A) < \frac{\epsilon}{1-\lambda}$$

Ce théorème montre que trouver un IFS Ω dont l'attracteur approxime le plus possible une forme donnée I est équivalent à minimiser la distance

$$d_H(I, \bigcup_{i=1}^{N} w_i(I)),$$

sous la contrainte que toutes les fonctions w_i sont contractantes. Cependant, lorsque le facteur de contractance est proche de 1, la borne perd tout son intérêt, ce qui montre la nécessité d'une estimation fine de ce facteur. Lorsque celle-ci est précise, il est alors préférable de minimiser

$$\frac{1}{1-\lambda} d_H(I, \bigcup_{i=1}^{N} w_i(I))$$

pour que la majoration conserve son sens. De plus, le calcul de la distance de Hausdorff lui-même est coûteux en temps de calcul.

Ces inconvénients mènent à considérer, comme dans [110], une fonction fitness basée sur l'algorithme de toss-coin et une distance euclidienne ou une distance qui calcule la différence des pixels entre deux formes.

5.3.2 Différents types d'IFS

Comme la fonction à optimiser est très complexe, des hypothèses restrictives supplémentaires sont nécessaires. Une approche fréquente est de limiter l'espace de recherche à celui des IFS affines, avec un nombre fini de fonctions (cf. [15, 170] pour les premières méthodes numériques). Plus récemment, des solutions basées sur les algorithmes évolutionnaires ont été proposées pour les IFS affines [173, 63, 124].

Dans le cas des IFS affines, les fonctions w_i qui composent un IFS Ω sont représentées (en 2D) par

$$w_i(x,y) = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} . \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix}$$

Le problème inverse consiste à optimiser les paramètres (a_i, b_i, c_i, d_i, e_i, f_i) pour obtenir l'attracteur qui approxime au mieux une forme cible donnée.

Cependant, ces IFS ne sont qu'un petit sous-ensemble de tous les IFS possibles. Dès lors que les w_i ne sont plus restreintes aux fonctions affines, certains auteurs [110] se sont intéressés à des IFS plus généraux (non affines) appelés *IFS mixtes* en utilisant la programmation génétique (GP) qui permet de faire évoluer n'importe quel type de fonction.

On rappelle que la programmation génétique manipule usuellement des structures arborescentes dynamiques (cf. chapitre 1, section 1.8.4), (considérés comme des programmes [104]). Cette structure de données est particulièrement adaptée à la représentation de fonctions : chaque w_i est représentée par un arbre (cf. par exemple 5.2-a) construit à partir de constantes prises dans [0,1], d'un ensemble de variables (x et y) et d'un ensemble de fonctions élémentaires donné par exemple par :

- $+$, $-$, $\times$
- cos, sin
- $div(x,y) = \dfrac{x}{0.0001 + |y|}$
- $root(x) = \sqrt{|x|}$
- $loga(x) = \log(1 + |x|)$

L'ensemble des arbres w_i représente alors l'IFS (figure 5.2-b). Toutefois, alors que l'obtention d'une condition nécessaire et suffisante pour la contractivité des fonctions affines est

triviale, la contractivité de fonctions générales définies par des arbres de programmation génétique ne peut être montrée que numériquement, *a posteriori*, et pour un coût de calcul très élevé. Cet inconvénient a motivé l'introduction très récente des IFS polaires [38] dans lesquels les fonctions sont définies – toujours en utilisant la GP – sous forme polaire autour de leur point fixe. L'approche est basée sur la construction de fonctions localement contractantes respectivement à un point P :

Définition 5.5

Une fonction $w : F \longrightarrow F$ *d'un espace métrique complet* (F,d) *dans lui-même est dite localement contractante relativement au point* P *s'il existe un nombre réel positif* $s < 1$ *tel que :*

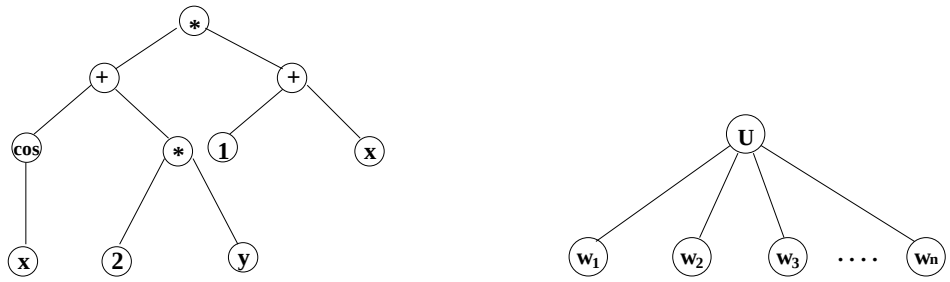
$$\forall\ M\ \in\ F,\quad d(P,w(M))\ \leqslant\ s.d(P,M)$$

Cette définition de contractance locale vis-à-vis d'un point s'accorde naturellement avec le système de coordonnées polaires. En effet, dans un tel système, un point M quelconque a pour coordonnées (ρ,θ) par rapport à P ($\overrightarrow{PM} = \rho e^{i\theta}$), où ρ représente la distance entre ces deux points. Une condition simple sur les fonctions ρ assure la contractivité locale autour de P (cf. [137] pour plus de détails). Cela ne permet pas de vérifier la contractivité globale, mais la proportion de fonctions contractantes dans cette classe de fonctions polaires est bien plus grande que dans les arbres GP généraux, permettant une résolution du problème inverse plus rapide et plus précise.

Malheureusement, au début de ce travail, seuls les programmes de GP pour l'identification des IFS mixtes étaient opérationnels. Les premiers résultats présentés dans les sections suivantes concernant l'utilisation de représentations IFS pour des problèmes d'optimisation topologique ont donc été obtenus en utilisant le programme d'IFS mixtes basé sur la programmation génétique [110].

5.4 Représentation par IFS : Résultats et discussion

L'idée de base de la représentation d'une forme par IFS est évidente : l'attracteur d'un IFS **est** la forme. Le programme de calcul de l'attracteur, utilisé est le même que dans [110]. L'attracteur d'un IFS donné est calculé sur le maillage qui sert à l'analyse par éléments finis et la performance est calculée comme dans la section 3.4 du chapitre 3.



(a) (b)

FIG. 5.2 – *(a) la fonction* $((cos(x) + 2y)(1 + x))$. *(b) Représentation d'un IFS mixte*

5.4.1 Conditions expérimentales

Les valeurs des paramètres utilisés pour l'expérimentation sont résumées dans le tableau 5.1. Ces paramètres ont été choisis après plusieurs tests de robustesse.

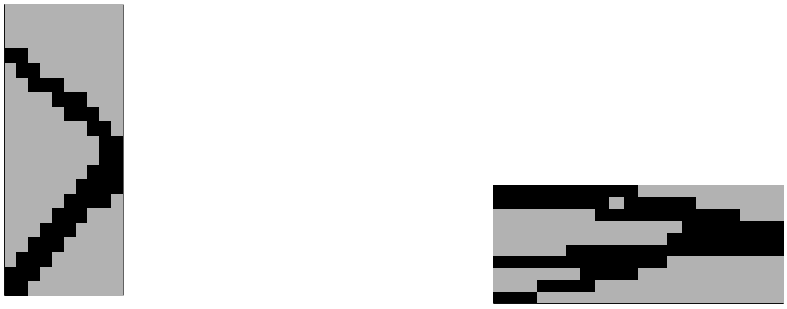
Taille de la population	30 à 50
nombre maximal de générations	2000 à 3000
Mode de sélection	Roulette
Remplacement	Générationnel avec élitisme
Probabilité de croisement	0.7
Probabilité de mutation	0.1 à 0.2
Intervalle des constantes	[0,1]
nombre min et max de fonctions contractantes	3 à 7

TAB. 5.1 – *Les paramètres*

5.4.2 Résultats et discussion

On reprend les cas-test des cantilevers 1×2 et 2×1 de la section 4.3. La figure 5.3 montre les meilleurs résultats obtenus après plusieurs calculs indépendants.

La première bonne nouvelle est qu'on obtient des structures raisonnables. De plus, leur allure ressemble plus à un treillis qu'en utilisant la représentation par cellules de Voronoï. Mais on peut entrevoir quelques défauts dans ces toutes premières expérimentations :



(a) : $D_{lim} = 20$, poids = 23.5% (b) : $D_{lim} = 220$, poids = 43%

FIG. 5.3 – *Résultats pour la représentation par IFS pour les deux cas-test de cantilever* 1×2 *et* 2×1

- La variance des résultats est très élevée ; certains résultats sont vraiment très mauvais.
- Nous avons utilisé la même stratégie de pénalisation adaptative que pour les représentations à base de diagrammes de Voronoï (cf. chapitre 3, section 3.5.5). Pourtant, alors que dans ce cas chaque calcul trouve au moins une solution satisfaisant les contraintes, de nombreux essais utilisant la représentation IFS ne trouvent aucune solution admissible.
- L'influence du raffinement du maillage sur la forme de l'attracteur obtenue après décodage de l'IFS n'est pas facile à comprendre, mais il semble que des maillages différents peuvent donner des formes très différentes pour un même IFS, et ce jusqu'à des finesses de maillages très importantes.

La figure 5.4 montre un des résultats les plus significatifs, obtenus sur le problème 10×1, pour une contrainte $D_{lim} = 12$, et un maillage régulier 100×10. Pour ce problème aussi la représentation par IFS n'a pas été capable de trouver de bon résultats. Les plaques consoles ne sont sans doute pas le bon problème pour mettre en valeur l'apport possible des IFS.

5.5 Conclusion

Nous avons présenté la représentation par IFS dans laquelle la structure est définie indirectement comme l'attracteur d'un ensemble de transformations contractantes définies sur

FIG. 5.4 – *Structure optimale sur un maillage* 100×10 *pour le cantilever* 10×1. $D_{lim} = 12$. *poids = 58%.*

le domaine de travail. Une telle représentation ne fait aucune hypothèse *a priori* sur la forme des blocs élémentaires constitutifs d'une solution d'un problème d'optimisation topologique. Cette technique devrait permettre d'atteindre des solutions plus complexes sans avoir à définir de gènes élémentaires spécifiques comme au chapitre 4 avec la représentation de Voronoï.

Pourtant, la représentation par IFS n'a pas été capable de trouver d'excellent résultats sur les cas-test qui ont été essayés. Bien sûr, il peut être objecté qu'une telle représentation, qui augmente la complexité du processus morphogénétique, ne doit prouver son efficacité que pour des problèmes pour lesquels la solution est complexe. Des études à venir tenteront de confirmer cette hypothèse. Mais il est aussi possible que le manque de feed-back direct de la structure mécanique sur son génotype (l'IFS) empêche des évolutions utiles sur les populations restreintes et de si petits nombres de générations. Des expérimentations sur des machines massivement parallèles avec des populations distribuées de plusieurs milliers d'individus sont nécessaires pour répondre à cette question.

Chapitre 6

Optimisation topologique multi-critères

Sommaire

6.1 Introduction

Dans le domaine de l'optimisation topologique de formes, la plupart des études sont consacrées à la minimisation simultanée du poids et de la rigidité (ou, d'une manière équivalente, le déplacement maximal sous l'action d'un chargement donné). Il est bien connu que ces objectifs [1] sont contradictoires (c'est-à-dire la diminution du poids entraîne généralement la diminution de la rigidité et vice-versa).

Une des approches les plus classiques pour traiter ces objectifs contradictoires consiste à minimiser un seul objectif tout en considérant les autres objectifs sous forme de contraintes [2] (par exemple minimiser le poids de la structure avec une contrainte sur le déplacement maximal). On est ainsi face à un problème mono-objectif sous contrainte déjà étudié dans les chapitres précédents.
Néanmoins, le développement des nouvelles méthodes d'optimisation multi-objectifs (ou multi-critères) permet de considérer les deux objectifs simultanément et d'essayer d'identifier l'ensemble des meilleurs compromis possibles entre les objectifs, aussi appelé le *front de Pareto*. Ce qui permet au décideur de choisir la solution à partir d'une information complète.

Nous rappelons dans ce chapitre les différentes approches multi-objectifs (section 6.2), en particulier, l'approche NSGA-II (section 6.3.6), qui sera appliquée dans la suite au problème d'optimisation topologique de formes (section 6.4).

Ce travail a été réalisé en collaboration avec **Olga Roudenko**, doctorante au C.M.A.P, et a fait l'objet d'une publication [80].

1. On utilise indifféremment les termes objectif et critère
2. L'autre approche (encore plus classique !) consiste à agréger les divers critères dans une seule fonction objectif, en se ramenant ainsi à un problème d'optimisation mono-objectif.

6.2 Optimisation multi-critères

Les problèmes rencontrés dans la réalité sont la plupart du temps multi-objectifs. De plus leur résolution nécessite des décisions qui doivent souvent être prises au vu d'objectifs contradictoires. Il est donc intéressant de fournir un ensemble de solutions afin de permettre aux décideurs d'avoir un choix varié et cela est d'autant plus important lorsqu'il est nécessaire de justifier les décisions avant de passer à une phase opérationnelle.

Depuis plusieurs années, le domaine de l'optimisation multi-critères connaît une évolution importante. Cette évolution s'est traduite par le développement d'un grand nombre de méthodes qui sont généralement regroupées en trois catégories :

- Approches, dites classiques, basées sur la transformation du problème en un problème mono-objectif (cf. section 6.2.2).
- Approches non-Pareto : elles utilisent un processus de recherche qui traite séparément les différents objectifs de manières "entrelacées" (cf. paragraphe 6.3.1).
- Approches Pareto basées sur la notion de dominance de Pareto (cf. section 6.3).

L'état de l'art des méthodes de résolution de problèmes multi-objectifs fait apparaître l'utilisation de méthodes d'optimisation diverses pour lesquelles les objectifs sont agrégés en une seule fonction objectif à optimiser. L'obtention de solutions alternatives se fait alors par initialisation multiple des algorithmes en modifiant des paramètres définissant les préférences des décideurs. Une alternative à cela est l'utilisation d'algorithmes d'évolution artificielle qui du fait de la gestion d'une population de solutions permettent d'obtenir les solutions alternatives en une seule exécution, profitant ainsi d'échanges d'informations entre solutions souvent meilleures les unes que les autres sur une partie des objectifs uniquement.

Afin de présenter les algorithmes d'optimisation multi-objectifs, nous devons tout d'abord introduire le principe de dominance (solutions dominées - ou non dominées - au sens de Pareto) entre solutions ainsi que la définition du front de Pareto.

6.2.1 Définitions

La notion de dominance

Soient $f_1, \ldots, f_n$ (des fonctions de E dans $\mathbb{R}$) les objectifs dont on suppose qu'on cherche à minimiser sur l'espace de recherche E.

Définition 6.1

Soient x et y deux points de E. On dira que x domine *y au sens de Pareto, noté $x \succ y$, si x est meilleure ou au moins équivalente à y sur tous les objectifs, et strictement meilleure sur au moins un objectif. Autrement dit,*

$$1) \quad \forall\, i \ \in \ [1..n] \ , \ f_i(x) \ \leqslant \ f_i(y)$$
$$2) \quad \exists\, i_0 \ \in \ [1..n] \ , \ f_{i_0}(x) \ < \ f_{i_0}(y)$$

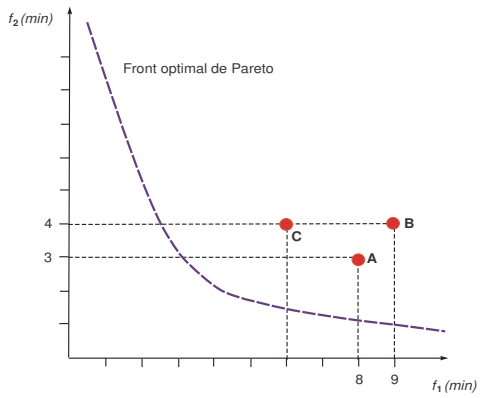


FIG. 6.1 – *Exemple de minimisation de deux objectifs f_1 et f_2. L'ensemble de l'espace de recherche est représenté dans l'espace des objectifs: plan (f_1,f_2). A(8,3) domine B(9,4) car $8 < 9$ et $3 < 4$, C(6,4) domine B(9,4) car $6 < 9$ et $4 = 4$. Par contre, A(8,3) et C(6,4) ne sont pas comparables (ordre partiel) car $6 < 8$ et $4 > 3$. On dit que A et C sont deux solutions non-dominées.*

Le front de Pareto optimal

Le but de l'optimisation multi-objectifs est de déterminer l'ensemble des solutions non-dominées, appelé front de Pareto. La figure 6.2 donne un exemple de front de Pareto : l'ensemble de l'espace de recherche est représenté dans l'espace des objectifs, et les points extrémaux pour la relation de dominance au sens de Pareto forment le front de Pareto du problème. Suivant le type d'objectifs à optimiser (minimisation ou maximisation), les courbes de Pareto peuvent se présenter suivant quatre formes différentes pour un problème à deux objectifs (cf. figure 6.3). A noter que l'on n'est pas toujours dans une situation aussi régulière que celles présentées dans les figures 6.2 et 6.3, et que le front de Pareto peut être concave, discontinu, ...

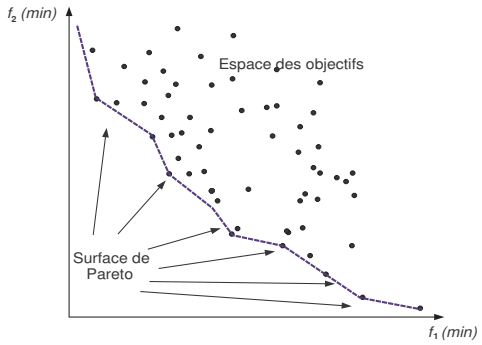


FIG. 6.2 – *Front de Pareto pour un problème de minimisation de deux objectifs : les points extrémaux de l'ensemble de l'espace de recherche forment le front de Pareto du problème.*

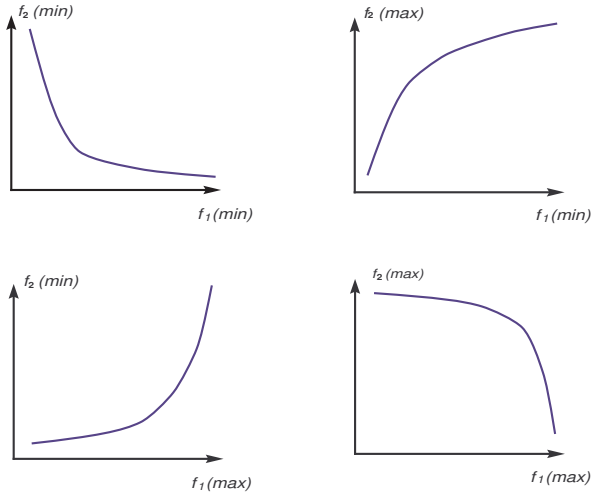


FIG. 6.3 – *Fronts de Pareto dans divers cas de maximisation/minimisation.*

6.2.2 Les méthodes classiques

L'approche séquentielle avec agrégation de critères

L'approche séquentielle consiste à agréger les divers objectifs f_i dans une seule fonction objectif $\mathcal{F}$, généralement de façon linéaire, en se ramenant ainsi à un problème d'optimisation

mono-objectif :

$$\mathcal{F}(x) = \sum_{i=1}^{n} \lambda_i f_i(x)$$

où les poids $\lambda_i \in [0,1]$ et $\sum_{i=1}^{n} \lambda_i = 1$. Ces fonctions pondérées nécessitent l'expression de manière quantitative des préférences relatives entre les différents objectifs. Cependant, les informations concernant les préférences relatives sont qualitatives et issues de l'expérience des utilisateurs, elles peuvent ainsi être très difficiles à représenter mathématiquement. Ces mêmes méthodes à objectifs agrégés peuvent ensuite être relancées plusieurs fois en modifiant le (ou les) paramètres représentant quantitativement les préférences relatives et ce en vue d'obtenir plusieurs solutions alternatives (solutions du front de Pareto).

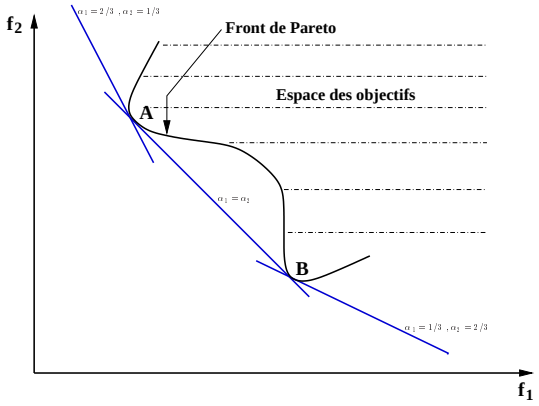


FIG. 6.4 – *Approche séquentielle.*

Considérons par exemple deux objectifs f_1 et f_2. Ces deux objectifs peuvent être agrégés en une seule fonction objectif $\mathcal{F}$ et leur importance relative représentée par un paramètre α :

$$\mathcal{F}(x) = \alpha f_1(x) + (1-\alpha) f_2(x)$$

On obtient alors un point de la surface de Pareto pour chaque valeur de α. Cependant, il est très difficile de choisir correctement les valeurs de α. De plus, dans certains cas, de telles représentations peuvent conduire à des résultats totalement inattendus en terme de préférences sur les objectifs. Ainsi, si l'on considère à minimiser f_1 et f_2 avec un front de Pareto concave (cf. figure 6.4); en supposant que le décideur a la même préférence pour les deux critères, il est naturel d'agréger les deux fonctions f_1 et f_2 en une seule fonction objectif qui décrit la somme pondérée (par des coefficients égaux α_1 et α_2) de f_1 et f_2. La minimisation de la fonction $(f_1 + f_2)$ ainsi obtenue correspond au déplacement de la ligne

($f_1 + f_2 =$ constante) à partir de l'origine (f_i prend des valeurs positives) vers le quart d'espace positif (cf. figure 6.4), et ce jusqu'à ce qu'un point de l'espace de recherche soit rencontré.
Malheureusement, dans des cas tels que celui-ci, la méthode d'agrégation des objectifs conduit aux points A ou B (figure 6.4). De plus, il est aisé de remarquer que pour ce cas toute combinaison de pondération de préférences se terminera par l'obtention des points A ou B. Cependant, toutes les solutions de la partie de la courbe entre A et B peuvent également constituer une alternative pour le décideur.

De manière plus générale, le profil de la courbe de Pareto étant inconnu, il n'y a souvent pas de moyen de savoir si le front de Pareto contient une région concave et ce même lors de l'obtention de résultats (excepté si le front de Pareto est représenté avec une précision suffisante), rendant ainsi inefficace l'application de méthodes d'agrégation de critères.

Méthode ϵ-contrainte

Cette méthode consiste à optimiser une seule fonction f_k parmi les objectifs et à considérer les autres comme des contraintes :

$$\begin{cases} \min f_k(x) \\ x \in E \\ f_j(x) \leqslant \epsilon_j \ , \quad j = 1,..,n \ , \ j \neq k \end{cases}$$

$\epsilon = (\epsilon_1,...,\epsilon_{k-1},\epsilon_{k+1},....,\epsilon_n)$. On résout ce problème avec différents vecteurs de ϵ_i pour obtenir différentes solutions de Pareto. Toutefois, la génération de plusieurs solutions nécessite l'exécution multiple de l'algorithme d'optimisation avec différentes contraintes, ce qui est un exercice coûteux en terme de calcul. C'est cette méthode qui a été utilisée avec le poids et le déplacement maximal dans les chapitres précédents.

Dans la section suivante, nous présentons les méthodes d'optimisation multi-objectifs évolutionnaires. Les principes de ces méthodes sont détaillés ainsi que les principales améliorations qui y ont été récemment intégrées.

6.3 Les méthodes évolutionnaires

Dans le domaine évolutionnaire, le but de l'optimisation multi-objectifs est d'échantionner le front de Pareto optimal (figure 6.2-a). Plusieurs techniques ont été proposées :

- l'approche séquentielle avec un algorithme d'évolution artificielle mono-objectif,

– le Vector-Evaluated GA (VEGA), dit Non-Pareto, qui n'utilise pas le principe de dominance,
– et les approches Pareto basées sur le principe de dominance (dont : MOGA, NPGA, NSGA, SPGA et NSGA-II).

Celles-ci sont présentées ci-dessous, selon l'ordre chronologique de leur apparition, ainsi que les avantages et désavantages inhérents à leur utilisation.

6.3.1 Vector-Evaluated GA (VEGA)

Cette méthode est la première se proposant de trouver un ensemble de solutions non-dominées. Proposée par Schaffer en 1984 [145], cette méthode repose sur une stratégie de sélection qui prend en compte l'ensemble des objectifs. VEGA implémente l'optimisation multi-objectifs de manière simple et directe. Dès lors que l'optimisation concerne m objectifs, la population de l'algorithme d'évolution artificielle est divisée en m sous-populations, chacune étant en charge d'une seule fonction objectif, indépendamment des autres (sélection parallèle).

Le principe en est le suivant (cf. figure 6.5) :

1. On divise la population en m sous-populations.
2. On sélectionne des individus, pour chaque sous-population, selon leurs performances pour un seul des objectifs.
3. On applique des opérateurs de croisement et de mutation sur la population complète.

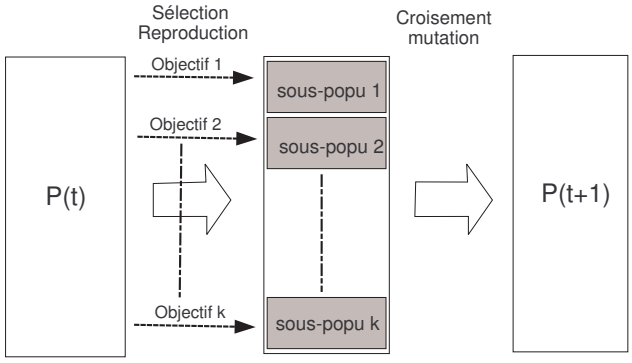


FIG. 6.5 – *Sélection parallèle dans l'algorithme VEGA.*

Avantages

L'avantage majeur de VEGA est qu'il utilise une idée simple et facile à implanter. Seules des modifications mineures sont à apporter à l'algorithme d'évolution artificielle classique. De plus, il ne génère pas de complexité de calcul additionnelle.

Désavantages

Cette méthode répartit la population sur les extrémaux du front de Pareto mais est limitée du fait de l'évolution mono-objectif des sous-populations qui ne permet pas la comparaison des diverses solutions multi-objectifs disponibles à chaque itération en vue d'en exploiter les caractéristiques. VEGA suppose que l'opérateur de croisement va combiner les bonnes solutions évaluées sur un seul des critères d'optimisation afin d'obtenir des solutions réalisant de bons compromis sur le front de Pareto. Cependant, et même lors de l'application de VEGA sur des espaces de recherche convexes, le croisement ne trouve souvent pas de solutions très diversifiées.

6.3.2 Multi-Objective GA (MOGA)

Proposée par Fonseca et Fleming en 1993 [58], cette approche est basée sur le classement non-dominé, dont le but est de classer les individus de la population par niveaux de non-dominance. C'est le premier algorithme incluant la préférence pour les solutions non-dominées conjointement au maintien de la diversité de la population. Il ne diffère de l'algorithme évolutionnaire mono-objectif que par la procédure d'affectation de la fitness, qui se calcule de la façon suivante :

1. Pour chaque individu i on calcule le nombre d'individus qui le dominent, noté p_i, dans la population courante, un rang r_i est alors affecté à l'individu $r_i = p_i + 1$. Ainsi, les solutions non-dominées ont un rang égal à 1.
2. Ensuite une fitness est affectée aux individus en utilisant une fonction de mise à l'échelle (souvent linéaire), un individu ayant un meilleur rang devant avoir une meilleure fitness.

Ce type de rang induit donc une plus forte pression de sélection, et peut causer une convergence prématurée. En vue de maintenir la diversité des solutions non-dominées, Fonseca et Fleming [58] ont introduit le principe de sharing (cf. section 1.5.1, chapitre 1) sur les solutions de même rang (avec $\sigma_{share} = 1$). La distance de sharing est calculée sur l'espace

objectif.

En résumé de cet algorithme, on retiendra que la sélection se fait sur les rangs et le sharing dans l'espace objectif.

Avantages

La procédure d'affectation de fitness est simple. De plus, si une bonne répartition sur l'espace objectif est désirée, MOGA peut constituer une bonne alternative.

Désavantages

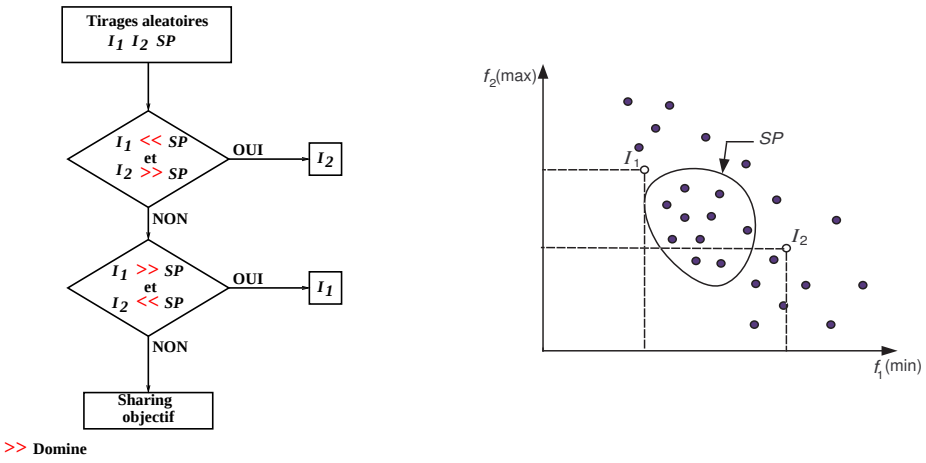
Le calcul de la fitness avec sharing n'assure pas que toute solution de rang inférieur a toujours une fitness inférieure à celle de solutions de meilleur rang. Ceci arrive lors de la présence d'une grande densité de solutions ayant un meilleur rang. De plus, ceci est susceptible de ralentir la convergence.
Le paramètre de sharing doit être initialisé arbitrairement. Cependant une mise à jour adaptative de ce paramètre en cours d'exécution est possible [58].

6.3.3 Niched Pareto GA (NPGA)

Proposé par Horn et Nafpliotis en 1993 [89], NPGA utilise le tournoi suivant au cours du processus de sélection (cf. figure 6.6) : Au début du processus, une sous-population SP de t_{dom} individus est tirée au hasard de la population, ainsi que deux individus I_1 et I_2 parmi lesquels on va sélectionner un gagnant. I_1 et I_2 sont comparés à tous les éléments de la sous-population SP. Si l'un est dominé et l'autre est non-dominé, alors le non-dominé est sélectionné. Sinon (les deux sont dominés ou non-dominés), on applique un sharing (cf. section 1.5.1) sur l'espace des objectifs. Le sharing ainsi utilisé sélectionne I_1 ou I_2 en fonction du taux d'agrégation de la population dans les voisinages respectifs de I_1 et I_2. Ainsi, l'individu le plus isolé sur le front de Pareto (ayant l'indice de nichage le plus petit) sera sélectionné.

Avantages

L'un des avantages majeurs du NPGA est qu'aucune procédure d'affectation de fitness n'est spécifiée. Un autre avantage est qu'il utilise un tournoi de sélection et que si la taille de la sous-population est inférieure à N, la complexité du NPGA ne dépend plus tellement du nombre d'objectifs. Il peut donc s'avérer intéressant d'utiliser le NPGA pour des



(a) (b)

FIG. 6.6 – *(a) Procédure de sélection (à base de tournoi) du NPGA. (b) Un exemple : I_1 est non-dominé par SP, tandis que I_2 est dominé par certains individus de SP; I_1 est alors le gagnant.*

problèmes avec un grand nombre d'objectifs.

Désavantages

NPGA nécessite le réglage de deux paramètres, σ_{share} et t_{dom}, dont le choix de leurs valeurs conditionne les performances de l'algorithme. Le paramètre t_{dom} contrôle la pression sélective : si t_{dom} est petit, la pression est faible et le nombre de solutions Pareto dans la population sera réduit, alors que si t_{dom} est grand, la pression sera forte et risque une convergence prématurée.

6.3.4 Non-Dominated Sorting GA (NSGA)

Comme MOGA, l'approche NSGA est basée sur la notion de non dominance, plus particulièrement sur la procédure du rang de dominance.

Rang de dominance : dans un ensemble de points de l'espace de recherche (e.g. une population), les solutions qui ne sont dominés par aucune autre solution se voient affecter le rang 1. Elles sont retirées de la population et les nouvelles solutions non-dominées se voient affecter le rang 2 (cf. figure 6.7). Ce processus continue jusqu'à l'affectation d'un rang à toutes les solutions.

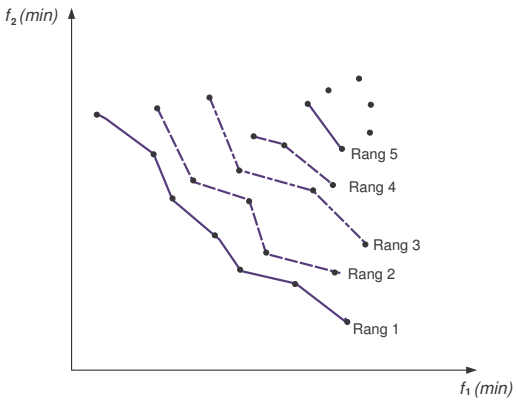


FIG. 6.7 – *Rang de Pareto pour un problème de minimisation de deux objectifs : une population donnée est partiellement ordonnée par la relation de dominance au sens de Pareto.*

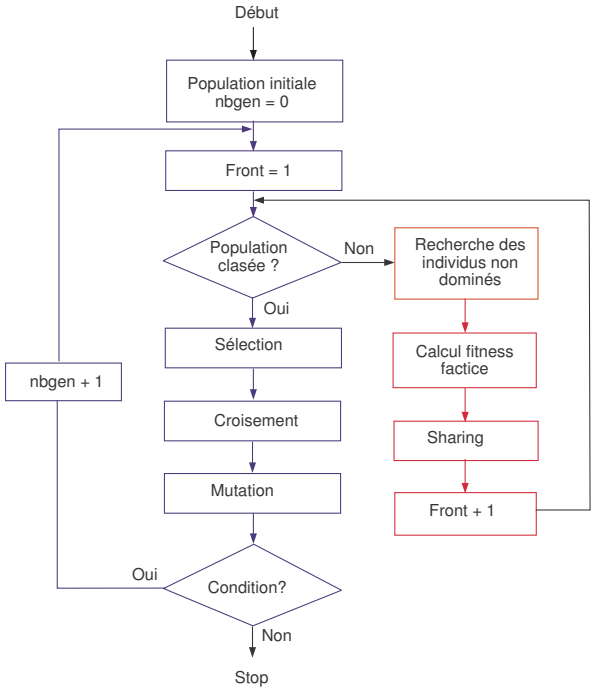


FIG. 6.8 – *Schéma de fonctionnement de NSGA*

Le concept de cette méthode a été initialement proposée par Goldberg [65] et implémenté par Srinivas et Deb [164]. Le principe de NSGA est présenté sur la figure 6.8 :

1. On commence par identifier, dans la population entière, les individus non-dominés. Ces derniers constituent le premier front.
2. On leur associe une valeur de fitness factice. Afin d'assurer la diversité dans la population, on applique un sharing sur ce premier front.
3. Ensuite, ces individus sont enlevés de la population.
4. On recommence cette procédure pour déterminer le second front. La fitness attribuée à ce second groupe est inférieure à la plus petite fitness après application de la fonction de sharing sur le premier groupe.

Ce processus est réitéré jusqu'à ce que tous les individus soient traités.

Un exemple d'application de NSGA est donné (figure 6.9 et table 6.1) dans ce qui suit : $f_1(x) = x^2$, $f_2(x) = (x-2)^2$ et $\sigma_{share} = 0.5$.

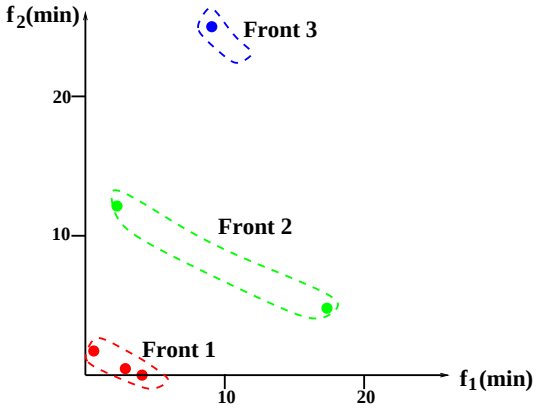


FIG. 6.9 – *Exemple pour NSGA - Voir la table 6.1*

solution	x	f_1	f_2	Front	fitness factice	fitness après sharing
1	−1.50	2.25	12.250	2	3.00	3.00
2	0.70	0.49	1.69	1	6.00	6.00
3	4.20	17.64	4.840	2	3.00	3.00
4	2.00	4.00	0.000	1	6.00	3.43
5	1.75	3.06	0.062	1	6.00	3.43
6	−3.00	9.00	25.000	3	2.00	2.00

TAB. 6.1 – *Table - Exemple de 3 fronts pour le NSGA*

Le sharing est effectué sur l'espace des objectifs, ainsi les solutions 4 et 5 sont très proches et leurs deux critères diminués, alors que la solution 2 garde la même valeur (6.00).

Avantages

L'avantage principal du NSGA est l'affectation de fitness au vu d'ensembles de solutions non-dominées. Dès lors que les ensembles de solutions non-dominées sont systématiquement différentiés, le NSGA progresse naturellement vers le front de Pareto optimal.

Désavantages

Le tri de la population est une heuristique intéressante mais cette procédure ralentit le processus de convergence de l'algorithme. De plus, la fonction de sharing requiert la spécification du paramètre σ_{share}, une mauvaise instantiation de ce paramètre pouvant conduire à une réduction des performances [164]. Aussi, une procédure d'instantiation adaptative de σ_{share} peut aussi être envisagée.

6.3.5 Strength Pareto GA (SPGA)

Proposée par Zitzler et Thiele en 1998 [178], l'approche SPGA utilise une sélection basée simultanément sur le concept de non-domination et l'élitisme[3]. Le schéma de fonctionnement de SPGA est présenté sur la figure 6.10 : à partir de la population initiale, l'algorithme construit une population externe avec les solutions non-dominées. Afin de réduire la taille de cet ensemble, une "clusterisation" est alors effectuée [121]. Les représentants de ces clusters sont ensuite utilisés pour participer au processus de sélection. Les opérateurs de croisement et de mutation sont alors appliqués pour construire la nouvelle population (génération suivante) à laquelle on applique à nouveau le processus de recherche des individus non-dominés afin d'enrichir l'ensemble de Pareto.

Pour définir les performances des individus, la population externe et la population courante sont combinées et les solutions non-dominées auront une performance qui dépend du nombre de points qu'elles dominent.

Avantages

Dès qu'une solution du front de Pareto optimal est trouvée, elle est immédiatement intégrée

3. L'élitisme consiste à maintenir une population autre que la population courante, qui permet d'archiver toutes les solutions Pareto optimales trouvées durant la recherche

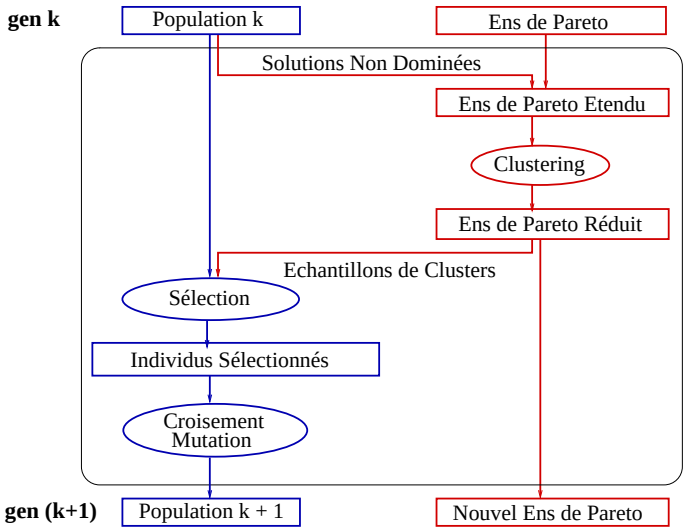


FIG. 6.10 – *Procédure du SPGA.*

dans la population des solutions non-dominées. Le seul moyen de l'en retirer est trouver une autre solution (qui la domine) Pareto-optimale, qui permet une meilleure couverture du front de Pareto. L'algorithme de clustering ne nécessite pas de paramètres, ce qui rend SPGA facile à utiliser. La procédure d'affectation de fitness est assez similaire à celle de MOGA qui est très simple à calculer.

Désavantages

SPGA introduit un paramètre supplémentaire représentant la taille de la population des non-dominées N_d. Un équilibre entre la taille N de la population courante et celle des non-dominées est très important en vue de garantir le succès de SPGA. Si un grand N_d (comparé à N) est choisi, la pression sur les solutions élites va être plus grande et SPGA peut dans ce cas être incapable de converger vers le front de Pareto optimal. A l'inverse, une taille de population externe très réduite réduit l'effet de l'élitisme. De plus, de nombreuses solutions ne seront pas dominées par un membre de la population des non-dominées. Les utilisateurs de SPGA préconisent un ratio de 1/4 entre N_d et N.

6.3.6 Non-Dominated Sorting GA (NSGA-II)

Dans cette deuxième version de NSGA [44], l'auteur tente de résoudre les critiques fortes sur NSGA, complexité, utilisation de sharing et non-élitisme. La sélection et le remplacement du NSGA-II sont basés sur deux critères à ordres hiérarchiques [44] : en vue de

comparer deux individus (lors d'un tournoi de sélection), leurs rangs de dominance sont comparés, le plus petit étant le meilleur (c'est le critère de Pareto) ; si les rangs de dominance sont égaux, une mesure locale de densité de population dans l'espace de recherche autour de l'individu est calculée (distance de "crowding" définie ci-dessous), une plus grande distance étant meilleure (c'est le critère de diversité). Ainsi, la relation d'ordre de la sélection et du remplacement utilisée par le NSGA-II est totale.

Distance de crowding

La distance de crowding est une mesure de la densité des solutions sur l'espace objectif. Elle n'inclut que les individus possédant le même rang, aussi appelé front de Pareto partiel. Chaque front de Pareto partiel est trié au vu d'une seule fonction objectif. La distance partielle de crowding d_i (pour un seul objectif i), pour chaque individu p, est définie comme la somme des distances de p à ses deux plus proches voisins dans la liste ordonnée. La distance de crowding totale est donnée par la somme, sur l'ensemble des objectifs, des distances partielles. Elle peut aussi être vue comme le plus large centroïde englobant les individus et ce sans inclure d'individus du même rang.

Cette mesure n'est basée sur aucun paramètre utilisateur. Cependant, elle requiert un tri de la population pour chaque objectif, devenant ainsi très coûteuse en terme de temps d'exécution pour des problèmes avec de nombreux objectifs et/ou des populations de grande taille.

Algorithme NSGA-II

Les principales étapes du NSGA-II peuvent se résumer de la manière suivante (cf. figure 6.11) :

1. Combiner les populations parents P_t et enfants Q_t. Appliquer un tri sur les solutions non-dominées (tri sur $2N$ solutions) en vue d'identifier les fronts de Pareto partiels Fi (classement par rang). Ce qui permet une validation globale de non dominance.
2. Initialiser une nouvelle population P_{t+1} vide, et y ajouter les individus des fronts partiels en commençant par le front 1 et jusqu'à atteindre un nombre N d'individus.
3. Appliquer la procédure de tri par crowding et inclure les solutions les mieux réparties sur les fronts triés F_i au vu de la distance de crowding dans P_{t+1}.
4. Créer la population des individus fils Q_{t+1} à partir de la population P_{t+1} en utilisant le tournoi de sélection par crowding et les opérateurs de mutation et de croisement.

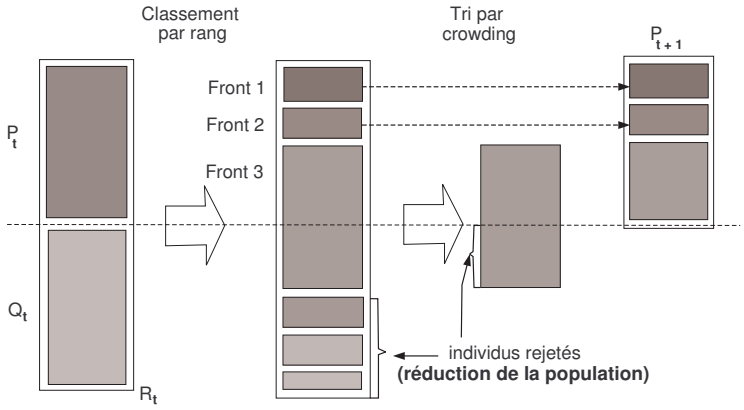


FIG. 6.11 – *Schéma de fonctionnement de NSGA-II, pris de [43]*

Avantages

La diversité des solutions non-dominées est introduite par l'utilisation de la procédure de comparaison de crowding qui est utilisée lors du tournoi de sélection et durant la phase de réduction de la population (cf. figure 6.11). Dès lors que les solutions sont en compétition au vu des distances de crowding (mesure de densité de la population dans le voisinage des solutions), aucun paramètre additionnel de nichage n'est nécessaire (tel que σ_{share} utilisé pour MOGA, NSGA ou le NPGA). D'autre part, et de la même manière, la distance de crowding qui est ici calculée sur l'espace objectif peut l'être sur l'espace de recherche.

Désavantages

Dès lors que la taille du premier ensemble de solutions non-dominées n'est pas plus grand que la taille de la population, cet algorithme les préserve toutes. Cependant, lors des générations suivantes, lors de l'existence de plus de N individus dans l'ensemble représentant le premier front de solutions non-dominées, des solutions non-dominées peuvent être retirées du processus. La convergence de l'algorithme pourrai ainsi être ralentie.
Un autre désavantage est que le tri sur les solutions non-dominées doit se faire sur une population de taille $2N$ au lieu d'une population de taille N pour la plupart des autres algorithmes.

Conclusion

Les approches d'optimisation multi-objectifs évolutionnaires basées sur la recherche du front de Pareto sont plus efficaces que celles basées sur l'agrégation d'objectifs et ce d'au-

tant plus qu'elles utilisent le principe de dominance. De plus, ces méthodes appliquent des techniques permettant de préserver la diversité, et permettent ainsi d'obtenir un échantillonnage riche et uniforme du front de Pareto. Aussi, et même si elles sont coûteuses en terme de temps de calcul, elles permettent d'obtenir plusieurs solutions alternatives en une seule exécution. C'est précisément, l'un des avantages majeurs de l'optimisation multi-objectifs par évolution artificielle.
L'une des techniques les plus utiles en pratique est l'élitisme qui permet de préserver les meilleurs individus de chaque génération, mais l'utilisation de l'élitisme doit être accompagnée de techniques de préservation de la diversité. Il est ainsi nécessaire de décider du nombre de solutions élites à préserver et de faire un choix quant à garder les solutions élites dans la population courante ou d'avoir une population externe incluant les solutions élites et uniquement utilisée pour la sélection.
Des difficultés apparaissent pour le choix des tailles de populations adéquates ainsi que du réglage du paramètre de sharing. De plus, le sharing ainsi que le crowding ralentissent la convergence, mais ceci est amélioré par l'utilisation de l'élitisme.

Dans ce chapitre, nous avons choisi d'appliquer l'approche NSGA-II sur le problème d'optimisation topologique de formes avec deux objectifs (poids et rigidité).

6.4 Optimisation topologique multi-objectifs

Les résultats d'optimisation topologique de formes présentés dans les chapitres précédents concernaient la minimisation du poids de la structure avec une contrainte sur le déplacement maximal en présence d'un (ou plusieurs) chargements donnés, alors qu'ils peuvent également être abordés comme la minimisation simultanée du poids **et** du ou des déplacements maximaux sous un ou plusieurs chargements. En utilisant les techniques d'optimisation de Pareto (section 6.3), on peut obtenir en un seul essai un échantillonnage du front de Pareto du problème. Cette section présente les résultats obtenus sur les deux problèmes test de la plaque console.

6.4.1 Conditions expérimentales

Pour toutes les expériences numériques présentées dans la suite, les paramètres ont été choisis (après plusieurs tests de robustesse) comme suit : le nombre maximal de sites par individu est égale à 40 ; la taille de la population est fixée à 300 et le nombre maximum de générations à 400 ; le taux de croisement est de 0.7 et le taux de mutation de 0.2 par individu ; les poids relatifs de chaque type de mutation sont de 1/2 pour la mutation de déplacement et 1/6 pour les trois autres types de mutation.

Les méthodes de remplacement et sélection sont celles de l'approche NSGA-II décrite dans le paragraphe 6.3.6, avec un tournoi de taille 2. Les temps CPU sont donnés pour des processeurs Pentium III à 800MHz.

L'approche proposée a été appliquée sur les deux problèmes test de la plaque console, de dimensions 1×2 et 2×1, discrétisées respectivement selon un maillage régulier 10×20 et 20×10.

6.4.2 Résultats et discussion

La figure 6.12 montre un ensemble de structures sélectionnées à partir des résultats obtenus par trois essais de l'algorithme évolutionnaire multi-objectifs NSGA-II pour le problème test du cantilever 10×20. Ces structures sont classées par poids décroissant, de gauche à droite et du haut en bas.

La figure 6.13 représente de même les résultats pour le problème de la plaque console 20×10.

Les figures 6.14 et 6.15 montrent les fronts de Pareto correspondant aux deux problèmes. En regardant de près le front de Pareto de la figure 6.15, et en faisant un zoom autour de la valeur 220 du déplacement maximal, on remarque que l'approche de Pareto a trouvé un résultat similaire à celui obtenu par l'approche mono-objectif avec une contrainte sur le déplacement maximal de 220 (cf. chapitre 4, figure 4.14) : les deux structures ont des comportements mécaniques et un poids très proches (figure 6.16).

Comparons maintenant les temps de calcul : une seule expérience numérique de l'approche mono-objectif avec contrainte a pris 27mn, alors que 55mn ont été nécessaire pour un test multi-objectifs. Mais, d'autre part l'approche multi-objectifs a eu besoin de 100000 évaluation (calculs éléments finis) pour obtenir les meilleurs compromis possibles (front), alors que l'approche mono-objectif nécessite en moyenne 130000 évaluations. Cependant, la différence du coût CPU observé est due au traitements spécifiques de l'approche multi-objectifs, comme la sélection par rang de Pareto et la distance de crowding (cf. section 6.3). Par contre, cette différence restera la même quelque soit le coût d'un calcul de fitness. En particulier, pour des problèmes réels avec des maillages très fins, elle devient négligeable.

En conclusion, l'approche multi-objectifs permet d'obtenir pour un coût très voisin d'un calcul mono-objectif l'ensemble du front de Pareto : l'avantage dans ce contexte est clairement au multi-objectifs !

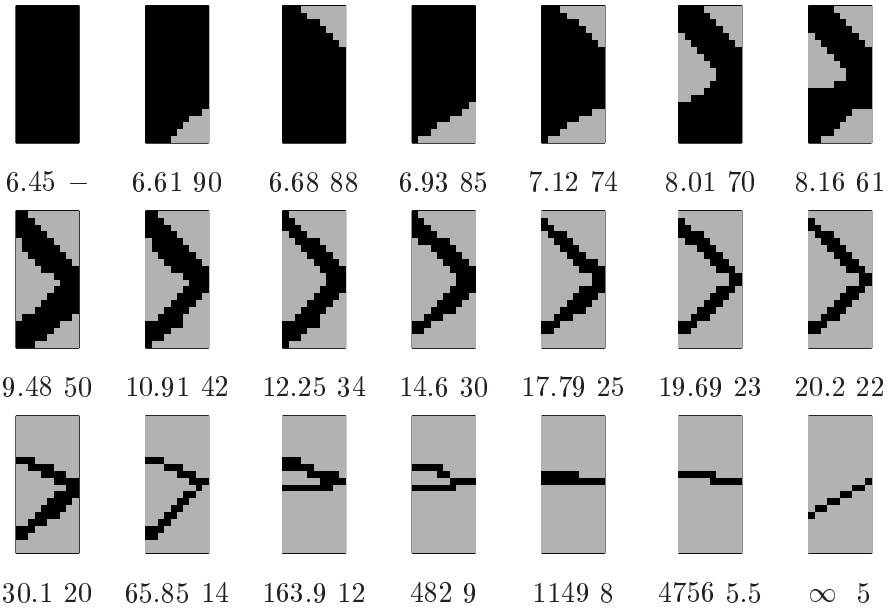


FIG. 6.12 – *21 compromis optimaux au sens de Pareto pour le problème de la plaque console* 10×20. *Le poids (en %) et le déplacement maximal sont indiqués sous chaque structure. La dernière structure atteint les limites de la modélisation.*

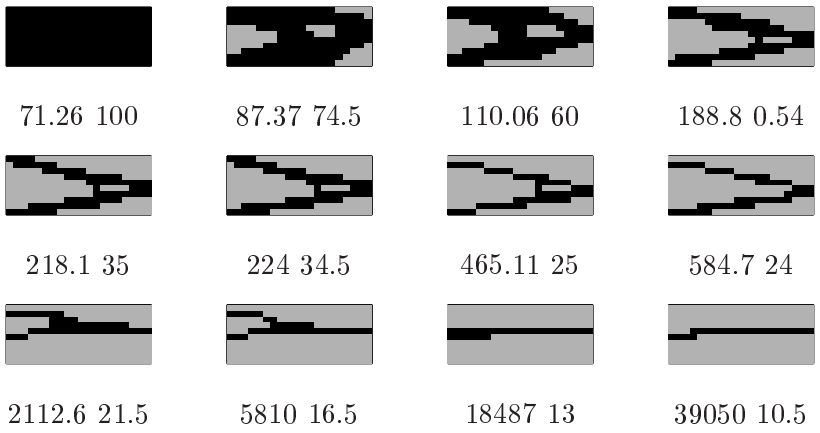


FIG. 6.13 – *12 compromis optimaux au sens de Pareto pour le problème de la plaque console* 20×10. *Le poids (en %) et le déplacement maximal sont indiqués sous chaque structure.*

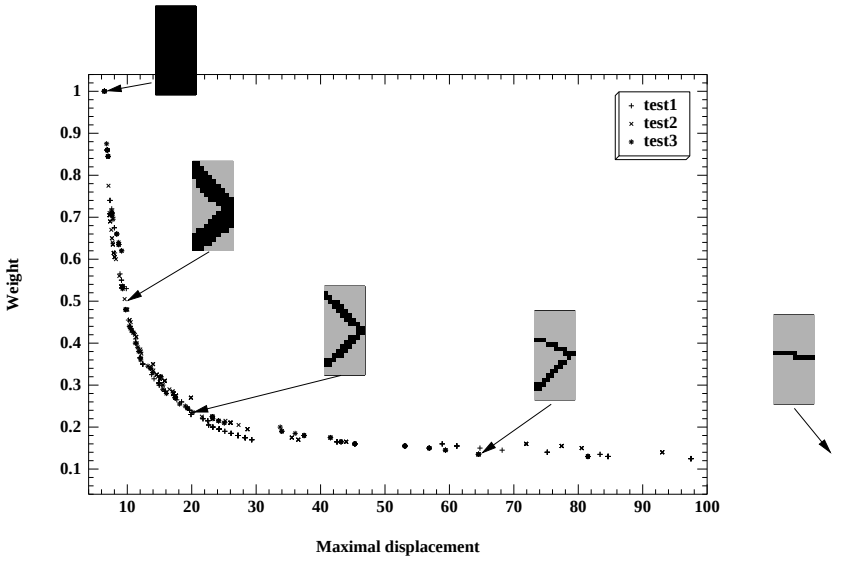


FIG. 6.14 – *Trois fronts de Pareto pour le problème de la plaque console* 10×20 *obtenus indépendamment après 400 générations de 300 individus (test1, test2 et test3).*

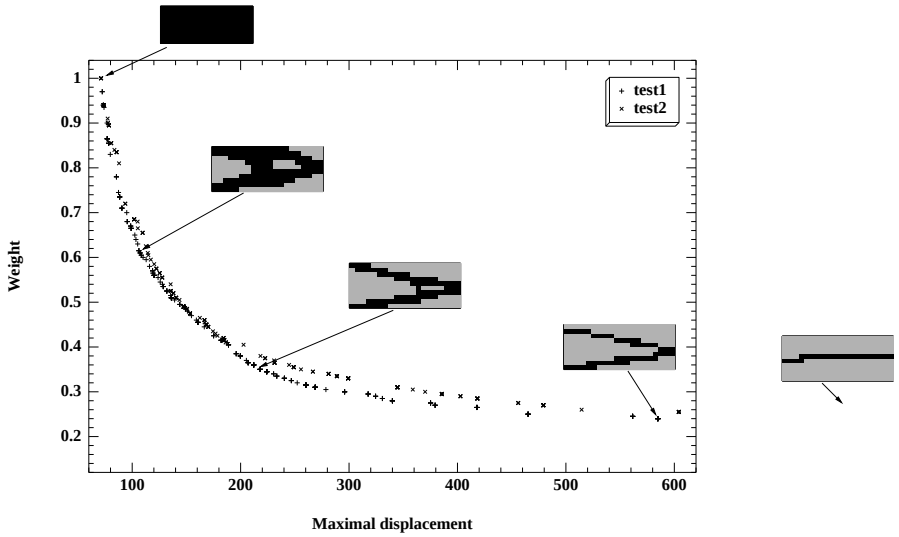


FIG. 6.15 – *Deux fronts de Pareto pour le problème de la plaque console* 20×10 *obtenus indépendamment après 400 générations de 300 individus (test1 et test2).*

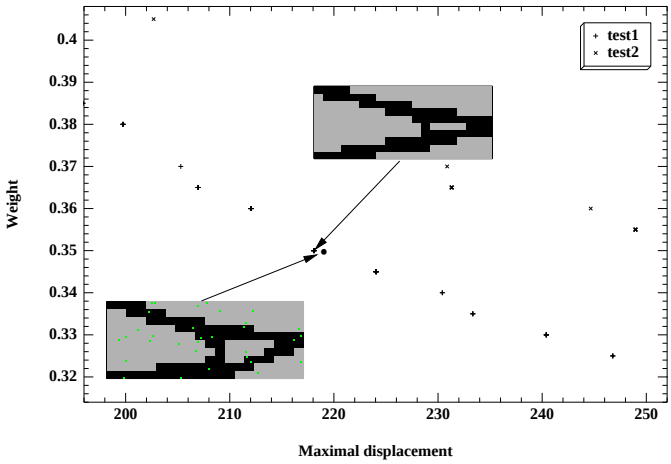


FIG. 6.16 – *Zoom sur le front de Pareto de la figure 6.15 au voisinage du déplacement maximal 220. La structure du haut a été obtenue par l'approche multi-objectifs, celle du bas par l'approche par contrainte.*

Remarque

Notons que de nombreuses structures sans intérêt font partie du front de Pareto (la structure pleine par exemple est optimale en terme de rigidité !) et qu'il faut en général en tenir compte dans l'algorithme pour éviter que de telles solutions n'envahissent la population.

6.5 Conclusion

L'optimisation multi-objectifs est un axe de recherche fondamental pour les scientifiques et les ingénieurs, non seulement à cause de la nature multi-critères des problèmes réels, mais aussi parce qu'il reste plusieurs questions ouvertes dans ce domaine.

Les résultats, sur des cas test simples, présentés dans ce chapitre doivent être considérés en tant que validation de l'application d'une approche multi-objectifs évolutionnaire combinée avec une représentation de Voronoï sur un problème d'optimisation topologique de formes. Les tests effectués ont permis d'obtenir un ensemble de structures (un échantillonnage du front de Pareto) qui sont très similaires à celles obtenues en utilisant l'approche mono-objectif avec contraintes (un test pour chaque structure). Si le but est d'obtenir

un tel échantillonnage, alors on peut statuer qu'une amélioration de l'effort de calcul à été réalisée. Cependant, des effort supplémentaires en vue du réglage de la stratégie de sharing peuvent encore être accomplis afin d'obtenir une meilleure couverture du front de Pareto.

De plus, l'approche multi-objectifs permet d'entrevoir de nouvelles perspectives en optimisation topologique de formes. Premièrement, ces méthodes peuvent être associées à d'autres représentations de structures, telles que celles présentées dans le chapitre 4 (par barres ou par dipôles), ou bien des représentations plus avancées basées sur la modularité (cf. chapitre 4).

Deuxièmement, il existe des critères à optimiser autres que le poids et la rigidité : l'optimisation modale est un domaine pour lequel l'approche multi-objectifs peut générer des améliorations : le but est généralement de maximiser la plus petite valeur propre, mais il est également envisageable de vouloir simplement éviter certaines plages de valeurs propres - autorisant les valeurs au dessus mais également en dessous de la plage interdite. Un autre type de problème où l'optimisation multi-objectifs peut être utile est celui de l'optimisation en multi-chargement : au lieu d'agréger tous les cas de charge dans une seule fonction objectif, l'idée consiste à considérer chaque cas comme un objectif ce qui permettra beaucoup plus de flexibilité dans la conception finale.

Conclusion et Perspectives

Cette thèse est consacrée à l'application des algorithmes évolutionnaires au problème de l'optimisation topologique de formes. Les principales contributions de ce travail concernent la représentation des structures, c'est-à-dire le choix de l'espace de recherche, choix crucial, qui conditionne la qualité des solutions que l'algorithme peut ensuite trouver.

Dans le domaine de l'optimisation topologique de formes, les algorithmes évolutionnaires ont permis de résoudre des problèmes sur lesquels les méthodes déterministes n'ont pas de prise, comme dans des contextes différents du contexte de l'élasticité linéaire. Par contre, la résolution d'un tel problème à l'aide d'un algorithme évolutionnaire ne saurait se passer d'une analyse détaillée de la représentation des structures et de la définition de la fonction performance.

Après une introduction au domaine de l'évolution artificielle avec une synthèse des principales approches (chapitre 1), et une présentation de l'état de l'art en optimisation de formes de structures en mécanique de solides (chapitre 2), nous avons présenté dans le chapitre 3 le problème mécanique considéré et la fonction performance. Le problème est formulé dans un premier temps comme un problème d'optimisation sous contraintes. Pour tenir compte de ces contraintes, nous avons proposé deux nouvelles stratégies, basées sur la pénalité adaptative ajustée automatiquement selon l'état courant de la population. Les résultats numériques obtenus sur deux problèmes test classiques de l'optimisation topologique de formes ont montré l'efficacité et la robustesse des approches adaptatives par rapport aux approches statiques et dynamiques utilisées dans des travaux antérieurs.

En ce qui concerne la représentation des structures, l'approche utilisée dans les travaux antérieurs est basée sur un maillage du domaine de design, dont la complexité était liée à celle dudit maillage. Afin d'appliquer la même technique à des maillages très fins, voir à des problèmes 3D, il fallait changer de représentation. C'est ainsi que, dans un pre-

mier temps, nous avons proposé et étudié de nouvelles représentations compactes et non structurées dont la complexité est indépendante de celle de tout maillage (chapitre 4). Ces représentations sont basées sur la théorie des diagrammes de Voronoï, depuis la représentation Voronoï simple jusqu'aux plus complexes représentations par dipôles ou barres. Ces trois types de représentations ont permis d'obtenir des méthodes où la complexité est auto-adaptative, i.e. pour lesquelles la complexité effective des individus évolue à travers l'algorithme.
Les résultats obtenus montrent que leur utilisation permet de repousser les limites de l'optimisation topologique de formes évolutionnaire – en particulier, des résultats originaux ont pu être obtenus en dimension 3, et pour des structures très élancées. Une étude expérimentale comparative montre que la représentation par barres de Voronoï semble être un bon choix pour l'optimisation topologique par algorithmes évolutionnaires en 2D, réalisant un bon compromis entre la taille du codage des individus et la facilité de recherche des bonnes solutions. Toutefois, alors que la généralisation des représentations Voronoï et par dipôles à trois dimensions est immédiate, la représentation par barres de Voronoï demande un peu plus d'efforts. En particulier, il sera sans doute nécessaire d'introduire des plaques et des barres de différentes sections dans le catalogue des gènes élémentaires.

Par ailleurs, nous avons proposés des représentations modulaires, permettant la réutilisation d'une partie de la représentation, sur le problème de cantilever 10×1. Les premiers résultats montrent la performance de ces approches surtout au niveau complexité des individus. Cependant, cette façon de faire "manuelle" est imprécise et nécessite la présence d'un expert pour guider la conception. Il serait donc intéressant de développer des représentations hiérarchiques adaptatives permettant l'évolution de la modularité. Deux approches sont envisagées dans des prochains travaux. D'une part, la représentation dite "des trous" peut être complétée par des opérateurs de reproduction des trous codés dans le génotype lui-même, et évoluant avec la forme des trous (exemple de tel opérateur : reproduction horizontale P fois avec décalage de N cm, oú N et P seraient sujets à l'évolution). D'autre part, la Programmation Génétique permet de décrire un programme de construction de la forme cherchée, et qui dit programme dit bien sûr possibilités de récurrence et de modularité.

Outre les représentations à base de diagrammes de Voronoï, nous avons proposé la représentation par IFS dans laquelle la structure est définie indirectement comme l'attracteur d'un ensemble de transformations contractantes définies sur le domaine de travail (chapitre 5). Les résultats expérimentaux d'une part justifient a posteriori l'introduction de cette représentation par le fait qu'elle permet d'obtenir des structures plus "découpées", mais mettent également en évidences les limites de cette approche.

Dans le dernier chapitre nous avons reformulé le problème de l'optimisation de formes comme un problème multi-objectifs et non plus sous contrainte. Nous avons commencé par une présentation détaillée des méthodes multi-objectifs dans la littérature, accompagnée d'une discussion de leurs avantages et leurs limites. Les approches d'optimisation multi-objectifs évolutionnaires basées sur la recherche du front de Pareto sont plus efficaces que celles basées sur l'agrégation d'objectifs puisqu'elles permettent, grâce à l'utilisation de techniques spécifiques, d'obtenir plusieurs solutions alternatives en une seule exécution. Nous avons choisi d'appliquer l'approche NSGA-II combinée avec la représentation par diagrammes de Vorornoï sur le problème d'optimisation topologique de formes avec deux objectifs (poids et rigidité). Les premiers résultats obtenus sont excellents : les tests effectués ont permis d'obtenir en un seul essai un ensemble de structures (un échantillonnage du front de Pareto) qui sont très similaires à celles obtenues en utilisant l'approche mono-objectif avec contraintes (un test pour chaque structure).

Le seul problème abordé dans cette thèse a été celui de l'optimisation en rigidité pour un poids minimum. Il est cependant très clair que tout problème mettant en jeu un comportement mécanique peut se traiter de la même manière. Dans le cadre de l'optimisation modale, on peut bien sûr maximiser la plus petite valeur propre, mais il est également envisageable de vouloir simplement éviter certaines plages de valeurs propres – autorisant les valeurs au dessus mais également en dessous de la plage interdite. Dans le cadre de l'optimisation de mécanismes, ou de propriétés thermo-mécaniques, de nombreuses applications sont également immédiates à mettre en oeuvre.

De plus, l'approche multi-objectifs ouvre de nouvelles perspectives en optimisation topologique de formes.

- Ces méthodes peuvent être associées aux représentations par barres ou par dipôles, mais aussi aux représentations plus avancées basées sur la modularité.
- Il est possible de trouver les meilleurs compromis entre plusieurs critères contradictoires, y compris des critères non différentiables (comme de s'éloigner au maximum d'une valeur propre donnée).
- Il serait intéressant d'appliquer l'approche multi-objectifs sur un problème d'optimisation multi-chargement, en considérant chaque cas comme un objectif ce qui permettra d'augmenter la flexibilité lors de la conception finale.

Un autre point important est qu'il serait certainement profitable de coupler l'algorithme évolutionnaire avec les algorithmes d'optimisation de forme déterministes, décrits dans le

chapitre 2, afin d'ajuster finement les solutions : c'est pour l'ajustement fin que les algorithmes évolutionnaires demandent beaucoup de temps de calcul.

Signalons enfin qu'il est très facile de paralléliser l'algorithme afin d'accélérer le processus de recherche.

Table des figures

Bibliographie

[1] G. Allaire, Z. Belhachmi, and F. Jouve. The homogenization method for topology and shape optimization. single and multiple loads case. *Revue Européenne des Eléments Finis*, 5:649–672, 1996.

[2] G. Allaire, E. Bonnetier, G. Francfort, and F. Jouve. Shape optimiztion by the homogenization method. *Nümerische Mathematik*, 76:27–68, 1997.

[3] G. Allaire, F. Jouve, and A. Toader. A level-set method for shape optimisation. *C. R. Acad. Sci. Paris*, 2002.

[4] G. Allaire and R. V. Kohn. Optimal design for minimum weight and compliance in plane stress using extremal microstructures. *European Journal of Mechanics, A/Solide*, 12(6):839–878, 1993.

[5] G. Ambrosio and L. Buttazo. An optimal design problem with perimeter penalization. *Calculs of Variations and Partial Differential Equations*, pages 55–69, 1993.

[6] G. Anagnostou, E. Ronquist, and A. Patera. A computational procedure for part design. *Computer Methods in Applied Mechanics and Engineering*, 97:33–48, 1992.

[7] Peter J. Angeline. Genetic programming's continued evolution. In Peter J. Angeline and K. E. Kinnear, Jr., editors, *Advances in Genetic Programming*, volume 2, pages 89–110. MIT Press, Cambridge, MA, USA, 1996.

[8] Peter J. Angeline and J. B. Pollack. The evolutionary induction of subroutines. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, Bloomington, Indiana, USA, 1992. Lawrence Erlbaum.

[9] S. Aubry. *Étude Théorique et Numérique de Quelques Problèmes d'Optimisation de Forme À l'Aide de Méthodes d'Homogénéisation.* PhD thesis, Universié Paris 6, 1996.

[10] T. Bäck. *Evolutionary Algorithms in theory and practice.* New-York:Oxford University Press, 1995.

[11] T. Bäck, G. Rudolph, and H.-P. Schwefel. Evolutionary programming and evolution strategies: Similarities and differences. In D. B. Fogel and W. Atmar, editors, *Proceedings of the 2^{nd} Annual Conference on Evolutionary Programming*, pages 11–22. Evolutionary Programming Society, 1993.

[12] J. E. Backer. Reducing bias and inefficiency in the selection algorithm. In J. J. Grefenstette, editor, *Proceedings of The 2^{nd} International Conference on Genetic Algorithms*, pages 14–21, 1987.

[13] J. E. Baker. Adaptive selection methods for genetic algorithms. In J. J. Grefenstette, editor, *Proceedings of the First international Conference on Genetic Algorithms and Their Applications*, pages 101–111, Hillsdale, New Jersey, 1985.

[14] M. Barnsley, V. Ervin, D. Hardin, and J. Lancaster. Solution of an inverse problem for fractals and other sets. In *Proceedings of National Academic Science*, volume 83, 1986.

[15] M. F. Barnsley. *Fractals everywhere*. Academic Press, 1988.

[16] Th. Bäck, D.B. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation*. Oxford University Press, 1997.

[17] M. Becker. Optimisation topologique de structure en variables discrètes. Technical report, Université de Liège, 1996.

[18] M. Bendsoe and N Kikushi. Generating optimal topologies in structural design using a homogenization method. *Computer Methods in Applied Mechanics and Engineering*, 71:197–224, 1988.

[19] M. P. Bendsoe. *Optimization of Structural Topology, Shape, and Material*. Spinger Verlag Berlin Heidelberg, 1995.

[20] S. BenHamida. *Algorithmes Évolutionnaires : Prise en Compte des Contraintes et Application Réelle*. PhD thesis, Université de Paris 11 – Orsay, 29 mars 2001.

[21] F.H. BennettIII, J.R. Koza, M.A. Keane, and D. Andre. Genetic programming: Biologically inspired computation that exhibits creativity in solving non-trivial problems. In *AISB'99 Symposium on Scientific Creativity*, pages 29–38. The Society for the Study of Artificial Intelligence and Simulation of Behaviour, 1999.

[22] L. Berke. An efficient approach to the minimum weight design of deflection limited structures. Technical Report AFFDL-TM-70-4-FDTR, USAF Technical Memorandom, 1970.

[23] S. S. Bhavikatti and C. V. Ramakrishnan. Optimum design of fillets in flat and round tension bars. *ASME Paper 77-DET-45*, 1977.

[24] T. Blickle and L. Thiele. A comparison of selection schemes used in genetic algorithms. Technical report, Computer Engineering and Communication Networks Lab, december 1995.

[25] J.-D. Boissonnat and M. Yvinec. *Géométrie algorithmique*. Ediscience International, 1995.

[26] V. Braibant. *Optimisation de forme des structures en vue de la conception assistée par ordinateur*. PhD thesis, Université de Liège, 1985.

[27] K. Saitou C. D. Chapman and M. J. Jakiela. Genetic algoritms as an approach to configuration and topology design. *Journal of Mechanical Design*, 116:1005–1012, 1994.

[28] J. Cea. Problems of shape optimum design. In E.J. Haug and J. Cea, editors, *Optimization of distributed parameter structures*, volume II of *NATO Series, Series*, pages 1005–1088, 1981.

[29] J. Céa. Conception optimale ou identification de forme, calcul rapide de la dérivée directionnelle de la fonction. *M.A.A.N.*, 20(3):371–402, 1986.

[30] J. Céa, S. Garreau, Ph. Guillaume, and M. Masmoudi. The shape and topological optimization connection. In *Fourth World Congress on Computationnal Mechanics*, Buenos Aires, Argentina, 1998. rapport MIP 98.22.

[31] R. Cerf. *une théorie assymptotique des algorithmes génétiques*. PhD thesis, Université de MontpellierII, March 1994.

[32] R. Cerf. An asymptotic theory of genetic algorithms. In J.-M. Alliot, E. Lutton, E. Ronald, and M. Schoenauer, editors, *Artificial Evolution*, volume 1063 of *LNCS*, pages 37–53. Springer Verlag, 1996.

[33] A. Cherkaev and R. Palai. Optimal design of three-dimensional axisymmetric elastic structures. *Int. J. of Structural and Multidisciplinary Optimization*, 1996.

[34] M. Chirehdast, , H-C Gea, N. Kikuchi, and P. Y. Papalambros. Structural configuration examples of an integrated optimal design process. *Journal of Mechanical Design. Transactions of the ASME*, 116:997–1004, 1994.

[35] K. K. Choi and E. J. Haug. Shape design sensitivity analysis of elastic structures. *J. Structural Mech.*, 11(2):231–269, 1983.

[36] P. G. Ciarlet. *Mathematical Elasticity, Vol I : Three-Dimensional Elasticity.* North-Holland, Amsterdam, 1978.

[37] P. Collet, E. Lutton, F. Raynal, and M. Schoenauer. Individual GP: an alternative viewpoint for the resolution of complex problems. In D.E. Goldberg & al., editor, *Proceedings of the Genetic and Evolutionary Conference 99*, volume 2, pages 974–981. Morgan Kaufmann, 1999.

[38] P. Collet, E. Lutton, F. Raynal, and M. Schoenauer. Polar IFS + individual GP = efficient inverse IFS problem solving. *Genetic Programming and Evolvable Machines*, 2000. To appear.

[39] G. Cretin, E. Lutton, J. Levy-Vehel, P. Glevarec, and C. Roll. Mixed IFS: Resolution of the inverse problem using genetic programming. In EA95editors, editor, *EA95*. Springer Verlag, 1996.

[40] 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.

[41] L. Davis. *Handbook of Genetic Algorithms.* Van Nostram Reinhold, New York, 1991.

[42] T. E. Davis and J. C. Principe. A simulated annealing like convergence theory for simple genetic algorithm. In R. K. Belew and L. B. Booker, editors, *Proceedings of the* 4^{th} *International Conference on Genetic Algorithms*, pages 174–181. Morgan Kaufmann, 1991.

[43] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms.* John Wiley, 2001.

[44] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In M. Schoenauer and al., editors, *Proceedings of the* 6^{th} *Conference on Parallel Problems Solving from Nature*, pages 849–858. Springer-Verlag, LNCS, 2000.

[45] 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.

[46] 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).

[47] A. Diaz and N. Kikuchi. Solutions to shape and topology eigenvalue optimization problems using a homogenization method. *Int. J. Num. Meth. Engng.*, 35:1487–1502, 1992.

[48] R. Dorne and J.-K. Hao. An evolutionary approach for frequency assignment in cellular radio networks. In D. B. Fogel, editor, *Proceedings of the Second IEEE International Conference on Evolutionary Computation*. IEEE, IEEE Press, 1995.

[49] R. Dorne and J.-K. Hao. A new genetic local search algorithm for graph coloring. In A.-E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Proceedings of the* 5^{th} *Conference on Parallel Problems Solving from Nature*. Springer Verlag, 1998.

[50] A.E. Eiben and Z. Ruttkay. Self-adaptivity for constraint satisfaction: Learning penalty functions. In T. Fukuda, editor, *Proceedings of the Third IEEE International Conference on Evolutionary Computation*, pages 258–261. IEEE Service Center, 1996.

[51] 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.

[52] C. Fleury. *Le Dimensionnement Automatique des structures Elastiques*. PhD thesis, Université de Liège, Belgique, 1978.

[53] C. Fleury and G. Sander. Dual methods for optimizing flexural systems. *Computer Methods in Applied Mechanics and Engineering*, 3(37):249–275, 1983.

[54] C. Fleury and L. A. Schmit. Structural synthesis by combining approximation concepts and dual methods. *AIAA Journal*, 18:1252–1260, 1980.

[55] 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, La Jolla, CA, 1992. Evolutionary Programming Society.

[56] D. B. Fogel. *Evolving artificial intelligence*. PhD thesis, University of California, San Diego, CA, 1992.

[57] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. New York: John Wiley, 1966.

[58] C. M. Fonseca and P. J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion, and generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, 1993.

[59] S. Garreau, Ph. Guillaume, and M. Masmoudi. The topological sensitivity for linear isotropic elasticity. In *Proceedings of European Conference on Computational Mechanics*, 1999. rapport MIP 99.45.

[60] R. A. Gellatly and R. H. Gallagher. A procedure for automated minimum weight structural design, part 1 - theoretical basis. *Aeronautical Quaterly*, 17(3):216–230, 1966.

[61] R. A. Gellatly and R. H. Gallagher. A procedure for automated minimum weight structural design, part 2 - applications. *Aeronautical Quaterly*, 17(4):332–342, 1966.

[62] C. Ghaddar, Y. Maday, and A. T. Patera. Analysis of a part design procedure. *Numerische Mathematik*, 73(4):465–510, 1995.

[63] B. Goertzel. Fractal image compression with the genetic algorithm. *Complexity International*, 1, 1994.

[64] D. E. Goldberg, , and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In G. J. E. Rawlins, editor, *Foundation of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

[65] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison Wesley, 1989.

[66] D. E. Goldberg, K. Deb, and J. H. Clark. Genetic algorithms, noise and the sizing of populations. *Complex Systems*, 6:333–362, 1992.

[67] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivations, analysis and first results. *Complex Systems*, 3:493–530, 1989.

[68] 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.

[69] J. J. Grefenstette and J. E. Baker. How genetic algorithms work: A critical look at implicit parallelism. In J. David Schaffer, editor, *Proceedings of The Third International Conference on Genetic Algorithms*, pages 20–27, San Mateo, CA, 1989. Morgan Kaufmann.

[70] D. Grierson and W. Pak. Discrete optimal design using a genetic algorithm. In M. Bendsoe and C. Soares, editors, *Topology Design of Structures*, pages 117–133. NATO Series, 1993.

[71] P. Guillaume and M. Masmoudi. Computation of high order derivatives in optimal shape design. *Numerische Mathematik*, 67:231–250, 1994.

[72] R. B. Haber, S. J. Chandrashekhar, and M. P. bendsoe. Variable topology shape optimisation with a control perimeter. *Advanced in Design Automation*, pages 69–82, 1994.

[73] R. B. Haber, C. S. Jog, and M. P. Bendsoe. A new approach to variable-topologie shape design using a constraint on perimeter. *Structural Optimization*, 11:1–12, 1996.

[74] J. Hadamard. Mémoire sur le problème d'analyse relatif à l'équilibre des plaques élastiques encastrées. In *Mémoire de Savants étrangers*, volume 33, 1908.

[75] A. B. Hadj-Alouane and J. C. Bean. A genetic algorithm for the multiple-choice integer program. Technical Report TR 92-50, Department of Industrial and Operations Engineering, The University of Michigan, 1992.

[76] R. T. Haftka and R. V. Grandhi. Structural shape optimization-a survey. *Computer Methods in Applied Mechanics and Engineering*, 57:91–106, 1986.

[77] S. Ben Hamida and M. Schoenauer. An adaptive algorithm for constrained optimization problems. In M. Schoenauer and al., editors, *Proceedings of the* 6^{th} *Conference on Parallel Problems Solving from Nature*, LNCS 1917, pages 529–538. Springer Verlag, 2000.

[78] **H. Hamda**, F. Jouve, E. Lutton, M. Schoenauer, and M. Sebag. Compact unstructured representations for evolutionary topology optimum design. *International Journal of Applied Intelligence*, 16:139–155, 2002.

[79] **H. Hamda**, F. Jouve, E. Lutton, M. Schoenauer, and M. Sebag. Représentations non structurées en optimisation topologique de formes par algorithmes évolutionnaires. In A. Blouza, I. Danaila, P. Joly, S.M. Kaber, B. Lucquin, F. Murat, and M. Postel, editors, *ESAIM Proceedings: Actes du 32ème Congrés d'Analyse Numérique*, volume 11, pages 153–179, 2002. http://www.emath.fr/Maths/Proc/Vol.11/ .

[80] **H. Hamda**, O. Roudenko, and M. Schoenauer. Multi-objective evolutionary topological optimum design. In I.C. Parmee, editor, *Adaptive Computing in Design and Manufacture V*, pages 121–132. Springer-Verlag, 2002.

[81] **H. Hamda** and M. Schoenauer. Adaptive techniques for evolutionary topological optimum design. In I. Parmee, editor, *Adaptive Computing in Design and Manufacture*, pages 123–136. Springer-Verlag, 2000.

[82] **H. Hamda** and M. Schoenauer. Topological optimum design with evolutionary algorithms. In *FGI2000, French - German - Italian conference on Optimization*, Montpellier-France, September 2000.

[83] **H. Hamda** and M. Schoenauer. Optimisation topologique de forme par algorithmes evolutionnaires. In *Cinqième Colloque National en Calcul des Structures*, volume 2, pages 805–813, Giens-France, Mai 2001.

[84] **H. Hamda** and M. Schoenauer. Topological optimum design with evolutionary algorithms. *Journal of convex analysis*, 9:503–517, 2002.

[85] **H. Hamda** and M. Schoenauer. *Innovative Tools For Scientific Computation in Aeronautical Engineering*, chapter Toward Hierarchical Representations for Evolutionary Topological Optimum Design, pages 331–345. CIMNE, Barcelona, June 2001. Eurodays 2000, in memoriam of B. Mantel.

[86] R. Hinterding, Z. Michalewicz, and A. E. Eiben. Adaptation in evolutionary computation: A survey. In T. Bäck, Z. Michalewicz, and X. Yao, editors, *Proceedings of the Fourth IEEE International Conference on Evolutionary Computation*, pages 65–69. IEEE Press, 1997.

[87] J. Holland. Outline for a logical theory of adaptive systems. *Journal of The Association of Computing Machinery*, 3, 1962.

[88] J. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, 1975.

[89] J. Horn and SN. Nafpliotis. Multiobjective optimization using the niched pareto genetic algorithm. Technical report, Department of General Engineering, University of Illinois at Urbana-Champaign, Illinois, 1993.

[90] 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 4th International Conference on Genetic Algorithms*, pages 31–36. Morgan Kaufmann, 1991.

[91] E. Jensen. *Topological Structural Design using Genetic Algorithms.* PhD thesis, Purdue University, November 1992.

[92] C. Jog, R. Haber, and M. Bendsoe. Topology design with optimized, self-adaptative materials. *Int. Journal for Numerical Methods in Engineering*, 37:1323–1350, 1994.

[93] J.A. Joines and C.R. Houck. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs. 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 579–584. IEEE Press, 1994.

[94] F. Jouve. *Modélisation mathématique de l'œil en élasticité non-linéaire*, volume RMA 26. Masson Paris, 1993.

[95] L. Kallel. *Convergence des algorithmes génétiques : aspects spatiaux et temporels.* PhD thesis, École Polytechnique, Février 1999.

[96] L. Kallel and M. Schoenauer. Alternative random initialization in genetic algorithms. In Th. Bäck, editor, *Proceedings of the 7^{th} International Conference on Genetic Algorithms*, pages 268–275. mk, 1997.

[97] C. Kane. *Algorithmes génétiques et Optimisation topologique.* PhD thesis, Université de Paris VI, July 1996.

[98] 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*. Springer Verlag, September 1995.

[99] C. Kane and M. Schoenauer. Topological optimum design using genetic algorithms. *Control and Cybernetics*, 25(5):1059–1088, 1996.

[100] C. Kane and M. Schoenauer. Optimisation topologique de formes par algorithmes génétiques. *Revue Française de Mécanique*, 4:237–246, 1997.

[101] R. N. Karnes and J. L. Tocher. Automated design of optimum hole reinforcement. Technical Report D6-23359, Boeing Report, 1968.

[102] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[103] J. R. Koza. *Genetic Programming: On the Programming of Computers by means of Natural Evolution.* MIT Press, Massachusetts, 1992.

[104] J. R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs.* MIT Press, Massachussetts, 1994.

[105] J. R. Koza and al. *Genetic Programming III: Automatic Synthesis of Analog Circuits.* MIT Press, Massachussetts, 1999.

[106] J. R. Koza, M. A. Keane, J. Yu, F. H. Bennett III, and W. Mydlowec. Evolution of a controller with a free variable using genetic programming. In R. Poli & al., editor, *EuroGP'2000*, volume 1802, pages 91–105. Springer Verlag, 2000.

[107] R. Leriche and R. T. Haftka. Optimization of laminate stacking sequence for buckling load maximization by genetic algorithms. *AIAA Journal*, 31(5):951–970, May 1993.

[108] R. G. Leriche, C. Knopf-Lenoir, and R. T. Haftka. A segregated genetic algorithm for constrained structural optimization. In L. J. Eshelman, editor, *Proceedings of the 6^{th} International Conference on Genetic Algorithms*, pages 558–565, 1995.

[109] C. Lin and P. Hajela. Genetic search strategies in large scale optimization. In *Structures, Structural Dynamics, and Materials Conference*, La Jolla, CA, April 1993. AIAA paper #93-1585.

[110] E. Lutton, J. Lévy Véhel, G. Cretin, P. Glevarec, and C. Roll. Mixed IFS: resolution of the inverse problem using genetic programming. *Complex Systems*, 9:375–398, 1995.

[111] 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.

[112] F. Mansanné. *Analyse d'Algorithmes d'Évolution Artificielle appliqués au domaine pétrolier : Inversion sismique et approximation de fonctions.* PhD thesis, Université de Pau, Decembre 2000.

[113] M. Masmoudi. The topological asymptotic. In H. Kawarada and J. Periaux, editors, *Computational Methods for Control Applications*, International series GAKUTO. To appear.

[114] M. Masmoudi. *Outils pour la conception optimale de formes*. PhD thesis, Université de Nice-Sophia-Antipolis, 1987.

[115] 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.

[116] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations for fast computing machines. *Journal of Chemical Physics*, 6(1087-1092), 1953.

[117] Z. Michalewicz. *Genetic Algorithms+Data Structures=Evolution Programs*. Springer Verlag, NewYork, 3rd edition, 1996.

[118] Z. Michalewicz and M. Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[119] A. Michell. The limits of economy of material in frame-structures. *Phil. Mag*, 8:589–597, 1904.

[120] M.I.Freidlin and A.D.Wentzell. *Random Perturbations of Dynamical Systems*. SpringerVerlag, 1984.

[121] J. N. Morse. Reducing the size of non-dominated set : Pruning by clustering. *Computers and Operation Research*, 7(1-2):55–66, 1980.

[122] F. Murat and J. Simon. Etude de problème d'optimum design. In *7th IFIP conf.*, 1976. Volume 41 of Lecture Notes in Computer Sciences.

[123] F. Murat and L. Tartar. *Calcul des Variations et homogénéisation*, pages 319–369. Les Méthodes de l'Homogénéisation: Théorie et Applications en Physique, Coll. Dir. Etudes et Recherches EDF. Eyrolles, 1985.

[124] D. J. Nettleton and R. Garigliano. Evolutionary algorithms and a fractal inverse problem. *Biosystems*, 33:221–231, 1994.

[125] A.E. Nix and M.D. Vose. Modeling genetic algorithms with markov chains. *Annals of Mathematics and Artificial Intelligence*, 5(1):79–88, 1992.

[126] S. Obayashi. Pareto genetic algorithm for aerodynamic design using the Navier-Stokes equations. In D. Quadraglia, J. Périaux, C. Poloni, and G. Winter, editors, *Genetic Algorithms and Evolution Strategies in Engineering and Computer Sciences*, pages 245–266. John Wiley, 1997.

[127] S. Osher and J. A. Sethian. Front propagating with curvature dependent speed : algorithms based on hamilton-jacobi formulations. *J. Comp. Phys.*, 78:12–49, 1988.

[128] B. Paechter, R.C. Rankin, A. Cumming, and T. C. Fogarty. Timetabling the classes of an entire university with an evolutionary algorithm. In A.-E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Proceedings of the* 5^{th} *Conference on Parallel Problems Solving from Nature*. Springer Verlag, 1998.

[129] P. Pedersen and C. L. Laursen. Design for minimum stress concentration by finite elements and linear programming. *J. Structural Mech.*, 10(4):375–391, 1983.

[130] O. Pironneau. *Optimal Shape Design for Elliptic Systems*. Springer-Verlag, 1984.

[131] B. Prasad and R. T. Haftka. Optimal structural design with plate finite elements. *ASCE J. Structural Div.*, 105(11):2367–2382, 1979.

[132] F. P. Preparata and M. I. Shamos. *Computational Geometry : An introduction.* Springer Verlag, 1985.

[133] A. Pétrowski. A clearing procedure as a niching method for genetic algorithms. In *Proceedings of the Third IEEE International Conference on Evolutionary Computation*, pages 798–803. IEEE Press, 1996.

[134] 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.

[135] N. J. Radcliffe and P. D. Surry. Real representations. In L. D. Whitley and R. K. Belew, editors, *Foundations of Genetic Algorithms 4*, pages 51–72. Morgan Kaufmann, 1997.

[136] J. Rasmussen and N. Olhoff. *Structural Optimization-Status and Promise*, chapter Status and Promise of Optimum Design System in Denmark. Kamat, M., 1992.

[137] F. Raynal. *Etudes d'outils pour la dissimulation d'information : Approche fractales, protocoles d'évaluation et protocoles cryptographiques.* PhD thesis, Université Paris XI, Mars 2002.

[138] I. Rechenberg. Cybernetic solution path of an experimental problem. *Roy. Aircr. Estab.*, 1965. libr. trnsl. 1122. Farnborough, Hants., UK.

[139] I. Rechenberg. *Evolutionstrategie: Optimierung Technisher Systeme nach Prinzipien des Biologischen Evolution.* Fromman-Hozlboog Verlag, Stuttgart, 1973.

[140] B. Rousselet. *Quelques résultats en optimisation de domaines.* Université de Nice- Sophia-Antipolis, 1982. Thèse de Doctorat d'Etat.

[141] G. Rudolph. Convergence analysis of canonical genetic algorithm. *IEEE Transactions on Neural Networks*, 5(1):96–101, 1994.

[142] G. Rudolph. *Convergence Properties of Evolutionary Algorithms.* Kovac, Hamburg, 1997.

[143] Conor Ryan, J. J. Collins, and Michael O Neill. Grammatical evolution: Evolving programs for an arbitrary language. In Wolfgang Banzhaf, Riccardo Poli, Marc Schoenauer, and Terence C. Fogarty, editors, *Proceedings of the First European Workshop on Genetic Programming*, volume 1391 of *LNCS*, pages 83–95, Paris, 14-15 April 1998. Springer-Verlag.

[144] B. Sareni and L. Krähenbühl. Fitness sharing and niching methods revisited. *Transactions on Evolutionary Computation*, 2(3):97–106, 1998.

[145] D.J Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In J. J. Grefenstette, editor, *Proceedings of the* 1^{st} *International Conference on Genetic Algorithms.* Laurence Erlbaum Associates, 1985.

[146] L. A. Schmit. Structural design by systematic synthesis. In *Proceedings of the 2nd Conference on Electronic Computation*, pages 105–122, New-York, 1960. ASCE.

[147] 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.

[148] M. Schoenauer. Shape representation and evolution schemes. In L. J. Fogel, P. J. Angeline, and T. Bäck, editors, *Proceeding of the* 5^{th} *Annual Conference on Evolutionary Programming.* Mit Press, 1996.

[149] M. Schoenauer, A. Ehinger, and B. Braunschweig. Non-parametric identification of geological models. In T. Fukuda, editor, *Proceedings of the Third IEEE International Conference on Evolutionary Computation*. IEEE Press, 1998.

[150] M. Schoenauer, L. Kallel, and F. Jouve. Mechanical inclusions identification by evolutionary computation. *Europeen Journal of Finite Elements*, 5(5-6):619–648, 1996.

[151] M. Schoenauer and Z. Michalewicz. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 1(4):1–32, 1996.

[152] M. Schoenauer and Z. Michalewicz. Evolutionary computation at the edge of feasibility. 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*, pages 245–254. Springer-Verlag, LNCS 1141, 1996.

[153] M. Schoenauer and Z. Michalewicz. Evolutionary computation. *Control and Cybernetics :Evolutionary Computation*, 26(3):307–338, 1997.

[154] M. Schoenauer, M. Sebag, F. Jouve, B. Lamy, and H. Maitournam. Evolutionary identification of macro-mechanical models. In P. J. Angeline and Jr K. E. Kinnear, editors, *Advances in Genetic Programming II*, pages 467–488, Cambridge, MA, 1996. MIT Press.

[155] M. Schoenauer and Z. Wu. Discrete optimal design of structures by genetic algorithms. In Bernadou and al., editors, *Conférence Nationale sur le Calcul de Structures*, pages 833–842. Hermes, 1993.

[156] A. Schumacher. *Topologieoptimierung von Bauteistrukturen unter Verwendung von Lopchpositionierungkrieterien*. PhD thesis, Universität-Gesambthochschule-Siegen, 1995.

[157] H.-P. Schwefel. Numerische optimierung von computer-modellen mittels der evolutionsstrategie. *Interdisiplinary systems research*, 26, 1977.

[158] H.-P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley & Sons, New-York, 2nd edition, 1995.

[159] M. Sebag, M. Schoenauer, and M.H. Maitournam. Evolutionary identification of rheological models. In A. Niku-Lari, editor, *EXPERSYS 96 (Expert Systems Applications in Artificial Intelligence)*, pages 465–470. IITT International, 1996.

[160] J. A. Sethian. Level set methods and fast marching methods : evolving interfaces in computational geometry. *fluid mechanics, computer vision and materials science*, 1999.

[161] O. Sigmund. *Design of Material Structures Using Topology Optimization*. PhD thesis, Technical University of Denmark, Dept. of Solids Mechanics, 1994.

[162] O. Sigmund. On the design of compliant mechanisms using topology optimization. *Mech. Struct. Mach*, 25, 1997.

[163] A. Smith and D. Tate. Genetic optimization using a penalty function. In S. Forrest, editor, *Proceedings of the* 5^{th} *International Conference on Genetic Algorithms*, pages 499–503. Morgan Kaufmann, 1993.

[164] N. Srinivas and K. Deb. Multi-objective function optimization using non-dominated sorting genetic algorithms. *Evolutionary Computation Journal*, 2(3):221–248, 1994.

[165] 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, year = 1996, number 1141 in LNCS, pages 366–375. sv.

[166] K. Suzuki and N. Kikuchi. A homogenization method for shape and topology optimization. *Comp. Methods Appl. Mech. Eng.*, 93:291–318, 1991.

[167] G. Syswerda. Uniform crossover in genetic algorithms. In J. D. Schaffer, editor, *Proceedings of the 3^{rd} International Conference on Genetic Algorithms*, pages 2–9. Morgan Kaufmann, 1989.

[168] L. Tartar. Estimation des coefficients homogénéises. In R. Glowinski and J. L. Lions, editors, *Computing Methods in Applied Sciences and Engineering*, volume 704 of *Lecture Notes in Math.*, pages 364–373. Springer, 1978.

[169] Astro Teller and Manuela Veloso. PADO: A new learning architecture for object recognition. In Katsushi Ikeuchi and Manuela Veloso, editors, *Symbolic Visual Learning*, pages 81–116. Oxford University Press, 1996.

[170] J. Lévy Véhel. *Analyse et synthèse d'objets bi-dimensionnels par des méthodes stochastiques.* PhD thesis, Université de Paris Sud, 1988.

[171] Jacques Lévy Véhel, Khalid Daoudi, and Evelyne Lutton. Fractal modeling of speech signals. *Fractals*, 2(3):379–382, September 1994.

[172] M.D. Vose. *The Simple Genetic Algorithm: foundations and theory.* MIT Press, 1999.

[173] R. Vrscay. Moment and collage methods for the inverse problem of fractal construction with iterated function systems. In *Fractal 90 Conference*, 1990.

[174] D. Whitley. Fundamental principles of deception in genetic search. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms.* Morgan Kaufmann, 1991.

[175] X. Yin and N. Germay. A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In *Proceedings of the Artificial Neural Nets and Genetic Algorithms*, 1993.

[176] A. Zalzala, editor. *Genetic ALgorithms in Engineering Systems: Innovations and Applications.* IEE, 1995.

[177] O. C. Zienkiewicz and J. C. Campbell. Shape optimization and sequential linear programming. In R. H. Gallagher and O. C. Zienkiewicz, editors, *Optimum Structural Design*, pages 109–126. Wiley, New York, 1973.

[178] E. Zitzler and L. Thiele. Multi-objective optimization using evolutionary algorithms: A comparative case study. In A.-E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Proceedings of the 5^{th} Conference on Parallel Problems Solving from Nature*, pages 292–301. Springer-Verlag, LNCS 1498, 1998.

[179] J. P. Zolesio and J. Sokolowski. *Introduction to Shape Optimization.* Springer-Verlag, 1992.