LaTeX
Sample Output

eurogen 6/10/1995 14:29—PAGE PROOFS for John Wiley & Sons Ltd (jwbook.sty v3.0, 12-1-1995)

$\LaTeX$ Sample Output

First Edition

J. SMITH
Text Processing, John Wiley & Sons Ltd, Chichester, West Sussex

JOHN WILEY & SONS
Chichester · New York · Brisbane · Toronto · Singapore

Contents

22 Shape Representations for Evolutionary Optimization and Identification in Structural Mechanics vii
22.1 Introduction vii
22.2 Bit-array representation ix
22.2.1 Bit-arrays are not bit-strings ix
22.2.2 Crossover operators x
22.3 Topological Optimal Design xiii
22.3.1 The mechanical problem xiii
22.3.2 The fitness function xiv
22.3.3 Deterministic methods xiv
22.3.4 From genotype to phenotype xv
22.4 Evolutionary Optimal Design with Bit-array Representation . . xvi
22.4.1 Results in linear elasticity xvii
22.4.2 Nonlinear mechanical model xvii
22.4.3 The need for another representation xix
22.5 Two new representations xx
22.5.1 The "holes" representation xx
22.5.2 Voronoï-based representation xxii
22.6 First comparative results xxiv
22.6.1 Strong vs weak inheritance xxiv
22.7 Conclusion: Evolutionary Computation and Structural Mechanics xxv
22.8 Summary xxvi

References xxix

22

Shape Representations for Evolutionary Optimization and Identification in Structural Mechanics

Marc Schoenauer [1]

22.1 Introduction

This paper presents an Evolutionary approach to problems of shape optimization and identification: the aim is to find a partition of a given *design domain* of the 2D-plane or the 3D-space into two subsets (e.g. material and void for the Optimal Design problems). The central issue of this paper is the representation of such repartition suitable for its evolutionary optimization, and its dependency with respect to the problem at hand.

Evolutionary Computation (EC) is a field that now regroups different stochastic optimization methods based on the same paradigm: *evolve* a *population* (set of fixed size) of *individuals* (points of the search space) using stochastic operators (e.g. *crossover* and *mutation*) in a crude parody of the Darwinian principle of *survival of the fittest*.

The main streams of EC are Genetic Algorithms (GAs), first described by [Hol75] and popularized by [Gol89a], Evolution Strategies (ESs), introduced by [Rec73, Sch81], and Evolutionary Programming (EP), imagined by [FOW66]. A good overview of EC can be found in the Proceedings of the first International IEEE Conference on the subject [ZMR94], or in [Fog95].

One of the main differences between these streams lies in the representation

[1] Centre de Mathématiques Appliquées, CNRS-URA756 - Ecole Polytechnique - 91128 Palaiseau - E-mail : Marc.Schoenauer@polytechnique.fr

issue: historically, GAs deal with bit-strings of fixed sizes, ESs with real vectors and EP with Finite State Automata. Another point of discussion,somewhat related to the representation issue, lies in the type of approach: *bottom-up* for GAs vs *top-down* for both ESs and EP: GAs basic assumption is that the optimal solution can be found by assembling *building blocks*, i.e. partial pieces of solutions, while ESs and EP assess for the environmental pressure to ensure the emergence of the best solutions. The most visible consequence of this debate deals with the recombination operator, viewed as essential for GAs, as potentially useful for ES, and as possibly harmful for EP.

The "modern" tendencies are more pragmatic, and put the discussions back to the representation issue: GAs users have turned to real number representations when dealing with real numbers, following experimental results by [JM91] and heuristic demonstrations by [Rad91]. ESs researchers have included recombination as a standard operator, as in [Sch81], and are designing specific operators for non real-valued problems (e.g. in [BS95]). And EP is being used on any representation, as for instance by [Ang93, FS95]. But the whole EC community now takes for granted that the definition of both the genotypic space (the space where the evolution operators apply) and the phenotypic space (the space on which the fitness is computed, or behavioral space) plays a central role in determining the success or failure of an EC optimization.

This paper introduces representations for 2-dimensional shapes, i.e. partitions of a compact set of $I\!R^2$ into two subsets, and discusses their properties in the light of their use for optimization by Genetic Algorithms. Section 22.2 first presents the "GA-natural" bit-oriented representation. Though standard bit-string crossover operators apply to that representation, the need to design specific crossover operators is emphasized. Such operators are introduced and experimental comparisons with the standard 1- and 2-points operators are performed. The Mechanical problem of Topological Optimal Design is then presented in section 22.3, and the design of the fitness function is carefully detailed, guided by some mechanical considerations. Section 22.4 then presents results obtained by GAs using this bit-oriented representation: in situations where standard methods are not applicable, a real breakthrough in Topological Shape Optimization is achieved. But these results also point out the limits of this bit-oriented representation, namely that it heavily depends on the a priori discretization of the design domain. So two other representations are introduced in section 22.5. Both representations overcome the dependency on a fixed discretization by allowing their complexity to evolve. They differ from the bit-array representation in the amount of *degeneracy* they contain, as defined in [Rad94]. Comparative results on the Optimal Design problem are given in section 22.6: both representation outperform the bit-oriented representation, and the representation in which genetic traits are strongly transmitted seems to give better results. Finally, section 22.7 discusses the pros and cons for EC in the field of Structural Mechanics, and presents on-going and further work.

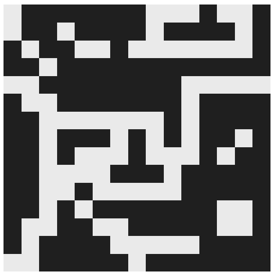
22.2 Bit-array representation

[2] When the goal is to find an optimal shape, i.e. a partition of the design domain into two subsets, the most "natural" representation for using GAs is a bit-oriented representation based on a regular discretization of the design domain: First, GAs were initially designed to work on bit-strings; Second, the application problems are Structural Mechanics problems, usually involving Finite Elements Analyzes, based on some discretization of the domain at hand. And such a discretization induces some error on the Analysis. A way to minimize the influence of this error is to work with a constant mesh, without re-meshing every shape. And a straightforward representation for a shape is then, for a given mesh, the corresponding bit-string, with 0 and 1 representing each one of the two subsets of the desired partition.

Figure 1 gives an example of such a representation for shapes in a square domain design, discretized here into 15×15 square elements, and the corresponding partition of the design domain.

```
100000001110110
100100001000010
010011011111110
001000000000000
110000000011111
011000000010000
001111111010000
001000101010010
001011101110100
001111000100000
001101111100000
001010000000110
011001100000110
010000111110000
110000010000000
```

a - *The Bit-array genotype.*



b - *The phenotype.*

Figure 1 Shape representation: GA applies on the bit-array genotype (a), and the corresponding phenotype is the partition of the design domain (b).

22.2.1 Bit-arrays are not bit-strings

This section discusses the differences between linear bit-strings and 2-dimensional bit-arrays. Specific crossover operators are introduced to account for that difference.

[2] All results based on that representation were obtained by C. Kane, and are part of her PhD dissertation [Kan95].

Schema analysis

By analogy with the bit-string case, schema can be defined in bit-arrays by adding the "*" symbol to the binary alphabet. But the analogy fails when it comes to define the building blocks, or short low-order schemata with above average fitness ([Gol89a]). Figure 2 illustrates such situations. From a geometrical point of view, the schema of Figure 2-a should be considered as a potential building block. But its emergence (if we suppose it is related to highly fit individuals) is not likely to happen in the early generations, if only one-dimensional crossover operators are used: its one-dimensional order is 31. On the opposite, the schema of Figure 2-b is erroneously considered as a low order schema (of order 2), although it is hardly related to the fitness.

```
1..............   ..............1
1..............   1..............
1..............   ...............
...............   ...............
...............   ...............
      (a)               (b)
```

Figure 2 Differences between one- and two-dimensional defining lengths of schemata. Schema (a) has a two-dimensional length of 3, but a one-dimensional length of 31. Schema (b) has a two-dimensional length of 15, but a one-dimensional length of 2.

More generally, when bit-arrays are handled as bit-strings, one-dimensional cross-over operators induce a strong geometrical bias against vertical building blocks.

A quantification of this bias remains to be done, for instance by studying the variance of the fitness on schemas, as a function of the schema length (in the line of what is done in [Rad94] for evaluating various representations of the TSP problem).

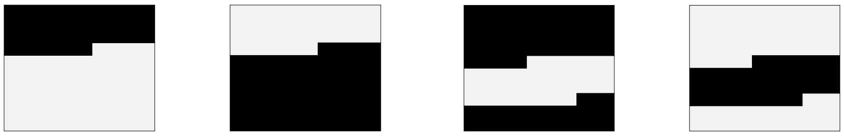
Nevertheless, the expected inaccuracy of classical n-point crossover (confirmed by experimental results, section 22.2.2) leads to design specific genetic operators, suited to our two-dimensional problem.

22.2.2 Crossover operators

This section is devoted to the study of the crossover operators for the bit-array representation. All crossover operators presented in this section are symbolically drawn on Figure 3 and 4: two offsprings of a black parent and a grey parent are plotted. The color of the mesh elements of the children only tells which parent this element comes from, regardless of its actual value (0 or 1).

Standard crossover operators

In order to evaluate the bias induced by handling a bit-array as a bit-string, we first consider the standard crossover operators; One-point, two-points [Gol89a] and uniform [Sys89] crossovers are considered.



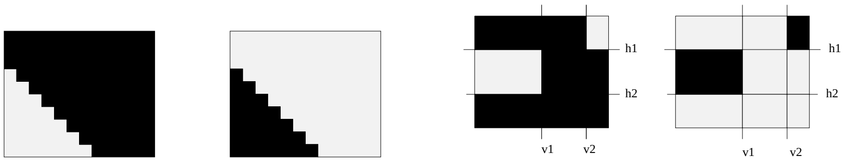
a - *1 point standard* b - *2 point standard*

Figure 3 Examples of offsprings from one dimensional crossover operators

As pointed out in section. 22.2.1, one- and two-points crossovers are geometrically biased: only horizontal bands of the parents get exchanged. Experimental results without mutation presented in next subsection witness for the bad influence of this bias. On the opposite, the uniform crossover (easy to imagine, though not presented in Figure 3) does not suffer from such a bias.

Specific two-dimensional crossover operators

Two specific two-dimensional operators are introduced and studied in this section.



a - *diagonal* b - *2-block*

Figure 4 Examples of offsprings from 2-dimensional crossover operators

Diagonal crossover.
The basic idea of diagonal crossover is to generalize the popular one-point crossover to the two-dimensional case. As shown on Figure 4-a, a randomly selected straight line separates the rectangle in two parts which are exchanged between both parents. A generalization of this operator has been recently introduced in [KM95].

The block crossover
First introduced in [Jen92], the idea of the block crossover is to cut the whole two-dimensional domain by two horizontal lines and two vertical lines and to exchange some of the large blocks defined by these lines. The values v_1 and v_2 (respectively h_1 and h_2) are chosen uniformly, and the 2 (or 3) blocks to

be exchanged are selected randomly. Figure 4-b shows an example of 2-block crossover.

Comparison without mutation

These operators have been experimentally compared on an instance of the problem described in next section. Figure 5 shows the results obtained without any mutation, and with a crossover rate of 1. Of course, these results give only hints that something goes wrong with standard crossover operators: the absence of mutation forbids to draw any definite conclusion. Nevertheless, such result confirm the points discussed in section 22.2.1.

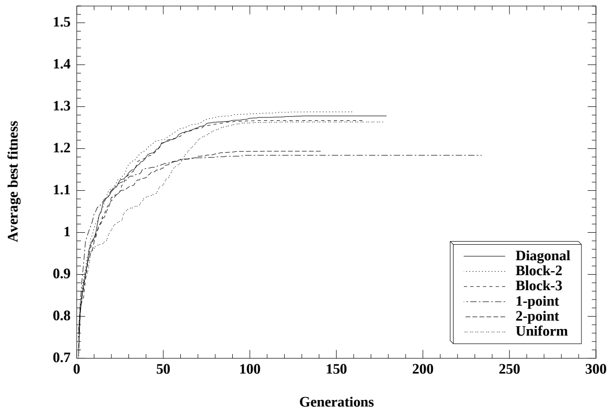


Figure 5 Performances of recombination operators without mutation on an Optimal Design Problem.

The first conclusion that can be drawn from Figure 5 is the ineffectiveness of the one-dimensional crossover operators, similarly outperformed by both uniform and two-dimensional crossovers.

The second remark is that the uniform operator performs poorly in the early stage of the runs, before catching up in the late generations. The explanation might be the following: in the beginning of a run, the uniform crossover equivalently disrupts schemas of any order, while all other crossovers (even one dimensional) preserve some large areas. These disruptive effects do not occur in the end of evolution, since neither kind of crossover does disturb regions where convergence already occurred: if the bits of the parents at a given position are the same, crossing over has no effect on those bits. In the meantime, the one-dimensional crossovers fail to beneficially exchange schema whose vertical precision is small compared to the horizontal one, like long vertical bars.

22.3 Topological Optimal Design

The shape representations presented in this paper can apply to different problems of Mechanical Engineering: Topological Optimal Design, where the optimization of a structure whose topology is not the continuous transformation of a given shape; Inclusion Identification, where a structure is made of an unknown repartition of two materials, and this unknown repartition is to be identified from a given set of experiments on that structure. This latter problem will not be addressed in this paper, though the first results obtained using the representations presented here are very good indeed (see [JS95]).

This section is devoted to the detailed presentation of the Problem of Topological Optimal Design, on the impact of this problem on the representation and on the design of the fitness function.

22.3.1 The mechanical problem

The goal is to find a design for a structure that minimizes some cost function (usually the weight of the structure, but it can include as well technological costs) while respecting some given constraints on mechanical behaviors (such as limit values for the displacement, the stress, and so forth) under some given loadings. The design is sought inside the design domain, a given subset of $\mathbb{R}^2$ or $\mathbb{R}^3$.

We shall restrict here to 2-dimensional cantilever problems, to plane strain model, and elastic or hyper-elastic materials. The design domain is a rectangular subset of $\mathbb{R}^2$, the plate is fixed on its left boundary, the loading is made of a single force applied at some point of the right boundary (see Figure 6), and a constraint is imposed on the displacement of that loading point.

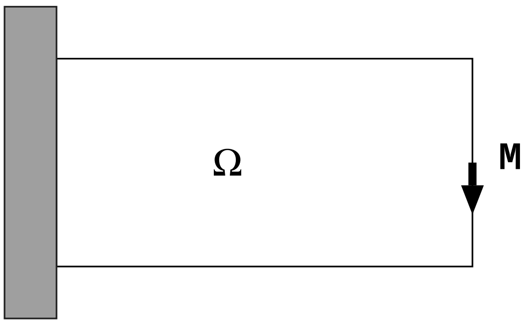


Figure 6 The 1×2 cantilever plate. Ω is the design domain, the plate is fixed on the $x = 0$ axis, and a force is applied at point M.

22.3.2 The fitness function

The problem here is a constrained problem. Many methods have been designed and used to take constraints into account in evolutionary algorithms (see [Mic95] for a survey of such methods). In particular, the method described in [SX93] has been successfully applied to another problem of structural mechanics, the optimization of truss structures. But our point here is to study the influence of representation, not the way to handle constraints. Therefore, the standard method of penalization has been chosen, as a non specific and quite robust way of handling constraints.

In summary, the first draft for the fitness function (to be minimized here) is computed from the weight of the structure (or its area in 2 dimensions) and the maximal value of the displacement, obtained through a Finite Element Analysis. It has the following expression:

$$\mathcal{F} = Area + \alpha(D_{Max} - D_{Lim})^+ \tag{22.1}$$

where D_{Max} is the maximal displacement of the structure, D_{Lim} the limit value for the displacement and α is a positive user-supplied parameter (a^+ denotes the positive part of a). Some mechanical difficulties detailed in next sections will be addressed by modifying this simple expression for the fitness.

22.3.3 Deterministic methods

There is only one known deterministic method for topological shape optimization, namely the homogenization method. It is based on a relaxation of the problem considering not only 0/1 repartition of void/material, but any continuous "mixing" of void and material with density in $[0, 1]$. Initiated by [BK88], it has been studied theoretically by [AK93], and wonderful results have been obtained by [ABFJ95], including 3-dimensional results. But the limits of that method are well-known.

- The homogenization method searches solutions in the space of composite materials having a density of material taking continuous values in $[0, 1]$. Hence, the solutions found have to be projected on feasible part (with density of material either 0 or 1). This projection step does not always succeed in giving feasible solutions.
- The only mechanical context in which the homogenization applies is that of linear elasticity;
- It gives only one solution, even when multiple optimal solutions exist;
- It is limited to a single loading;
- It cannot handle loadings on the unknown boundary.

A more extensive discussion between the homogenization method and the GA approach, together with some comparative results, can be found in [KJS95].

22.3.4 From genotype to phenotype

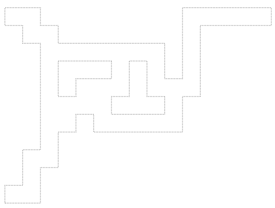
We have to reconsider the definition of the phenotype given in section 22.2 in the light of the mechanical problem at hand: a solution to that problem is a shape of the design domain *that link the loading point(s) to the fixed boundary*, otherwise the structure is statically unstable, that is not viable (any Finite Element Analysis on such structure will abort abruptly). The first difficulty for representing shapes of structures is that possible shapes (repartition void/material) do not correspond to viable structures. Moreover, only the material parts connected to both the fixed boundary and the loading point have something to do with the underlying mechanical problem. So two different shapes can actually represent the same structure.

The first difficulty can be overcome by giving null fitness to non viable structures. But there is a risk that almost all random structures are non viable (for long narrow shapes for instance). A better approach is to smoothly increase the cost function as the gap between the fixed boundary and the connected component containing the loading point increases, thus favoring almost connected structures.

But the second difficulty raises the problem of the *degeneracy* of the representation with respect to the problem at hand, as defined in [Rad94]: a representation is said to be degenerate when the application from the genotypic space into the phenotypic space is not injective. Figure 7 gives a new look at the example presented in Figure 1. Any genotype belonging to the genotypic schema represented in Figure 7-a (i.e. having any value replacing the "*" signs) correspond to the same structure (Figure 7-b). So even though the bit-array representation is not degenerate to represent partitions of the design domain into void and material, it becomes degenerate when it comes to represent actual structures, if no precaution is taken when designing the fitness.

```
***************
***************
***************
00********00000
110******011111
011000000010000
*01111111010***
*01000101010***
*01011101110***
*0111100010****
*0110111110****
*010*00000*****
0110***********
010************
110************
```

a - *The genotypic schemata.*



b - *The actual structure.*

Figure 7 An example of degeneracy: The Finite Element Analysis applies on structure (b), to compute the fitness of any genotype of schemata (a).

There are 3 ways to handle this problem-dependent degeneracy:

- Replace the genotype by that of the mechanical structure, removing all unconnected bits. But possibly valuable genetic material would then be lost, therefore resulting in bad performance of the genetic search.
- Ignore the degeneracy, using the fitness defined in section 22.3.2. But degeneracy is acknowledged to be generally bad for GA optimization.
- Account for the unconnected material in the fitness function, thus getting rid of the degeneracy.

This last option was chosen here, and the draft fitness function (22.1) is modified and becomes, for viable shapes:

$$\mathcal{F} = Area_{usefull} + \lambda Area_{useless} + \alpha(D_{Max} - D_{Lim})^{+} \qquad (22.2)$$

Of course, the choice of λ is critical: too high, some actual optima can be lost. Too low, it has no effect. In the following, a value of 0.1, experimentally determined, was used for all experiments presented in the following.

22.4 Evolutionary Optimal Design with Bit-array Representation

The results presented in this section definitely witness for the power of the evolutionary approach on the Optimal Design problem: such results are a real breakthrough in Optimal Design. The price to pay is what was expected in terms of computational cost.

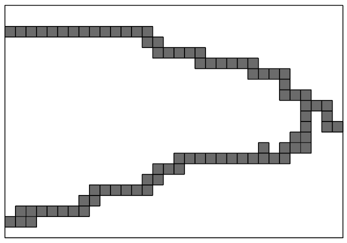
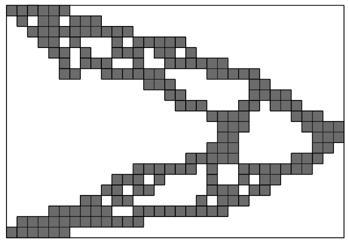
		
D_{Lim}	2.463	0.1293
D_{Max}	2.439	0.129286
$Area$	0.2075	0.4675

Figure 8 Optimal cantilever plates for different values of the displacement constraint D_{Lim}. D_{Max} is the actual maximum displacement of the structure.

22.4.1 Results in linear elasticity

In order to validate the evolutionary approach, the linear elasticity model is used on the standard 2×1 benchmark cantilever plate. The results presented in Figure 8 are the best results obtained out of 10 independent runs. The plate is discretized according to a 32×22 regular mesh. The population size for all runs is 125, and the number of generations arbitrarily fixed to 1000. One run thus requires about 100000 Finite Element Analyzes, taking approximately 6 hours of a powerful HP workstation for the 32×22 discretization. The block crossover is applied at a rate of 0.6, and the standard mutation (presenting no geometrical bias) at a rate of 0.01. All these parameters were adjusted after exhaustive tests (see [Kan95]).

These first results can be obtained more precisely and more rapidly (up to 2 orders of magnitude) by the homogenization method, and only demonstrate the validity of the approach.

Further results in linear elasticity deal with problems the homogenization method cannot handle (nor any other method, as far as we know): multiple optimal or quasi-optimal solutions can be found [KJS95], simultaneous multiple loadings can be considered, including uniform pressure on the unknown boundary [Kan95].

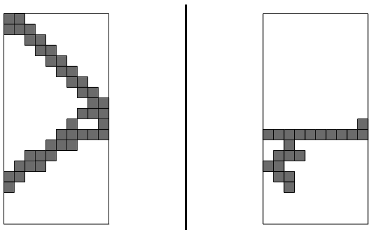
But the most important field that opens to optimization is that of nonlinear elasticity. From the GA point of view, it seems at first sight a straightforward adaptation of the linear case: replace the linear Finite Element Analyzes by Nonlinear Finite Element Analyzes. The following section demonstrates how such an approach might lead to erroneous results.

22.4.2 Nonlinear mechanical model

This section considers standard plane stress problems in the context of large displacements. The material still obeys a linear law (the extension to any other law is straightforward), but the nonlinear geometric effects due to the large displacement hypothesis are taken into account. A thorough description of the theoretical model together with the numerical algorithm can be found in [Cia88]. Details on the Finite Element Method and its implementation used here can be found in [Jou93].

The first experiments use the penalized fitness function 22.2. The idea was to use different loadings $\mathcal{F}$, with a fixed ratio $\mathcal{F}/D_{Lim}$ (D_{Lim} is the constraint on the displacement). In the purely linear case, all such problems are of course equivalent. The problem is the standard benchmark 1×2 cantilever problem described in section 22.3 (the plate is fixed on its left side only).

A typical disastrous result of the GA based optimization in the large displacement context is shown in Figure 9-b, together with the result on the same problem in the pure linear model (Figure 9-a). And different values of $\mathcal{F}$ and D_{Lim} do produce similar results. It is disastrous at first sight for any Mechanical engineer, as having a too large stress at the point where the structure becomes thin. A closer look at the maximum value of the stress of the solution σ_{Max} confirms that first diagnosis. The reasons for that are both: First, the large displacement model can have different solutions for the same loading, some with higher stress than others. Second, the stress field itself, on a rough domain like the structures obtained for the cantilever plate problem, and with such coarse discretization, presents some singularities. The value of σ_{Max} in Figure 9-b witnesses for that phenomenon, being more than an order of magnitude higher than those of the linear case (for the same loading).



D_{Max}	0.022607	0.0199
σ_{Max}	0.076	0.77
$Area$	0.41	0.20
	(a): small displacements.	(b): large displacements.

Figure 9 Disastrous optimal designs found by GAs with displacement constraints. $\mathcal{F} = 0.009$ and $D_{Lim} = 0.02285$.

This suggests to incorporate a constraint on the maximal stress in the fitness function. The fitness 22.2 (to be minimized) becomes

$$Area_{usefull} + \lambda Area_{useless} + \alpha(D_{Max} - D_{Lim})^{+} + \beta(\sigma_{Max} - \sigma_{Lim})^{+} \quad (22.3)$$

where σ_{Lim} is the maximal stress allowed, σ_{Max} is the maximal stress (computed) of the structure, and β is some positive penalty parameter which has to be adjusted.

Figure 10 shows the optimal designs obtained with that modified fitness: Reasonable solutions are found, as long as the stress is imposed a strong constraint.

Moreover, the nonlinear effects are clear: all experiments of Figure 10 would have given the same result in the linear case, as the ratio $\mathcal{F}/D_{Lim}$ is constant and no multiple solution was ever observed on the simple 1×2 cantilever problem.

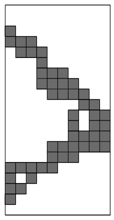
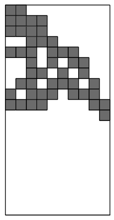
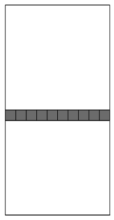
			
$\mathcal{F}$	0.009	0.018	0.09
D_{Lim}	0.22856	0.457	2.2856
σ_{Lim}	0.53	1.0622	5.3
D_{Max}	0.2143	0.4504	1.687
σ_{Max}	0.550	0.9835	4.379
$Area$	0.21	0.47	0.1

Figure 10 Optimal designs for nonlinear elasticity with displacement and stress constraints, with fixed ratio $\mathcal{F}/D_{Lim}$.

22.4.3 The need for another representation

The preceding subsections have demonstrated the power of the evolutionary optimization of structures using the bit-array representation. Many dead-ends for the homogenization method have been put wide open:

- multiple solutions can be found when they exist
- any mechanical model can be used
- multiple loading, or loading on the unknown boundary, can be treated (though not presented here)

But the limits of the method also crudely appear, when it comes to refine the mesh (rather coarse in the results presented), or when we think of 3-dimensional problems: in both cases, the size of the individual will grow tremendously, and probably make the problem intractable for any evolutionary algorithm. In order to overcome this limitation, two other representations are now presented in the next section. They will then be compared to the bit-array representation, both by the cost of the resulting optimization process and by the accuracy of the solution.

22.5 Two new representations

22.5.1 The "holes" representation

Consider the design domain to be made of plain material, and imagine cutting holes into it. Neither the number of holes nor their size is fixed. The result of such cutting is a shape included in the design domain.

More precisely, we restrict ourselves to rectangular holes (merely for technical reasons), defined by their centers and side lengths. An individual is then a number R of such holes, and the description of the R holes. The corresponding shape is easily computed by superposing the holes in the design domain (see Figure 11).

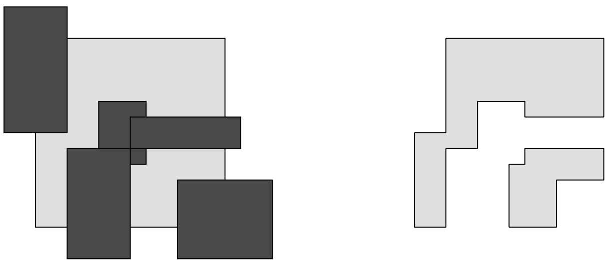


Figure 11 Five rectangular holes, and the corresponding structure.

The advantages of such a representation over the bit-array representation described in section 22.2 are quite clear.

- This representation is totally independent of any discretization of the design domain; Of course, the shape obtained from this representation will be discretized during the Finite Element Analysis. But the size of the genotypes is totally independent of this discretization;
- Large areas can be described by a few numbers of holes: the complexity of the representation is not fixed. Moreover, adaptative complexity can be achieved by including evolutionary operators that change the number of holes of an individual.

Genetic operators for the holes representation

The crossover operator is a straightforward adaptation of the *diagonal* crossover presented for the bit-array representation in section 22.2.1: a random line is drawn across the design domain, and the holes whose center are on one side of this line are exchanged between the parents (Figure 12).

There are 4 mutation operators:

- Add a hole

- Delete a hole
- Move a hole. The coordinates of the center of the hole are moved by addition of a Gaussian function, whose standard deviation depends on the fitness of the structure at hand, in an EP-like way [3].
- Change the size of a hole. Here again both sides of the rectangular holes are changed according to a Gaussian law, with the same remarks than for the "move" mutation.

The probability of application of each mutation operator is governed by user-supplied weights (only one at the time is applied).

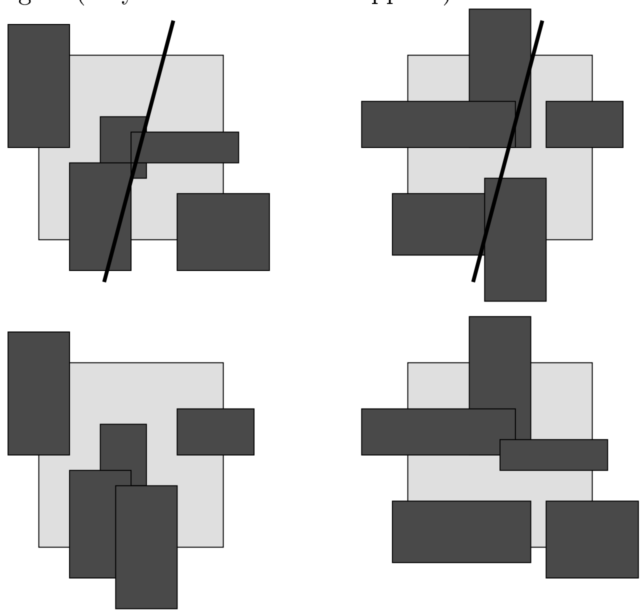


Figure 12 Example of crossover between two holes-represented structures.

Degeneracy of the holes representation

The holes representation clearly contains some degeneracy: some rectangular holes might be completely included in the union of other holes. Of course, another mutation operator could be used, aiming at deleting useless holes, i.e. holes that are themselves included in other unions of holes; but even without useless holes, the holes representation would remain highly degenerate, as different rectangles configurations can give the same shape without any useless hole.

[3] Adaptative mutation by having each hole carry its standard deviation, in an ES-way, is the following step for smart mutations

Another argument go against the holes representation: diagonal shape boundaries require a lot of holes (but allowing triangular holes would probably fix that problem, and is on-going work).

Anyway, the Voronoï representation, introduced in next section, is an attempt to overcome these a priori weaknesses of the holes representation.

22.5.2 Voronoï-based representation

Voronoï diagrams provide nice way to describe any partition of the design domain into polygonal elements.

Voronoï diagrams

Given some points $S_1, \ldots, S_l$ of $I\!R^n$ called the *sites*, the Voronoï diagram defined by these sites is a partition of the space into convex polygons, or *cells*, each cell being defined as

$$Cell(S_i) = \{M \in I\!R^n; d(M, S_i) = min_{i=j,n} d(M, S_j)\}$$

To define a shape from a Voronoï diagram, it is sufficient to give each site (thus each cell) a color. Figure 13 gives an example of a shape, together with the associated Voronoï diagram.

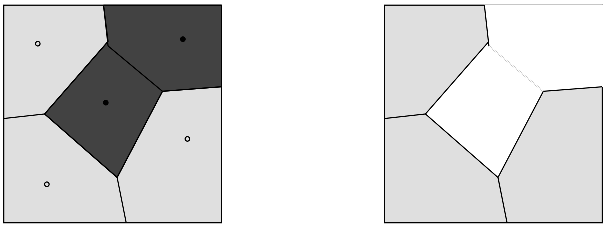


Figure 13 A Voronoï diagram with 5 sites, and the corresponding partition of the design domain.

Genetic operators

The genetic operators for the Voronoï representation look much like those for the holes representation: the crossover operator uses the same *diagonal* random line to exchange Voronoï sites between the parents. The mutation operators are here only three, namely the "add", "delete" and "move" operator, all acting on the Voronoï sites. Figure 14 shows an example of the crossover operator. An important remark is that each operator for the Voronoï representation is more global that the corresponding operator for the holes representation, due to the fact that each Voronoï cell actually depends on all sites surrounding it, where a hole is not modified whatever modification happen to other holes.

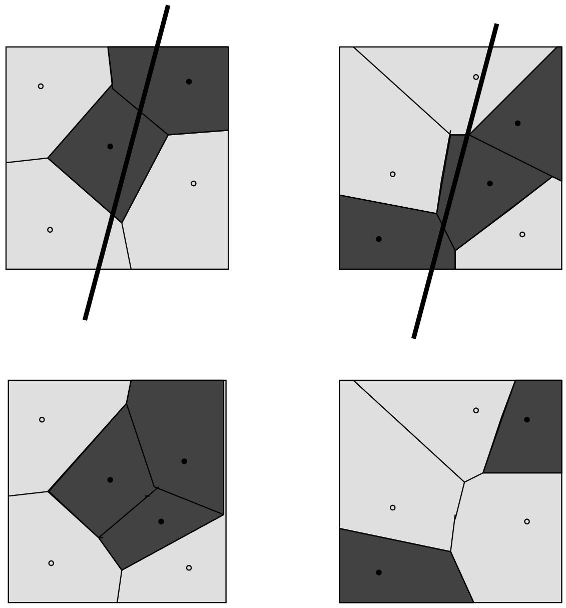


Figure 14 Example of crossover between two Voronoï diagrams.

Degeneracy of the Voronoï representation

The amount of degeneracy in the Voronoï representation is clearly not as obvious as for the holes representation. Nevertheless, it does happen that some sites are useless: they are surrounded by sites of the same color, and the resulting shape is not modified if they are removed. As in the holes representation, these sites could be removed, either systematically, or by some mutation operator, though at a somewhat high computational cost. And a Voronoï representation without useless sites is indeed a non degenerate representation.

The following section compares the new representations (the holes and the Voronoï representations) with the bit-array representation of section 22.2. The implementations of the holes and the Voronoï representations are straightforward; No attempt to remove any degeneracy is yet implemented.

22.6 First comparative results

A comprehensive experimental comparison of the three representations presented so far was made on the benchmark 2×1 cantilever plate using a 20×10 discretization, thus resulting in about 1 hour for 10000 evaluations. Averages over 10 runs are considered, and the computational resources was limited to 7000 Finite Element Analyzes for all runs.

As expected, and in spite of its absence of intrinsic degeneracy, the bit-string representation was consistently outperformed by both other representations - and for a rather coarse discretization. But in every experiment, the holes representation performs significantly better than the Voronoï representation, both in average fitness and in best result. Figure 15 plots an example of such comparative experiments: For that particular problem, the solution is known (the one-element wide beam of weight 0.1), as the constraint on the displacement is very large. Out of 10 runs, it was found exactly 3 times by the holes representation, and never by the Voronoï representation. And similar results were observed, whatever D_{lim} and α.

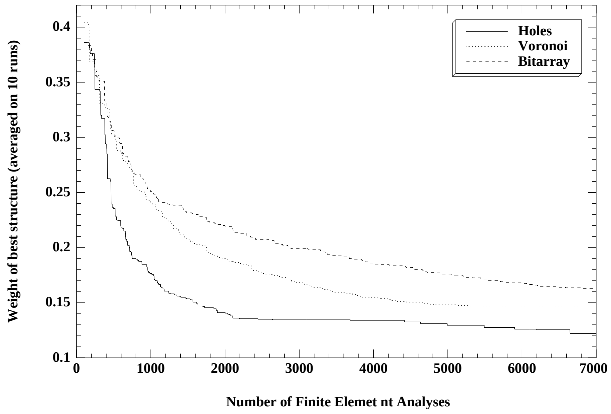


Figure 15 Comparison of the three representations on the benchmark cantilever problem. The weights are given in percentage of the full plate, $D_{Lim} = 371$, so that the perfect horizontal beam of weight 10% is <u>the</u> solution ($D_{Max} = 370.6$), and $\alpha = 1$.

22.6.1 Strong vs weak inheritance

An important feature that all works on problem instances of GAs try to capture is the amount of useful genetic traits that is transmitted from the parents to the offsprings during crossover. And a partial explanation of the reason why the holes representation seems to outperform the Voronoï representation can be derived by a close examination of that feature in both representations: when a hole is inherited by an offspring, the area covered by this hole is void, once and for all. But when a Voronoï site is transmitted to an offspring, the amount of what is really transmitted, in terms of void/material repartition, is totally dependent on the other sites present in the offspring. The holes representation implies *strong inheritance* of genetic traits (at least for the "void" trait), while the Voronoï representation achieves only *weak inheritance*, in which any genetic trait can only be determined by looking at the whole genetic characteristics of the individual.

More careful experiments are needed in order to make these statements precise, such as fitness variance analysis [Rad94] or fitness distance correlation [JF95]. And theoretical explorations certainly can help to understand these results, and more generally to a priori predict the behavior of a given representation on a given problem.

22.7 Conclusion: Evolutionary Computation and Structural Mechanics

Apart from the computational cost, which can hardly be denied, the lack of "sense of Mechanics" is generally the argument of many engineers and researchers in structural Mechanics against evolutionary methods. But such a position is generally the results of a "black box" use of GAs, resulting from the "find a bit-string representation for your problem and you're done" paradigm. We hope to have given arguments in that paper for a different approach to evolutionary optimization:

- at the level of the representation, first. It has been shown how a change of representation can increase the power of evolutionary optimization, while decreasing the complexity of the whole process. Both the Voronoï and the Holes representation outperform the bit-array representation, present a lower complexity for the representation of the same problem. Moreover, they are discretization-independent representations, allowing some iterated process: first find (rather quickly) a coarse solution, and then end with a few generations using a finer discretization, to refine the coarse solution. And finally, the same hierarchy should look even clearer in three dimensions: the bit-array representation clearly reached its limits with 2-dimensional problems, while Voronoï and holes representations hopefully can be used with success on 3-dimensional problems. But this remains to be experimentally demonstrated.
- At the level of the genetic operators, it should be clear from section 22.2.1 that these should be chosen according the problem at hand, and that the success can depend on that choice.

But even more specific operators can be designed, depending on the problem, not only on the representation, as in section 22.2.1, and relying on that "sense of Mechanics": For the Optimal Design problem, the whole stress field provides some indications on where the structure at hand is the strongest and the weakest. Such information could be used by the mutation operator, adding material with higher probability where the stress field is large, and more likely removing material where the stress field is small.

- Finally, at the fitness level, when it comes to understand the results. As was shown for the Optimal Design problem in the nonlinear case in section 22.4.2, only experts in the field can sometimes prevent the evolutionary process from giving non-viable answers.

Many more refinements of the evolutionary process can be imagined. They require knowledge in both Evolutionary Computation (e.g. to respect the fundamental principles of EC, as described in [Gol89b]) and the application domain, (e.g. to both give hints and respect here again the underlying basic principles of the domain). Further work tries to use such synergy to design intelligently biased genetic operators for both the Optimal Design problem and other Structural Mechanics problems (e.g. Inclusion Identification).

22.8 Summary

Three representations for 2D or 3D shapes have been presented, and tested on the problem of Topological Optimal Design of cantilever plates.

The first important issue is demonstrated by extensive experiments using the "natural" bit-array representation. Evolutionary Computation can bring a real breakthrough to the field of Structural Optimization, provided the design of the fitness function takes into account mechanical expertise: a modification of the expression of the fitness has been necessary, first to handle the problem-dependent degeneracy (section 22.3.4), and then after introducing the nonlinear mechanical model, to get mechanically reasonable solutions (section22.4.2).

But the bit-array representation is clearly limited, as its complexity depends on the underlying discretization of the design domain: refining this discretization, or optimizing 3D structures rapidly leads to intractable problems. The Holes and Voronoï representations both seem to overcome partially these difficulties, as their complexity is not fixed, and is evolved by the Evolutionary Algorithm. But they both exhibit a high level of degeneracy. Nevertheless, the first comparative experiments demonstrate they both outperform the non-degenerate bit-array representation, suggesting that complexity plays a greater role than degeneracy, for the problem at hand.

Moreover, the Holes representation consistently performs best on the Optimal Design problem, both in quality of the results and in average computational cost. This may be due to the property of strong inheritance of some genetic traits that the Voronoï representation fails to possess. Nevertheless, theoretical studies and further experiments including different problems are needed in order to draw more affirmative conclusions.

References

[ABFJ95] G. Allaire, E. Bonnetier, G. Francfort, and F. Jouve. Shape optimization by the homogenization method. *submitted to Nümerische Mathematik*, 1995.

[AK93] G. Allaire and R. V. Kohn. Optimal design for minimum weight and compliance in plane stress using extremal microstructures. *European Journal of Mechanics, A/Solids*, 12(6):839–878, 1993.

[Ang93] P. J. Angeline. *Evolutionary Algorithms and Emergent Intelligence.* PhD thesis, Ohio State University, Colombus, 1993.

[BK88] 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.

[BS95] T. Bäck and M. Schütz. Evolution strategies for mixed-integer optimization of optical multilayer systems. In 5^{th} *Annual Conference on Evolutionary Programming*, March 1995.

[Cia88] P. G. Ciarlet. *Mathematical Elasticity, Vol I : Three-Dimensional Elasticity.* North-Holland, Amsterdam, 1988.

[Fog95] D. B. Fogel. *Evolutionary Computation. Toward a New Philosophy of Machine Intelligence.* IEEE Press, 1995.

[FOW66] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution.* New York: John Wiley, 1966.

[FS95] A. Fadda and M. Schoenauer. Evolutionary chromatographic law identification by recurrent neural nets. In J. R. McDonnell, R. G. Reynolds, and D. B. Fogel, editors, 5^{th} *Annual Conference on Evolutionary Programming*, pages 219–235. MIT Press, March 1995.

[Gol89a] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison Wesley, 1989.

[Gol89b] D. E. Goldberg. Zen and the art of genetic algorithms. In *Proceedings of the* 3^{rd} *International Conference on Genetic Algorithms*, pages 80–85, 1989.

[Hol75] J. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, 1975.

[Jen92] E. Jensen. *Topological Structural Design using Genetic Algorithms.* PhD thesis, Purdue University, November 1992.

[JF95] T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In L. Eschelman, editor, *Proceedings of* 6^{th} *International Conference on Genetic Algorithms*, pages 184–192. Morgan Kaufmann, 1995.

[JM91] C. Z. Janikow and Z. Michalewicz. An experimental comparison of binary and floating point representations in genetic algorithms. In R. K. Belew and L. B. Booker, editors, *Proceedings of 4th International Conference on Genetic Algorithms*, pages 31–36. Morgan Kaufmann, July 1991.

[Jou93] F. Jouve. *Modélisation mathématique de l'œil en élasticité non-linéaire*, volume RMA 26. Masson Paris, 1993.

[JS95] F. Jouve and M. Schoenauer. Evolutionary lamé coefficient identification. *In preparation*, 1995.

[Kan95] C. Kane. *Algorithmes génétiques et Optimisation topologique*. PhD thesis, Université de Paris VI, 1995. Soutenance prévue en cours d'année.

[KJS95] C. Kane, F. Jouve, and M. Schoenauer. Structural topology optimization in linear and nonlinear elasticity using genetic algorithms. In *Proceedings of the 21st Design Automation Conference*. ASME, Sept. 1995. To appear.

[KM95] A. B. Kahng and B. R. Moon. Toward more powerful recombinations. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference on Genetic Algorithms*, pages 96–103. Morgan Kaufmann, 1995.

[Mic95] Z. Michalewicz. Genetic algorithms, numerical optimization and constraints on genetic algorithms. In L. J. Eshelman, editor, *Proceedings of the* 6^{th} *International Conference*, pages 151–158. Morgan Kaufmann, 1995.

[Rad91] N. J. Radcliffe. Equivalence class analysis of genetic algorithms. *Complex Systems*, 5:183–20, 1991.

[Rad94] N. J. Radcliffe. Fitness variance of formae and performance prediction. In D. Whitley and M. Vose, editors, *Foundations of Genetic Algorithms 3*. Morgan Kaufmann, 1994.

[Rec73] I. Rechenberg. *Evolutionstrategie: Optimierung Technisher Systeme nach Prinzipien des Biologischen Evolution*. Stuttgart: Fromman-Holzboog Verlag, 1973.

[Sch81] H.-P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley & Sons, New-York, 1981.

[SX93] M. Schoenauer and S. Xanthakis. Constrained ga optimization. In Forrest S., editor, *Proceedings of* 4^{th} *International Conference on Genetic Algorithms*, pages 573–580. Morgan Kaufmann, 1993.

[Sys89] 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 Kauffman, 1989.

[ZMR94] J. M. Zurada, R. J. Marks, and C. J. Robinson, editors. *Computational Intelligence Imitating Life*. IEEE Press, June 1994.