Adaptive techniques for Evolutionary Topological Optimum Design

Hatem Hamda and Marc Schoenauer
CMAP – UMR CNRS 7641 – École Polytechnique – France
{Hatem.Hamda,Marc.Schoenauer}@polytechnique.fr

To appear in ACDM'2000, Plymouth, April 2000

Abstract

This paper introduces some advances in Evolutionary Topological Optimum Design, thanks to extensive use of adaptive techniques. On the genotypic side, a variable length representation is used: the complexity of the representation of each individual is evolved by the algorithm rather than being prescribed by some fixed mesh of the design domain, resulting in self-adaptive complexity. On the phenotypic side, an original adaptive mechanism is proposed that maintains both feasible and infeasible individuals, thus exploring both sides of the boundary of the feasible region, where the optimum structure is known to lie. Not only does this improves the results of past work in on Evolutionary Topological Optimum Design on standard benchmark bidimensional cantilever problems, but it also allows to address three-dimensional problems who had up to now stayed beyond reach for evolutionary algorithms.

1 Introduction

Early works on adaptivity in the framework of Evolutionary Algorithms (EAs) mainly involved the on-line tuning of operator parameters, like the population-level tuning of operator probabilities for Genetic Algorithms [1] and mutation variance in the 1/5th rule for Evolution Strategies [2], or the individual-level self-adaptive mutation step-size in Evolution Strategies [3] or Genetic Algorithms [4]. Some useful definitions and a survey can be found in [5].

A few work addressed the issue of adaptive fitness, either by considering a prescribed series of fitness functions to gradually drive the population toward the optima of the target objective [6] or by adapting some penalty coefficient for CSP [7] or integer programming [8].

Very few work deal with adaptivity at the representation level, and they are concerned with binary representations [9, 10]. However, in the context of Design, the importance of adaptivity in representation has been recently highlighted: In the previous edition of this event, J. Gero's contribution [11] emphasized what could be called ’off-line adaptivity’: a new representation is derived from a careful analysis of the results of a first EA that uses a rather raw representation. Using that new representation, much better results are obtained. And in this volume, P. Bentley’s paper [12] praises what he calls component-based representation, by opposition to more standard parameter-based representations.

This paper focuses on adaptivity for EAs in the context of Topological Optimum Design (TOD), regarding both the representation and the fitness function.

By opposition to the binary representation used in most previous works, the Voronoï representation for the TOD problem is adaptive in the sense that it is a variable length representation: the number of ’genes’ is evolved by the algorithm. Hence the complexity of the representation is self-adaptive (i.e. is adjusted by the evolution itself) at the individual level.

The problem at hand is a constrained problem: An original population-level adaptive penalty method is proposed, ensuring that some feasible and some infeasible individuals remain in the population. This leads to an efficient exploration of the neighborhood of the boundary of the feasible domain, where the solution is known to lie.

The paper is organized as follows. Section 2 presents the mechanical background and briefly reviews some previous works, discussing their limitations. Section 3 introduces the variable length Voronoi representation, together with its variation operators. Its advantages over the standard binary representation are discussed and enforced by mesh-dependency experimental results. Section 4 describes the original adaptive penalty method used to explore the neighborhood of the boundary of the feasible region. Comparative results involving different penalty methods demonstrate that, for the TOD problem at hand, the adaptive strategy outperforms all other methods. In section 5, experimental results on difficult cantilever benchmark problems are presented: the proposed algorithm finds good quality solutions for the 2D 10×1 cantilever, and is able to propose alternative original solutions to a 3D problem. Section 6 closes the paper with some discussion and proposal for further research directions.

2 Background

2.1 The mechanical problem

The general framework of this paper is the problem of finding the optimal shape of a structure (i.e. a repartition of material in a given *design domain*) such that the mechanical behavior of that structure meets some requirements (e.g. a bound on the maximal displacement under a prescribed loading). The optimality criterion is here the weight of the structure, but it could involve other technological costs.

The mechanical model used in this paper will be the standard two-dimensional (except in section 5.2) plane stress linear model, and only linear elastic materials will be considered (see e.g. [13]).

Throughout this paper, the most popular benchmark problem of Optimum Design will be used, that is the optimization of a cantilever plate: the design domain is rectangular, the plate is fixed on the left vertical part of its boundary (displacement is forced to 0), and the loading is made of a single force applied on the middle of its right vertical boundary. Figure 2.1 shows the design domain for the 2×1 cantilever plate problem.

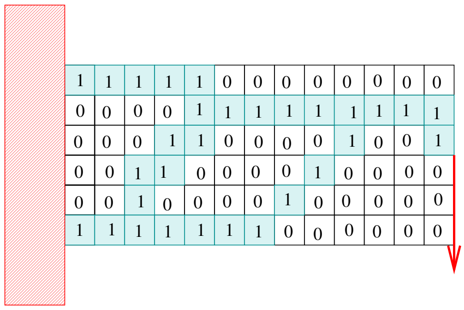


Figure 2.1: *The* 2×1 *cantilever plate test problem, and a bitarray representation of a structure derived from a regular mesh (here a* 13×6 *mesh).*

2.2 Previous works

This paper concentrates on TOD problems, in which no assumption is made on the topology of the structure, in contrast with the domain of shape optimization [14]. The most up-to-date approach to TOD is that of homogenization, introduced in [15], which deals with a continuous density of material in $[0,1]$. This relaxed problem is known to have a unique solution in the case of linear elasticity and for one single case [16] – and the corresponding numerical method does converge to that non-physical solution [17], which is further forced to a feasible solution (with boolean density). This approach is insofar limited to the linear-elasticity/single-loading case, and cannot address loadings that apply on the (unknown) actual boundary of the shape (e.g. uniform pressure).

A possible approach to overcome these difficulties of TOD is to use stochastic optimization methods. Simulated Annealing has been used to find the optimal shape of the cross-section of a beam in [18]; and Evolutionary Algorithms have been used to solve the cantilever problem presented in Section 2.1 in [19, 20, 21]. The above limitations of the homogenization method have been successfully overcome by these works – in [21, 22] for instance, results of TOD in nonlinear elasticity, as well as the optimization of a bicycle (a 3-loading case) and that of an underwater dome (where the loading is applied on the unknown boundary) are presented: all are out of reach for the homogenization method.

2.3 Representations of structures for TOD

The most crucial step when constructing an EA is the choice of representation. All the works cited in previous section that address TOD problems with EAs use the same 'natural' binary representation, termed *bitarray* in [21]: it relies on a mesh of the design domain – the same mesh that is used to compute the mechanical behavior of the structure in order to give it a fitness (see section 2.4). Each element of the mesh is given value 1 if it contains material, 0 otherwise.

In spite of its successes in solving TOD problems [21], bitarray representation suffers from a strong limitation due to the dependency of its complexity on that of the underlying mesh. Indeed, the size of the individual (the number of bits used to encode a structure) **is** the size of the mesh. Unfortunately, according to both the theoretical results in [23] and the empirical considerations in [24] the critical population size required for convergence should be increased at least linearly with the size of the individuals. Hence it is clear that the bitarray approach will not scale up when using very fine meshes. This greatly limits the practical application of this approach to coarse (hence imprecise) 2D meshes, whereas Mechanical Engineers are interested in fine 3D meshes!

These considerations appeal for some more compact representations whose complexity does not depend on a fixed discretization. The ultimate step in the direction of complexity-free representation is to let the complexity itself evolve and be adjusted by the EA.

Two such alternative representations have been proposed in [25], and used to solve non destructive control problem in [26]. The rest of the paper will use one of these, namely the *Voronoï* representation (see section 3.1).

2.4 Fitness computation

The problem tackled in this paper is to find a structure of minimal weight such that its maximal displacement stays within a prescribed limit D_{lim} when some given pointwise force is applied on the loading point (see Figure 2.1). The computation of the maximal displacement is made using a Finite Element Analysis solver (kindly provided by F. Jouve [27]).

From mechanical considerations, all structures that do not connect the loading point and the fixed boundary are given an arbitrary high fitness value. Moreover, the material in the design domain that is not connected to the loading point – and has thus no effect on the mechanical behavior of the structure – is discarded (see [21, 22] for a detailed discussion on both these issues). In summary, for connected structures, the problem is to minimize the (connected) weight subject to the constraint $D_{Max} \leq D_{lim}$, where D_{Max} its maximal displacement computed by the FEM under the prescribed loading. The constraint handling method will be detailed in section 4.

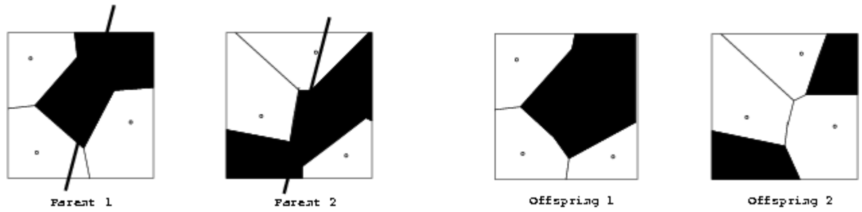


Figure 3.1: *The Voronoï representation, and its crossover operator: A random line is drawn across both diagrams, and the sites on one side are exchanged*

3 Adaptive representation

3.1 Voronoï representation

Voronoï diagrams: Consider a finite number of points $V_0, \ldots, V_N$ (the *Voronoï sites*) of a given subset of $\mathbf{R}^n$ (the design domain). To each site V_i is associated the set of all points of the design domain for which the closest Voronoï site is V_i, termed *Voronoï cell*. The *Voronoï diagram* is the partition of the design domain defined by the Voronoï cells. Each cell is a polyhedral subset of the design domain, and any partition of a domain of $\mathbf{R}^n$ into polyhedral subsets is the Voronoï diagram of at least one set of Voronoï sites (see [28] for a detailed introduction to Voronoï diagrams, and a general presentation of algorithmic geometry).

The genotype: Consider now a (variable length) list of Voronoï sites, each site being labeled 0 or 1. The corresponding Voronoï diagram represents a partition of the design domain into two subsets, if each Voronoï cell is labeled as its associated site. Example of Voronoï representations can be seen in Figure 3.1.

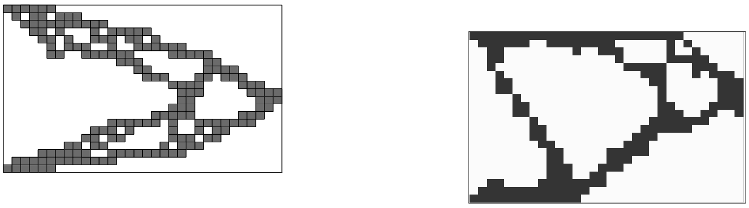
Decoding: Practically, the fitness of all structures will be evaluated using a fixed mesh. A partition described by Voronoï sites is easily mapped on any mesh: the subset (void or material) an element belongs to is determined from the label of the Voronoï cell in which the gravity center of that element lies.

Initialization: the initialization procedure for the Voronoï representation is a uniform choice of the number of Voronoï sites between 1 and a user-supplied maximum number, a uniform choice of the Voronoï sites in the structure, and a uniform choice of the boolean void/material label.

Variation operators: The variation operators for the Voronoï representation are problem-driven:

- The **crossover operator** exchanges Voronoï sites on a geometrical basis. In this respect it is similar to the specific bitarray crossover described in [29]. Figure 3.1 is an example of application of this operator.

- The **mutation operator** is chosen by a roulette wheel selection based on user-defined weights among the following operators:
  – the *displacement mutation* performs a self-adaptive Gaussian mutation on the coordinates of the sites, as defined in [3]: one standard deviation is attached to



(a) - *Bitarray result (weight = 0.267)* (b) - *Voronoï result (weight = 0.272)*

Figure 3.2: *Result for the* 32×22 *mesh after approx. 80000 FEM analyses.*

each coordinate, and undergoes log-normal mutation before being used as step-size for the mutation of the coordinate.
– the *label mutation* randomly flips the boolean attribute of one site.
– the *add* and *delete mutations* are specific variable-length operators that respectively randomly add or remove one Voronoï site on the list.

3.2 Evolutionary experimental conditions

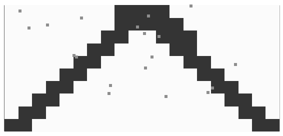
Unless otherwise stated, the experiments presented further on have been performed using the following settings: Standard GA-like evolution (linear rank-based selection and generational replacement of all parents by all offspring) with populations size 80; At most 40 Voronoï sites per individual; Crossover rate is 0.6 and mutation rate per individual is 0.3; Weights among the different mutations are 1/2 for the displacement mutation and 1/6 for the 3 other mutations; All runs are allowed at most 2000 generations, and the algorithm stops after 300 generations without improvement; all plots are the result of 21 independent runs; All CPU times are given related to a Pentium II processor running at 300MHz under Linux.

3.3 Bitarrays vs Voronoï

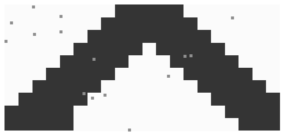
The first experiments performed with Voronoï representation aimed at comparing it with the bitarray representation, in term of both quality of the solutions and computational cost.

Figure 3.2 shows a typical example of a result obtained using the bitarray representation (taken from [21]) and the result on a very similar case obtained using the Voronoï representation (though for technical reasons, the exact mechanical conditions could not be reproduced). The problem is the 2D 2×1 cantilever plate discretized into a 32×22 mesh. Both trials were run for 100 000 evaluations, using a fixed penalty approach for the fitness function (see section 4).

The main difference between both final solutions is that the structure obtained using the Voronoï representation does not exhibit the small holes that appear in the solution of the bitarray representation. However, this difference is probably not due to the well known difficulty that binary GAs usually have to fine tune the very last bits:



(a) - $D_{lim} = 20$, $D_{max} = 19.77$
weight = 0.21, 19 sites



(b) - $D_{lim} = 10$, $D_{max} = 9.959$
weight = 0.44, 13 sites

Figure 3.3: *Result for the Voronoï representation on the* 10×20 *mesh for the* 1×2 *cantilever plate problem (the structures are fixed at the bottom and the force applies at center top) for two different values of the constraint* D_{lim}. *CPU cost is around 2.6s per generation.*

remember that the solution to the TOD problem lies in the relaxed space of structures made of infinitely many holes of infinitely small size (see section 2.2). Hence it seems that the bitarray approach tries to go toward that solution, limited only by the mesh it relies on. On the other hand, the Voronoï representation can only generate more regular shapes, finally giving a much more satisfactory solution, from a technological perspective.

Things are quite different for the case of the 1×2 cantilever problem, where the solution is known to be the perfect "V" shape: Figure 3.3 shows that the Voronoï representation is able to find such shape whereas the bitarray representation could hardly fine-tune the boundary of the structures (see [22]).

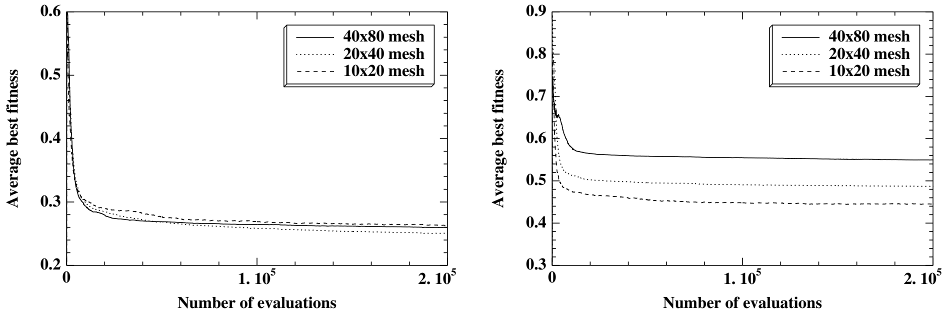
3.4 Mesh-dependency results

In order to acknowledge the non-dependency of the results of the Voronoï representation with respect to the complexity of the mesh, different regular meshes for the 2×1 cantilever plate were used under the same mechanical and evolutionary conditions.

Monitoring the evolution of the result in terms of number of fitness evaluations should show whether the refinement of the mesh does modify the convergence speed, i.e. the number of FEM analyses needed to reach a given solution (note that the computational time needed for one fitness computation increases with the size of the mesh, and that there is no way to avoid it).

Figure 3.4 shows the evolution of the fitness of the best individual (averaged over 21 runs) for the 10×20,20×40 and 40×80 meshes for two different values of the constraint on the maximal displacement ($D_{lim} = 10$ and $D_{lim} = 20$). Whereas Figure 3.4-a shows the expected perfect independence w.r.t. mesh size, the results of Figure 3.4-b first seemed to contradict our hypothesis. However, a closer look at the solutions gave the explanation:

The best solution obtained in a run on the 10×20 (weight 0.44, maximal displacement 0.997) was projected on the 20×40 mesh, it ended up with a weight of 0.43125 and a maximal displacement of 11.265! So the projection error due to the coarseness of the mesh was responsible for the difference in the plots – and as the relative difference is greater for light structures, this explains that it shows mainly on the first plot (see Figure 3.2). But note that even on Figure 3.4-b, the dynamic behavior of the EA is roughly the same for the three meshes – up to the difference in final



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

Figure 3.4: *Weight of optimal structure w.r.t. number of evaluations for the* 10×20, 20×40 *and* 40×80 *meshes of the* 1×2 *cantilever – see solutions Figure 3.3. The CPU costs are respectively about 2.6s, 7.7s and 80s per generation.*

solution. Moreover, the number of Voronoï sites in the best solutions for the three series of runs is roughly the same: respectively 18.1, 19.9 and 21.6 for the 10×20, 20×40 and 40×80 meshes.

4 Adaptive penalty

The problem at hand is a constrained problem (see section 2.4). Constrained optimization has been recently a very active field in Evolutionary Computation, and many specific methods have been designed (see e.g. [30] for a survey).

Moreover, it is clear from mechanical considerations that the solution lies on the boundary of the feasible domain – at least for the continuous problem. Furthermore, specific methods exist to explore the boundary of the feasible domain when the constraint is know to be active at the optimum [31]. Unfortunately, for the TOD problem, first, the solution of the discretized problem does not lie exactly on that boundary, and second, that boundary is out of reach for direct sampling. On the other hand, the penalty method is straightforward to implement in any situation. Hence, as in all previous work [21], the constraint on the maximal displacement of the structure was handled by penalization.

4.1 Penalty methods

Introducing the positive penalty parameter α, the fitness function to minimize is (x^+ is the positive part of x)

$$Weight + \alpha(D_{max} - D_{lim})^+ \tag{1}$$

However, adjusting α is not an easy task (see again [31]). Static penalty, where α is kept constant, can give very good results, but requires a very careful tuning. Dynamic penalty, where α is modified according to a user-defined schedule, as proposed in [32] or in the framework of TOD in [21], requires a lucky guess for the schedule.

Adaptive penalty parameters have been successfully used in the context of (discrete) Constraint Satisfaction Problems [7], where the objective is to find at least one feasible individual. In the context of parameter optimization, an adaptive scheme has been presented in [33]: the penalty parameter is updated according to the feasibility of the best individual in the population only. The new adaptive penalty method proposed here updates the penalty parameter based upon global statistics of feasibility in the population. Its main goal is to explore the neighborhood of the boundary of the feasible region by trying to keep in the population individuals that are on both sides of that boundary (the same idea lead to the Segregated GA [34], that used two different fixed penalty parameters to achieve the same goal).

4.2 Population-based adaptive penalty

The objective is to maintain in the population a minimum proportion of feasible individuals as well as a minimum proportion of infeasible individuals. Denote by $\Theta^k_{feasible}$ the proportion of feasible individuals at generation k, and by Θ_{inf} and Θ_{sup} two user-defined parameters. As small penalty parameters favor the infeasible individuals (and vice-versa), the following update rule for α is proposed to try to keep $\Theta^k_{feasible}$ in $[\Theta_{inf}, \Theta_{sup}]$:

$$\alpha_{k+1} = \begin{cases} \beta \cdot \alpha_k & \text{if } \Theta^k_{feasible} < \Theta_{inf} \\ (1/\beta) \cdot \alpha_k & \text{if } \Theta^k_{feasible} > \Theta_{sup} \\ \alpha_k & \text{otherwise} \end{cases} \tag{2}$$

with $\beta > 1$. User-defined parameters of this method are Θ_{inf}, Θ_{sup}, β and the initial value α_0. Note the limit case $\Theta_{inf} = \Theta_{sup}$ was also tried, but was discarded after the first experimental trials.

The robust values $\beta = 1.1, \Theta_{inf} = 0.4$, and $\Theta_{sup} = 0.8$ were used in all experiments presented here.

Note that the variations of α are non monotonous, and hence there is no a priori guarantee that the best individual in the population is feasible. It can even happen that the population contains no feasible individual – though in that case the steady increase of α will rapidly disadvantage infeasible individuals.

4.3 Comparative results

This section summarizes and discusses the comparative results obtained for different approaches of penalty function.

Initially, several numerical tests were performed on the 10×20 regular mesh using static penalty, and confirmed that for small values of α (e.g. 0.01) the optimal structures are very light, but violate the constraint, while for large values of α (e.g. 10000) all structures visited during evolution are feasible, but the resulting structures are heavy. A good intermediate value (30) was chosen, giving reasonable results in most cases. It will also be used as the starting value for dynamic and adaptive methods.

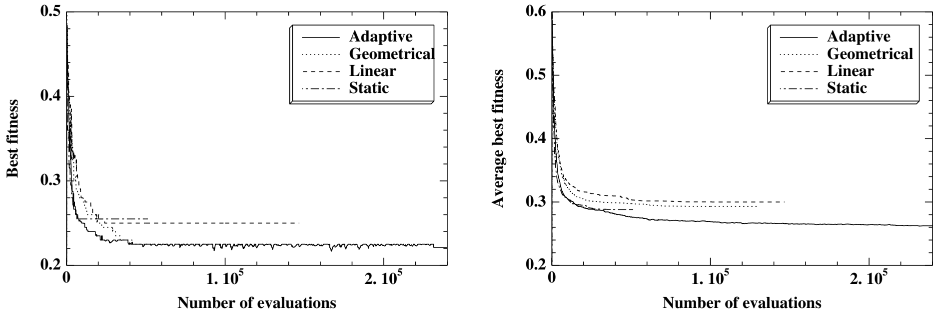


Figure 4.1: *Best and averaged fitnesses (out of 21 runs) for different penalty approaches on the* 10×20 *cantilever problem with* $D_{lim} = 20$.

Two dynamic penalty methods were also included in the comparison, namely a linear increase of α from α_0 to $10 \times \alpha_0$ and the geometrical increase defined in [21] where α is multiplied by a factor $\beta = 0.01$ every 10 generations.

Figure 4.1 shows plots of the best and average fitnesses for the four methods. All final best individuals of all runs were feasible.

First, the plot for the adaptive method clearly shows that the best fitness in the population sometimes correspond to an infeasible individual, which explains the rough aspect of the plot for the best fitness.

With respect to comparison, Figure 4.1 is a rather typical situation: the adaptive strategy clearly outperforms all other methods on average, and equals or outperforms the best result of any of the other method. On the other hand, the geometrical dynamic approach can give results that are as good as the best ones from the adaptive approach, but shows a much larger variance over the 21 runs. Hence it performs poorly on average. The static penalty, when the penalty parameter is carefully tuned, can give rapidly quite good results, but proves unable to improve any more after a rather small number of generations.

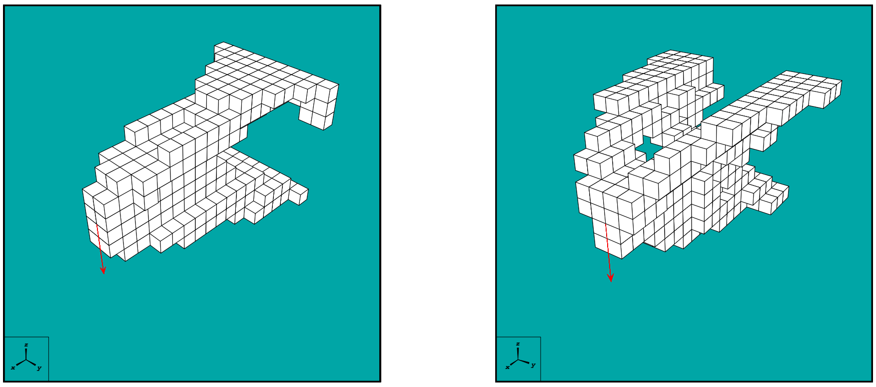
5 Innovative results

Needless to say, all results of [21, 22] demonstrating the flexibility of Evolutionary TOD compared to the homogenization method can be reproduced or even improved by the approach presented here. But the rest of the paper concentrates on purely original results as far as Evolutionary TOD is concerned.

5.1 The 10×1 cantilever

The problem of the 10×1 cantilever (discretized using a 100×10 regular mesh) raises an additional difficulty: most of initial random structures do not connect the fixed boundary and the point where the loading is applied. Hence an alternate initialization procedure was used, where the average weight of random structures can be tuned (see [35] for details). Furthermore, the maximal number of sites for each individuals was

Figure 5.1: *Optimal structure on the* 100×10 *mesh for* 10×1 *cantilever plate.*$D_{lim} = 12$*, number of cells = 105. weight = 0.479. CPU time = 14s/gen.*



(a)- *weight=0.15178, 103 sites* (b) - *weight=0.166, 109 sites*

Figure 5.2: *Two results for the symmetrical three-dimensional problem using a* $16 \times 7 \times 10$ *mesh for half of the structure, with same constraint (CPU time = 6mn/gen).*

increased (to 120), and the best results were obtained with a population size of 120. Figure 5.1 shows the most significant result for $D_{lim} = 12$.

5.2 Three-dimensional problem

This section introduces the first results of 3D TOD obtained using Evolutionary Computation (as far as we are aware of).

The design domain is a quadrangle subset of $\mathbf{R}^3$, and the problem is symmetrical: only half of the domain is discretized, according to a $16 \times 7 \times 10$ mesh. Its left face is fixed, and the loading is applied on the middle of the right face.

Here again the higher complexity of the problem lead to modify the settings: the population size was increased to 120 and the maximum number of Voronoï sites was increased to 120.

Figure 5.2 demonstrates that the algorithm was able to find some good solutions in . . . a few days of CPU time (3D FEM analyses are far more costly than 2D for the same mesh size). Moreover, it also stress the ability of EAs to find multiple quasi-optimal solutions to the same problem, some of them quite original indeed when compared to the results of the homogenization method on the same problem.

6 Discussion and conclusion

The power of adaptivity in Evolutionary Computation is now widely acknowledged, and the results presented in this paper witness for that at two levels: the representation and the fitness.

Many issues remain open for the adaptive penalty method proposed in this paper. The main question is that of the performance of this method when the optimum is not close to the boundary of the feasible region. Automatic computation of the parameters (e.g. the initial guess for the penalty parameter α) from initial sampling of random individuals is also desirable. And of course that method has to be compared to other evolutionary constraint handling methods.

But the main new direction regarding the fitness that could bring huge improvement for Mechanical Engineers might be the use of multi-objective techniques rather that constraint handling methods. There is now a large body of work in the area of multi-objective Evolutionary Computation (see e.g. K. Deb's contribution [36]). And indeed, even the problem of the cantilever plate actually amounts to minimize both the weight and the maximal displacement under several loadings – which is a multi-objective problem. Moreover, modal optimization problems also are multi-objective, as one usually wants the part that is being optimized against bad vibrations to have some minimal stiffness in some prescribed loading situations.

On the representation side, the adaptive Voronoï representation was able to discover simple structures using few Voronoï sites (section 3.3) and more complex structures using many more sites (section 5). Allowing more sites to the simple problems did not significantly modify the results, while limiting the maximum number of sites for the complex problems forbid the emergence of good solutions. As the upper bounds fixed for the number of sites were never hit, one hypothesis, that further experiments will try to check, is that the main effect of the maximum number of sites takes place during initialization.

But though the effects of self-adaptive complexity clearly appear, a close look at the final solutions shows that many Voronoï sites could be removed without any modification of the phenotype (even on the simple structures of Figure 3.3). On the one hand, removing such useless sites could focus the fine tuning on the really useful sites. And on the other hand, if only crucial sites remain in the structures, it might be possible to deduce new and more efficient representations for Evolutionary TOD, along the lines of [11]. Further work will try to remove such useless sites, either by increasing the rate of the delete mutation or by deterministically looking for the useless sites.

Finally, let us quote that adaptivity is also important in the mind of the engineer/programmer: only by getting rid of his past habits will he be able to achieve what will look like artificial creativity, and will only be the reflection of his own.

References

[1] L. Davis. Adapting operator probabilities in genetic algorithms. In J. D. Schaffer, editor, *Proceedings of the* 3^{rd} *International Conference on Genetic Algorithms*, pages 61–69.

Morgan Kaufmann, 1989.

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

[3] H.-P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley & Sons, New-York, 1981. 1995 – 2^{nd} edition.

[4] T. Bäck and M. Schütz. Intelligent mutation rate control in canonical GAs. In Z. W. Ras and M. Michalewicz, editors, *Foundation of Intelligent Systems 9th International Symposium, ISMIS '96*, pages 158–167. Springer Verlag, 1996.

[5] 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.

[6] M. Schoenauer and S. Xanthakis. Constrained GA optimization. In S. Forrest, editor, *Proceedings of the 5^{th} International Conference on Genetic Algorithms*, pages 573–580. Morgan Kaufmann, 1993.

[7] 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.

[8] J. C. Bean and A. B. Hadj-Alouane. A dual genetic algorithm for bounded integer programs. Technical Report TR 92-53, Department of Industrial and Operations Engineering, The University of Michigan, 1992.

[9] C.G. Schaefer. The argot strategy: Adaptive representation genetic optimizer technique. In J. J. Grefenstette, editor, *Proceedings of the 2^{nd} International Conference on Genetic Algorithms*, pages 50–55, 1987.

[10] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivations, analysis and first results. *Complex Systems*, 3:493–530, 1989.

[11] J.S. Gero. Adaptive systems in designing: New analogies from genetics and developmental biology. In I. Parmee, editor, *Adaptive Computing in Design and Manufacture*, pages 3–12. Springer Verlag, 1998.

[12] P. Bentley. Exploring component-based representations. In *This volume*, 2000.

[13] P. G. Ciarlet. *Mathematical Elasticity, Vol I : Three-Dimensional Elasticity*. North-Holland, Amsterdam, 1978.

[14] J. Cea. Problems of shape optimum design. In E. J. Haug and J. Cea, editors, *Optimization of distributed parameter structures - Vol. II*, volume 50, pages 1005–1088. NATO Series, Series E, 1981.

[15] 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.

[16] 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.

[17] G. Allaire, E. Bonnetier, G. Francfort, and F. Jouve. Shape optimiztion by the homogenization method. *Nümerische Mathematik*, 76:27–68, 1997.

[18] G. Anagnostou, E. Ronquist, and A. Patera. A computational procedure for part design. *Computer Methods in Applied Mechanics and Engineering*, 97:33–48, 1992.

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

[20] C. D. Chapman, K. Saitou, and M. J. Jakiela. Genetic algorithms as an approach to configuration and topology design. *Journal of Mechanical Design*, 116:1005–1012, 1994.

[21] C. Kane and M. Schoenauer. Topological optimum design using genetic algorithms. *Control and Cybernetics*, 25(5):1059–1088, 1996.

[22] C. Kane. *Algorithmes génétiques et Optimisation topologique*. PhD thesis, Université de Paris VI, July 1996.

[23] R. Cerf. An asymptotic theory of genetic algorithms. In J.-M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution*, volume 1063 of *LNCS*, pages 37–53. Springer Verlag, 1996.

[24] D. E. Goldberg, K. Deb, and J. H. Clark. Genetic algorithms, noise and the sizing of populations. *Complex Systems*, 6:333–362, 1992.

[25] 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.

[26] M. Schoenauer, L. Kallel, and F. Jouve. Mechanical inclusions identification by evolutionary computation. *European Journal of Finite Elements*, 5(5-6):619–648, 1996.

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

[28] F. P. Preparata and M. I. Shamos. *Computational Geometry: an introduction*. Springer Verlag, 1985.

[29] C. Kane and M. Schoenauer. Genetic operators for two-dimensional shape optimization. In J.-M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Artificial Evolution*, number 1063 in LNCS. Springer Verlag, Septembre 1995.

[30] Z. Michalewicz and M. Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[31] M. Schoenauer and Z. Michalewicz. Boundary operators for constrained parameter optimization problems. In Th. Bäck, editor, *Proceedings of the* 7^{th} *International Conference on Genetic Algorithms*, pages 322–329. Morgan Kaufmann, 1997.

[32] 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.

[33] 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.

[34] R. G. Leriche, C. Knopf-Lenoir, and R. T. Haftka. A segragated 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.

[35] 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. Morgan Kaufmann, 1997.

[36] K. Deb. Multi-objective evolutionary algorithms: Past, present and future. In *this volume*, 2000.