A Segregated Genetic Algorithm for Constrained Structural Optimization

Rodolphe G. Le Riche
Laboratoire LG2MS,
Division MNM,
UTC,
60200 Compiègne, France.
leriche@univ-compiegne.fr

Catherine Knopf-Lenoir
Laboratoire LG2MS,
Division MNM,
UTC,
60200 Compiègne, France.
cklv@univ-compiegne.fr

Raphael T. Haftka
Dept. of Aerospace Engineering,
Mechanics and Engineering Science,
The University of Florida,
Gainesville, FL 32611-6250, USA.
HAFTKA@nervm.nerdc.ufl.edu

Abstract

The problem of minimizing by genetic algorithms the weight of a composite laminate subjected to various failure constraints is considered. Constraints are accounted for through penalty functions. The amount of penalty for each constraint violation is typically controlled by a penalty parameter that has a crucial influence on the performance of the genetic algorithm. An optimal value of each penalty parameter exists. It is larger than the smallest value of the penalty for which the global optimum is feasible. A generational elitist genetic algorithm is found to be less efficient for laminate optimization than a genetic algorithm with a more conservative selection procedure (a "superelitist" algorithm). A segregated genetic algorithm is proposed that uses a double penalty strategy and is superelitist. The segregated genetic algorithm performs as well as the superelitist genetic algorithm for optimal amounts of penalty. In addition, the segregated genetic algorithm is less sensitive to the choice of penalty parameters.

1 Introduction

A general approach to constraints handling in genetic algorithms (GAs) is the use of penalty functions. Conventional wisdom in GAs is that the penalty should be proportional to the distance to the feasible domain, so as to discriminate between solutions having different levels of infeasibility [Richardson et al., 1989]. It has also been argued that the penalty should be kept as low as possible, just above the limit below which infeasible designs become optimal (Minimal Penalty Rule, [Davis, 1987], [Le Riche and Haftka, 1993], [Smith and Tate, 1991]). The flaw of heavily penalizing infeasible designs is that it limits the exploration of the design space to feasible regions, precluding short cuts through the infeasible domain. Thus, the general procedure is to tune the penalty functions on each problem case to assure convergence in the feasible domain and still permit short cuts leading to the optimum solution through the infeasible domain. In [Schoenauer and Xanthakis, 1993], a strategy to tune the penalty parameters based on the Behavioural Memory Paradigm is proposed. Such a strategy is equivalent to having a very large penalty put on one constraint until all members in the population satisfy it, after which the penalty associated to this constraint could be removed. However, the Behavioural Memory Paradigm applies only to design spaces that are dense and, because it violates the Minimal Penalty Rule, it is a costly procedure. Apart from tuning, there is no general solution to the problem of adjusting penalties, neither in classical numerical methods, nor in GAs. Other approaches that do not rely on penalty functions to handle constraints have been proposed. In [Michalewicz and Janikow, 1991], data structuring is proposed as a way to handle linear constraints. "Repair operators" that project infeasible points into the feasible domain have also been used ([Le Riche and Haftka, 1994], [Orvosh and Davis, 1993]). However, the last two approaches to constraints handling are highly problem dependent.

Instead of finding a strategy for tuning the penalty parameters, the purpose of this paper is to desensitize the GA to the choice of the penalty parameters. A "segregated GA" is proposed, where two values of penalty parameters are used for each constraint, instead of one in traditional GAs. The population is split in two coexisting and cooperating groups. The fitness of each group is evaluated using either one of the two penalty parameters. The two groups converge in the design space along two complementary trajectories, which helps locate the optimal region faster and makes the algorithm less sensitive to the choice of penalty parameters.

Results are shown for a structural optimization problem, the minimization of the weight of a composite laminated plate subjected to strength and buckling constraints. The paper starts with the description of the optimization problem and the basic genetic algorithm used to solve it. Then, the segregated genetic algorithm (SGGA) is presented. Results are given that compare four different algorithms, namely, an elitist generational GA (GA), a GA with a very conservative selection procedure (called "superelitist" GA, or EGA), an algorithm similar to the ES($m + m$) of Evolutionary Strategies ([Schwefel, 1981]), and a SGGA. Different values of the penalty parameters are tested. In the process, exceptions to the Minimal Penalty Rule are demonstrated.

2 Problem Description

2.1 Minimal Weight Design of a Composite Laminated Plate

Composite materials typically consist of fibers made of stiff materials, such as graphite, embedded in a soft matrix, such as epoxy resin. Their high strength-to-weight and stiffness-to-weight ratios make them attractive for aerospace applications. The simply supported laminate shown in Figure 1 is loaded in the x and y directions by X_i and Y_i, respectively. The laminate is composed of N plies, each of thickness t. Because of manufacturing considerations,

- the orientation of the fibers in the plies are restricted to the discrete set (0^o, $+45^o$, -45^o, and 90^o),
- the laminate must be symmetric about its mid-plane,
- the laminate must be balanced (same number of $+45^o$ and -45^o layers).

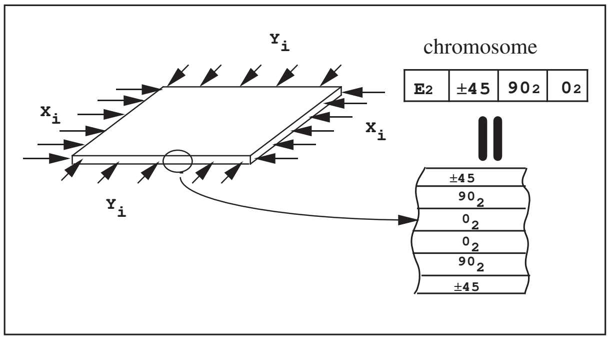

Figure 1: Simply supported laminated plate subjected to normal in-plane loads and coding

The plate is designed not to fail, neither from buckling nor from insufficient strength. To quantify the margin of safety, the applied loads are multiplied by a load factor λ, that is, they become λX_i and λY_i. Analytical expressions are available ([Le Riche and Haftka, 1993]) to predict the buckling load factor λ_{cb} and the strength failure load factor $\lambda_{c\varepsilon}$. If λ_{cb} is larger than 1, the laminate can sustain the actual applied loads X_i and Y_i without buckling, and vice versa. Similarly, if $\lambda_{c\varepsilon}$ is smaller than 1, the plate is assumed to fail because of insufficient strength. The critical load factor λ_{cr} is,

$$\lambda_{cr} \ = \ \min(\lambda_{cb}, \lambda_{c\varepsilon}). \qquad (1)$$

If λ_{cr} is larger than 1, the plate does not buckle or suffer strength failure. To alleviate matrix cracking problems, we also require that there are no more than four contiguous plies with the same fiber orientation. If many load cases (X_i,Y_i) are considered simultaneously, λ_{cr} is the smallest among the λ_{cr} determined for each load case.

2.2 Optimization Formulation

The purpose of the optimization is to find the thinnest symmetric and balanced laminate satisfying the 4-ply contiguity constraint that will not fail. The optimization is performed through the choice of the total number of plies and the fiber angles for each ply, i.e., we optimize the "stacking sequence" of the laminate.

To reduce the size of the optimization problem and to automatically satisfy the balance condition we consider only laminates made up of 2-ply stacks 0_2 (two 0^o plies), 90_2 (two 90^o plies), or ± 45 (a pair of $+45^o$ and -45^o plies). The constraint on symmetry is readily implemented by considering only one half of the laminate (the other half being deduced by symmetry). As a result a laminate with under N plies can be represented by a chromosome of length N/4. To accomodate variable thickness in a fixed string length we add an empty stack and denote it as E_2. Figure 1 shows an example of a laminate cross-section, and the associated coded chromosome assuming that the maximum thickness needed is $N = 16$ plies. To avoid empty layers in the laminate, the characters E_2 that can appear between full layers in the course of the optimization are immediately pushed to the left of the chromosome (outside of the laminate).

The unconstrained objective function that we want to minimize is the total number of plies N in the laminate. The constrained objective function Φ is defined as,

$$\begin{aligned} \text{if } \lambda_{cr} \geq (1-\delta), \quad \Phi &= P_c^{n_c}\{N + \varepsilon[(1-\delta) - \lambda_{cr}]\}, \\ \text{if } \lambda_{cr} < (1-\delta), \quad \Phi &= P_c^{n_c}\{\tfrac{N}{\lambda_{cr}^p}\} + S. \end{aligned} \qquad (2)$$

The constraint on failure of the laminate is enforced by the penalty functions $(1/\lambda_{cr}^p)$ and S. p and S are penalty parameters for failure. $(1/\lambda_{cr}^p)$ is proportional to the distance to feasibility, while S is a fixed amount of penalty for being infeasible. A penalty function of the form $(1/\lambda_{cr}^p)$ rather than a linear additive penalty function was chosen in order to be able to apply rules from the strength of materials to find the order of magnitude of p ([Kogiso et al., 1994]). The contiguity constraint is enforced by multiplying the objective by $P_c^{n_c}$, where P_c is the penalty for the ply contiguity constraint, and n_c is the number of stacks of two plies in excess of the constraint value of two stacks (i.e., 4 plies) in one half of the laminate. For example, the objective function corresponding to the chromosome 0_2 0_2 0_2 90_2 is multiplied by P_c. Because of the discrete nature of the problem, there may be several designs of the same minimum thickness. Of these designs, we define the optimum to be the design with the largest failure load λ_{cr}. Therefore, the objective function is linearly reduced in proportion to the failure margin for designs that satisfy or almost satisfy the failure constraint (term $\varepsilon[(1-\delta)-\lambda_{cr}]$). δ is a small tolerance parameter added for more flexibility in the definition of feasibility, and ε is a small parameter for rewarding constraint margin. Values of ε, P_c and S were derived in [Kogiso et al., 1994] and [Le Riche and Haftka, 1994], for laminates that are 48 plies thick and are used here: $\varepsilon = 6$, $P_c = \sqrt{\frac{10}{9}}$, $S = 1.05$. Additionally, δ is set to 0.005.

The setting of p is the subject of the present study. Considering the plate and loadings described in the Results section, for low values of the penalty parameter p, the objective function is minimized by an infeasible 44-ply design ($\lambda_{cr} \approx 0.8$). For higher values of p the objective function is minimized by a feasible 48-ply design.

2.3 A GA for Laminate Design

The basic GA developed in [Le Riche and Haftka, 1994] is used as the baseline for the present work. It is a generational elitist GA where selection is performed according to the normalized rank of each individual in the population ([Goldberg, 1989]). The coding was presented in the previous section.

Two consecutive calls of selection are made to pair two parent designs. Then an offspring is generated by successive applications of crossover, mutation and one-stack swap operators. Each of these operators has its own probability of being applied. The crossover used here is a one-point thick crossover ([Le Riche and Haftka, 1994]), where the break point is chosen in the full part of the thicker laminate.

The mutation is composed of three independent events, each with its own probability of occuring. Mutation can randomly change the fiber orientation (for example, a 0_2 becomes a ± 45), add, or delete plies anywhere in the laminate. An example of stack addition is when a E_2 becomes a 90_2. Conversely, if a 0_2 becomes a E_2, a stack is deleted.

The last operator is called one-stack swap. This operator selects randomly two stacks in the full part of the laminate and swaps them:

$$\textbf{Before stack swap} \quad \underbrace{90_2}_{1} \quad 90_2 \quad \underbrace{\pm 45}_{2} \quad 0_2 \quad \pm 45$$

$$\textbf{After stack swap} \quad \underbrace{\pm 45}_{2} \quad 90_2 \quad \underbrace{90_2}_{1} \quad 0_2 \quad \pm 45$$

In conformity with results presented in [Le Riche and Haftka, 1994], the population size is 8, the probability of applying crossover is 1, the probability of changing the fiber orientation is 0.01 per building block, the probability of adding and deleting two stacks of two plies is 0.05 per design, and the probability of one-stack swap is 1. The combination of a small population size and a high probability of stack swap were found to be an optimal set of parameters for this class of problems in [Le Riche and Haftka, 1993] and [Le Riche and Haftka, 1994].

2.4 Performance Criterion

The design space for our problem (see Results Section for a complete description) was found to include a large number of near optimum designs. For this reason, designs that are feasible, of optimal weight, and whose λ_{cr} is within a 10th of a percent of λ_{cr} of the global optimum are called **practical optima**. The measure of the performance is the **reliability** of the algorithm. It is the probability the GA has of finding a practical optimum after a given number of analyses (function evaluations). The reliability of the algorithm is evaluated in this paper by running 3000 independent searches. The **price** of the search is the number of analyses necessary to achieve 80% probability of finding a practical optimum.

Table 1: Reliability of Basic GA, average reliability $\bar{r}$ and standard deviation σ

no of analyses	$r = \bar{r} \pm 1.96\sigma$
500	0.15 ± 0.05
1000	0.37 ± 0.01
1500	0.53 ± 0.04
2000	0.63 ± 0.06
3000	0.77 ± 0.06
4000	0.86 ± 0.06
5000	0.90 ± 0.08
6000	0.94 ± 0.04

Table 1 shows the average reliability $\bar{r}$ and the standard deviation σ of a GA where $p = 0.5$. There is 95% probability that the actual reliability r is $r = \bar{r} \pm 1.96\sigma$. It may appear extravagant to perform 3000 genetic optimizations in order to calculate the reliability of the algorithm. However, even with this large number, there is considerable scatter in the calculated reliability, as shown in Table 1. This indicates the danger of assessing the performance of a genetic algorithm based on a small number of runs.

3 A Segregated GA

3.1 Principle

The basic idea of the segregated GA (SGGA) is to use two values of the penalty parameter (p_1 and p_2 here) instead of one. The two values of the penalty parameters are associated with two groups of solutions that have a different level of satisfaction of the constraints. Each of the groups corresponds to the best performing individuals with respect to one penalty parameter. The two groups interbreed, but they are segregated in terms of rank. Two advantages are expected. First, because the penalty parameters are different, the two groups will have distinct trajectories in the design space. Because the two groups interbreed, they can help each other out of local optima. The SGGA is thus expected to be more robust than the GA. Second, in constrained optimization problems, the global optimum is typically located at the boundary between feasible and infeasible domains. If one selects one of the penalty parameters large (say p_1) and the other one small (p_2), one can achieve simultaneous convergence from both the feasible and the infeasible side. The global optimum will then be rapidly encircled by the two groups of solutions, and since the two groups of designs interbreed, the global optimum should be located faster.

For example, in structural optimization, one usually seeks to minimize the weight of a structure. The "p_1 group" typically contains heavy designs that do not fail, while the "p_2 group" contains light designs that fail. The optimum design which is a compromise between weight and constraint satisfaction is located somewhere between the p_1 and p_2 groups.

3.2 Implementation of Segregation

Figure 2 gives the flow chart of the SGGA. The algorithm starts by creating $2m$ designs randomly, where m is the population size. Then, the $2m$ individuals are ranked in two lists, each list corresponding to a value of the penalty parameter. As a protection against premature uniformization of the population, duplicates of designs are pushed to the bottom (low rank) of the list. Then, from these two lists of $2m$ ranked individuals, one single population of m individuals is built which mixes the relative influences of the two lists. One starts by selecting the best individual of the list established using the highest penalty parameter (p_1). Then, one chooses the best individual of the other list

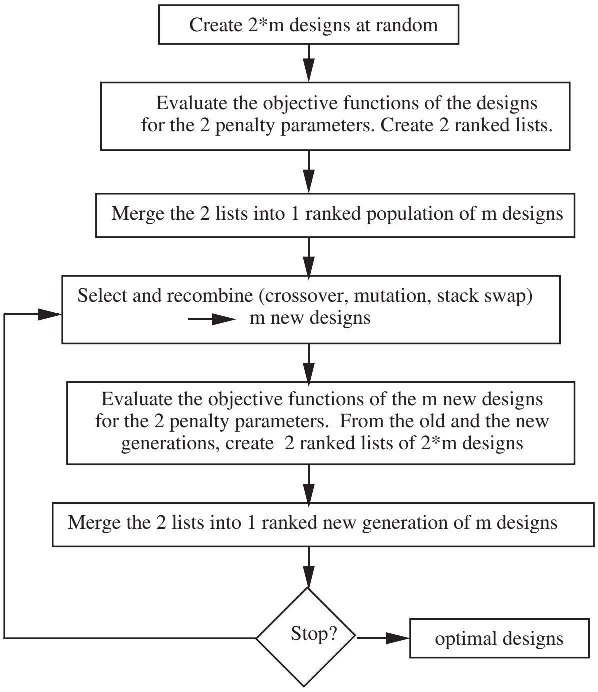

Figure 2: Flow chart of a segregated genetic algorithm

that has not yet been selected. The process is repeated alternatively on each list until m individuals have been selected. When m is even the population is evenly divided between the two lists. Then, reproduction occurs as usual by application of linear ranking selection, crossover, mutation, and stack swap to the combined list, creating m offsprings. They are added to the m parents, and the entire process is repeated.

Besides the use of two penalty parameters, a difference between the SGGA and the GA is population replacement. If the two penalty parameters p_1 and p_2 are the same, the population replacement strategy takes the m children, adds the m parents, and keeps the m best individuals for further processing. This type of population replacement already exists in Evolutionary Strategies ([Schwefel, 1981]) and has been called ES(m + m) selection. It is an important feature of the SGGA because it permits to balance the influence of the two penalty parameters. For example, let us assume that the two lists roughly correspond to feasible and infeasible points (that happens if one penalty parameter is large and the other small), and the rate of production of infeasible points is higher than the rate of production of feasible points. In such a case, if all parent designs are not kept, the population will rapidly be filled with infeasible designs. To avoid this, the SGGA controls the overtaking of the population by one class of individuals by killing no parent design a priori. Because any parent design can be transmitted unchanged to the next generation, the population has a strong tendency to become uniform. Mechanisms that protect diversity in the population, such as giving bad ranks to duplicates, are important for a successful implementation of the SGGA.

We call a segregated GA where p_1 is different from p_2 SGGA. If p_1 is equal to p_2, the SGGA is equivalent to an algorithm where linear ranking selection is superimposed to ES($m + m$) selection. We call this algorithm "superElitist" GA (EGA). If the two penalty parameters are the same and linear ranking is removed, the algorithm is denoted ES($m + m$). Table 2 summarizes the algorithms compared in this study.

Table 2: Different types of algorithms considered

Name	Selection	Penalty
GA	linear ranking + elitism	p
EGA	linear ranking + ES(m+m) selection	p_1=p_2
SGGA	linear ranking + ES(m+m) selection	$p_1 \neq p_2$
ES(m+m)	ES(m+m) selection	$p_1 \neq p_2$ or p_1=p_2

4 Results

We consider a graphite-epoxy plate with longitudinal and lateral dimensions a = 20 in. and b = 5 in., respectively. The material properties are: $E_1 = 18.50 \times 10^6$ psi; $E_2 = 1.89 \times 10^6$ psi; $G_{12} = 0.93 \times 10^6$ psi; $\nu_{12} = 0.3$; $t = 0.005$ in. (t is the basic ply thickness). The maximum thickness for a laminate is assumed to be 64 plies, i.e., the string length is $64/4 = 16$.

One optimization problem is addressed in this paper, where three loadings are considered simultaneously: [X_1 = 12,000. lb/in, Y_1 = 1,500. lb/in], [X_2 = 10,800. lb/in, Y_2 = 2,700. lb/in], [X_3 = 9,000. lb/in, Y_3 = 4,500. lb/in], and the ultimate allowable strains are $\epsilon_1^{ua} = 0.008$, $\epsilon_2^{ua} = 0.029$, $\gamma_{12}^{ua} = 0.015$. This optimization problem has been treated over 100000 times, so the best-known design is taken as the optimum. Another load set has been considered (cf. [Le Riche et al., 1995]). All the conclusions presented in this paper are confirmed by this other optimization problem.

The performances of SGGA, EGA, ES($m + m$) and GA will be compared in terms of price of the search obtained from 3000 independent runs). We look at three values of the penalty parameter p, 0.4, 0.5 and 5.0. These values were found experimentally. $p = 0.5$ is the optimal value of the penalty parameter for the GA that permits the highest reliabilities between 0 and 6000 analyses (thus the lowest price of the search). p = 0.4 is a small value of the penalty parameter because there are many cases where the search is still in the infeasible domain after 6000 analyses. $p = 5.0$ is much larger than 0.5, so we consider it a large value of the penalty parameter. Figures 3, 4 and 5 show the reliability of GA, EGA, and SGGA, respectively, for different values of the penalty parameters p, p_1 and p_2. Table 3 summarizes the results in terms of price of the search. In addition to the figures 3, 4 and 5, Table 3 gives the performance of ES(m+m).

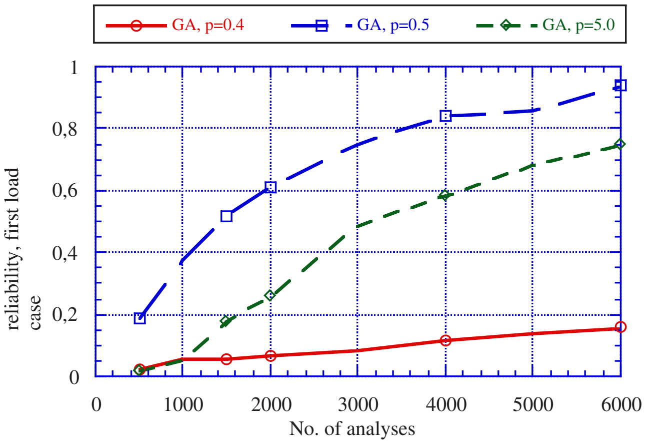

Figure 3: Reliability r of GA

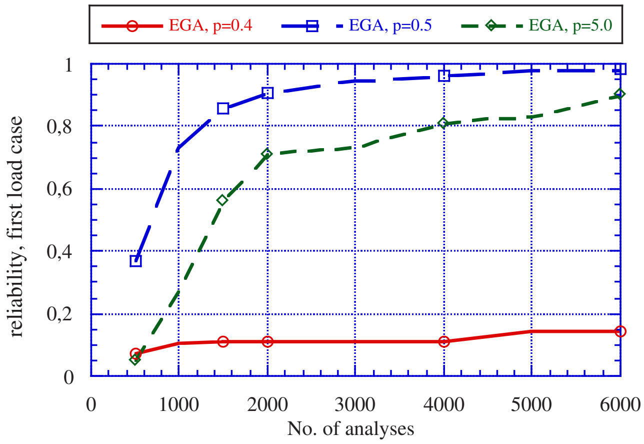

Figure 4: Reliability r of EGA

Before we start comparing different versions of the algorithm, note that Figure 3 contradicts the Minimal Penalty Rule ([Davis, 1987], [Le Riche and Haftka, 1993], [Le Riche and Haftka, 1994], [Smith and Tate, 1991]). The Minimal Penalty Rule says that, in order for the genetic search to be most efficient, the penalty should have the smallest value for which the global optimum is in the feasible domain. For our example, the global optimum is in the feasible domain for p larger or equal to 0.4. However, when p = 0.4, the genetic algorithm has a very low reliability, as shown in Figure 3. Almost every time the algorithm does not converge to the optimum it converges to an infeasible local optimum Even though the Minimum Penalty

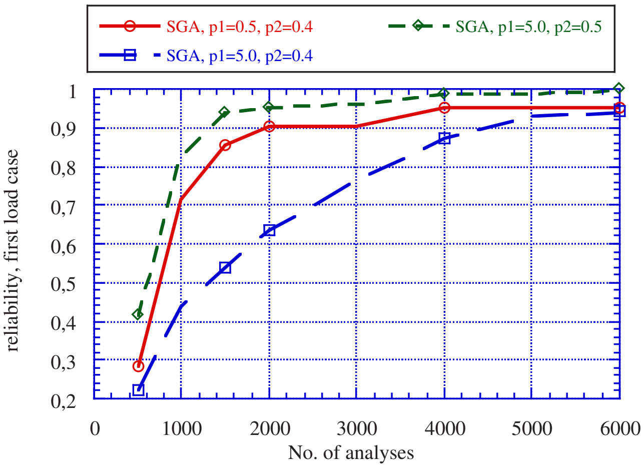

Figure 5: Reliability r of SGGA

Rule would advise choosing $p = 0.4$, our experiments show that a larger value of the penalty parameter is preferable. We think that for a penalty parameter as small as 0.4, the GA spends most of its time in the infeasible domain, climbing the road towards the global optimum, but not getting there on time (before 6000 analyses). It is a well known fact that too large a penalty parameter makes the objective function GA-hard ([Davis, 1987], [Le Riche and Haftka, 1993], [Richardson et al., 1989]). Likewise, our experiments show that too small a penalty parameter (as predicted by the Minimal Penalty Rule) can also make the objective function GA-hard. The "path length" ([Horn et al., 1994]) seems to be the type of difficulty that a wrong choice of penalty parameter introduces.

Table 3: Price of the search of different GAs

GA type	Price of search
GA p=0.4	r = 0.15*
GA p=0.5	3580
GA p=5.0	r = 0.75*
EGA p=0.4	r = 0.15*
EGA p=0.5	1380
EGA p=5.0	3980
SGGA p_1=0.5 p_2=0.4	1270
SGGA p_1=5.0 p_2=0.4	3350
SGGA p_1=0.5 p_2=0.5	990
ES(m+m) p_1=p_2=0.4	r = 0.13*
ES(m+m) p_1=p_2=0.5	2010
ES(m+m) p_1=p_2=5.0	3080
ES(m+m) p_1=0.5 p_2=0.4	2780
ES(m+m) p_1=5.0 p_2=0.4	3390
ES(m+m) p_1=5.0 p_2=0.5	1290

* : reliability at 6000 analyses. The price is not defined because 80% reliability were not achieved within 6000 analyses

We now turn to the relative performances of the GA, the superelitist GA (EGA), the ES($m + m$), and of the segregated GA (SGGA). It is seen in Table 3 that EGA has a lower price than GA. Note however that the EGA gets trapped just like the GA in local infeasible optima for $p = 0.4$. Combinations of the penalty parameters (large, small), (optimal, small) and (large, optimal) are tested using the SGGA. The best tuned SGGA is ($p_1 = 5.0$, $p_2 = 0.5$), and the price of the search (number of analyses required for 80% reliability) is 990 analyses. It performs better than the best EGA and the best GA (both obtained with p optimal equal to 0.5) which have a price of 1380 and 3580, respectively.

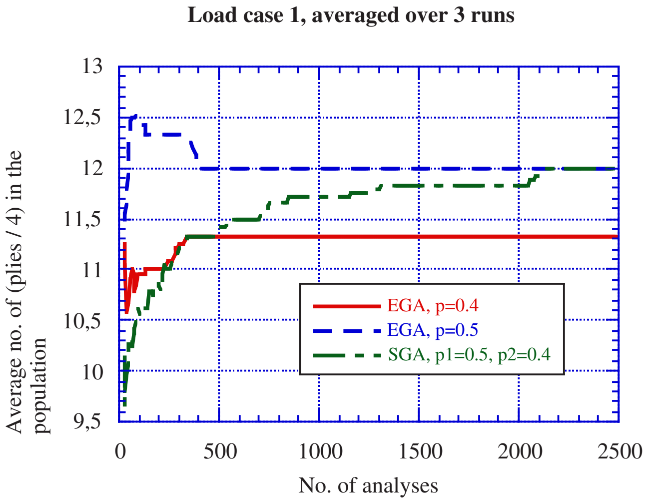

Figure 6: Comparison of convergences in terms of weight in the population

Note also that the three curves of Figure 5 are closer together than those in figures 3 and 4. Especially, the curve $p_1 = 5.0$, $p_2 = 0.4$, which corresponds to the worst possible choice of a pair of penalty parameters, (large, small), is quite close to the curves resulting from better choices of penalty parameters. Thanks to interbreeding between the two segregated groups in the population, the SGGA is less sensitive to the choice of penalty parameters. Another illustration of the working of the SGGA is given in Figure 6 where the average weight of the population is given in terms of the number of analyses. EGA with $p = 0.5$ converges to the optimal weight ($48/4 = 12$) from above. With $p = 0.4$, EGA converges to less than the optimal weight from below (it frequently gets stuck at 44 plies). SGGA with ($p_1 = 0.5$, $p_2 = 0.4$) converges to the optimal weight from below. Not only the SGGA preserves diversity by segregation, but it also preserves the precise information that permits escaping infeasible local optima.

A series of experiments was run where linear ranking selection was suppressed. The only selection left is ES(m+m) population replacement strategy. Because of its lower selection pressure, ES(m+m) has a larger price of search than EGA or SGGA do (cf. Table 3). On the other hand, towards the end of the search, the reliabilities achieved by ES($m + m$) are marginally better than those of SGGA and EGA: ES($m + m$), $p_1 = p_2 = 0.5$, has a reliability of 100% at 5000 analyses, while EGA, $p_1 = p_2 = 0.5$, achieves 98% reliability at 6000 analyses. At 6000 analyses, the reliability of ES($m + m$), $p_1 = 5.0$, $p_2 = 0.4$, is 98% and the reliability of EGA, $p_1 = 5.0$, $p_2 = 0.4$, is 94%. Logically, for $p_1 = p_2 = 5.0$, which is an excessive amount of penalty, reducing the selection pressure helps not getting trapped in the feasible domain far from the optimum, i.e., ES($m + m$) has a price of 3080 analyses while EGA has a price of 3980 analyses.

Globally, in the general case of restricted information about the amount of penalty that is necessary to put on infeasible designs, a strategy based on segregation performed the best: the best tuned EGA or ES($m + m$) does not do better than a reasonably tuned SGGA, and the SGGA is much less sensitive to the choice of the penalty parameters.

5 Conclusions

A segregated genetic algorithm (SGGA) has been proposed for composite laminate design. Two components of the SGGA have been thoroughly tested when minimizing the weight of a composite laminate. First, a selection scheme where all of the parent generation is preserved and competes with the children has proven to be a more efficient strategy than a generational-elitist GA. Second, a fitness with two different penalty parameter permits convergence towards the optimum region of the design space from two different directions. Interbreed between individuals converging from both directions makes the search less sensitive to the setting of the penalty parameters.

Making the GA less sensitive to the choice of the penalty parameter is thought to be of practical importance. Indeed, tests presented in this and other studies have shown that the performance of the GA on constrained optimization problems depends a lot on the setting of the penalty parameters. In general, there is no procedure to select the best amount of penalty. Examples that contradict the most common rule for tuning penalty parameters, the Minimal Penalty Rule, have been given.

Acknowledgements

The first author aknowledges support by the Conseil Régional de Picardie through the Pôle Régional de Modélisation. Also, special thanks to Marc Schoenauer and the people at the Centre de Mathématiques Appliquées de l'École Polytechnique.

References

Davis, L., 1987 , *Genetic Algorithms and Simulated Annealing*, Pitman, London.

Goldberg, D.E., 1989 , *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley Publishing Company, Inc., Reading, MA.

Horn, J., Goldberg, D.E., Kalyanmoy, D., 1994 , "Long Path Problems", *Proceedings of the Internatinal Conference on Parallel Problem Solving from Nature*, Jerusalem, Israel, Davidor, Y., Schwefel, H.-P., and Männer, R., Eds., Springer-Verlag Publ., Berlin, Germany, pp. 149-158 (also accepted for publication in *Parallel Problem Solving from Nature - PPSN III*).

Kogiso, N., Watson, L.T., Gürdal, Z., Haftka, R.T., and Nagendra, S., 1994, "Minimum Thickness Design of Composite Laminates subject to Buckling and Strength Constraints by Genetic Algorithms", *Mechanics of Composite Materials and Structures*, 1(1), pp. 95-117.

Le Riche, R., and Haftka, R.T., 1993 , "Optimization of Laminate Stacking Sequence for Buckling Load Maximization by Genetic Algorithm", *AIAA Journal*, 31(5), pp. 951-970.

Le Riche, R., and Haftka, R.T., 1994 , "Improved Genetic Algorithm for Minimum Thickness Composite Laminate Design", *Composites Engineering*, 3(1), pp121-139.

Le Riche, R.,Knopf-Lenoir C., and Haftka, R.T., 1995, "A Segregated Genetic Algorithm for Constrained Structural Optimization", Technical Report, Université de Technologie de Compiègne, France, available by anonymous ftp on ftp.univ-compiegne.fr as pub/utc/lg2ms/genetic-leriche.ps.gz

Michalewicz, Z., Janikow, C.Z., 1991 , "Handling Constraints in Genetic Algorithms", *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Mateo, CA, Morgan Kaufmann Publishers, pp. 151-157.

Orvosh, D., Davis, L., 1993 , "Shall We Repair? Genetic Algorithms, Combinatorial Optimization, and Feasibility Constraints", *Proceedings of the Fifth International Conference on Genetic Algorithms* , Univ. of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, pp. 650.

Powell, D., and Skolnick, M.M., 1991 , "Using Genetic Algorithms in Engineering Design Optimization with Non-linear Constraints", *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Mateo, CA, Morgan Kaufmann Publishers, pp. 424-431.

Richardson, J.T., Palmer, M.R., Liepins, G., and Hilliard, M., 1989, "Some Guidelines For Genetic Algorithms with Penalty Functions", *Proceedings of the Third International Conference on Genetic Algorithms* , George Mason Univ., Morgan Kaufmann Publishers, pp. 191-197.

Schoenauer, M., and Xanthakis, S., 1993 , "Constrained GA Optimization", *Proceedings of the*

Fifth International Conference on Genetic Algorithms , Univ. of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, pp. 573-580.

Schwefel, H.-P., 1981 , *Numerical Optimization of Computer Models*, Chichester, Wiley.

Smith, A.E., and Tate, D.M., 1991 , "Genetic Optimization Using A Penalty Function", *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Mateo, CA, Morgan Kaufmann Publishers, pp. 499-505.