A Genetic Algorithm for Laminate Layups

N. R. Ball, P. M. Sargent, D. O. Ige

Engineering Design Centre, Department of Engineering, University of Cambridge, Trumpington Street, Cambridge, CB2 1PZ U.K

ABSTRACT

The design of laminate layups using a genetic algorithm function optimizer has been investigated using a generate-and-test evaluation function and the layup-synthesis rule-base from a knowledge based design system. A simple coding of plies to genes is shown to be applicable in each case when coupled with a penalty function that constrains genetic search to permissible layup candidates. Experiments with the generate-and-test evaluation demonstrate the robustness and accuracy of the system in the design of small scale asymmetric laminates. Comparative tests with a knowledge based laminate design system are also presented for four different load types which confirm the ability of the genetic system to search laminate configuration spaces with a rule-based evaluation function.

KEYWORDS

composite, crossover, evaluation, gene, genetic algorithm, knowledge-based system, laminate, layup, mutation, optimization, penalty function, ply, search.
INTRODUCTION

The design of a composite laminate can be considered in two interdependent procedures. Firstly the decision of how many of which orientations of plies to use and secondly the layup or sequence in which these plies are arranged. This paper reports recent research into the latter layup problem. The two procedures are separable in practice because the numbers and identities of plies are determined by the in-plane shear and tensile loads on the component being designed whereas the layup sequence is determined by out of plane bending loads. Typically one designs the in-plane situation first. The procedures described in this paper are applicable to all laminate materials whether based on polymer, metal or ceramic matrices and whether continuous or short fibre reinforcement is used. However the main objective of the work is to enable mechanical engineers to get more out of established and well-characterized materials such as continuous carbon fibre reinforced epoxy resin polymer (CFRP).

A description of the laminate layup problem will now be briefly described in computational terms. For a laminate made up of n plies, where the number m of distinct plies of different orientations is between 1 and n, an algorithm is required that will generate a sequence of complete layups to be evaluated for fitness of purpose, where fitness is calculated as a real number between 0 and a fixed maximum limit. The configuration space of possible solutions for this problem is large - for example an asymmetric 10mm laminate consisting of equal numbers of 4 different orientations of plies 0.125mm thick can be laid up in over $2.10^{45}$ configurations. The corresponding figure for the symmetric case (where the two halves are mirror images) is still a daunting $4.10^{21}$ (Ige90a, Sar90).

The evaluation function for generate and test methods returns a single numeric fitness value based on the current solution's applied stress state and failure tendency (as measured against a set of failure mechanisms). The function assesses the relative suitability of a complete layup. Any such function is limited in that some key parameters of the problem domain may not be captured by it and/or the weighting associated with each parameter may be inappropriate. These two factors inject noise into the function and imply that any search method cannot be relied upon to generate the best design but should only be used to generate a set of potentially good designs that can then be ranked by other methods. The inadequacy of evaluation functions is due simply to our incomplete knowledge of the relative importance of failure modes and loading conditions for laminate composites.

Previous work has demonstrated that random and greedy algorithms are fast enough and complex enough to permit some initial exploration of the configuration of laminate layups and provide useful guidance (in terms of evaluation functions) for the development of a system based on genetic algorithms 1. Two main permutation operators were utilized in those algorithms -

1. a swap operator that exchanged two plies randomly selected from the laminate.
2. a cycle operator that cyclically shifted three adjacent plies.

Experiments on the greedy algorithm were also conducted using a combination of (1) and (2).

Application of these different permutation operators as search steps in the configuration space revealed that there were optimum proportions of different step types which most rapidly converged to solutions. However these proportions were dependent on the specific layup problem and evaluation function. Thus a system able to configure its behaviour to take advantage of the particular properties of individual problems (i.e. vary its step mechanism in some manner) would offer advantages over random/greedy algorithms. It is this requirement that prompted the study of genetic algorithms to the laminate layup problem.

The remainder of this paper is structured as follows. Section 1 presents the layup problem. Section 2 gives an overview of the Genetic Algorithm and describes how they can be applied to the problem domain. Section 3 presents results using different penalty functions on a 10-ply problem. Sections 4 and 5 discuss the rule-based approach to layup design and present a method of embedding such rules into the Genetic Algorithm. Finally Section 6 compares the performance of the GA and rule-based methods.

1. LAYUP DESIGN BY THREE METHODS

The first genetic representation considered was a mapping of plies to genes based on the process of "embryological" development: the decoding of an artificial chromosome (structure) into a solution (laminate) whose fitness can then be evaluated 1. This representation was favoured because (a) it ensured that all genetic operators produced valid genetic structures (in terms of physical layups) and (b) genetic sub-structures or building blocks would be correctly propagated during evolution of the final solution.

A different genetic algorithm (GA) approach, reported in this paper, is to relate the sequence of plies within a layup directly to the sequence of genes within a structure. This simple representation can support all genetic operators only when coupled with a "penalty function" that constrains the genetic algorithm to valid parts of the search space of all possible layup configurations. The penalty value is a function of the number of invalid plies in the current structure and reflects the "distance" the structure has to travel in configuration space to reach a correct (but not necessarily optimal) solution. The evaluation or fitness function applied to complete laminates in the GA system is the same as that applied by the random and greedy search algorithms, namely one that considers (a) the applied stress state, (b)
the set of failure mechanisms and (c) the tendency of the proposed layup to fail.

1.1. PRINCIPLED LAYUP BY PARTIAL EVALUATION

An alternative approach to the design of laminate layups is that of "principled synthetic design" where a layup is synthesized systematically by a knowledge based design system on a ply by ply basis. Every additional ply is considered individually so the system requires an evaluation of partial layups. This approach capitalizes on the fact that some of the most important evaluations for a ply can be made by considering only its immediate neighbours. A rule-based principled design tool COMPLEX using this method has been prototyped. The rule-base embedded in this prototype has been mapped into an equivalent GA evaluation function and the two systems compared in the design of a symmetric laminate under four different loading types.

2. CONSTRAINED GENETIC ALGORITHMS

Genetic algorithms (GAs) mimic some of the processes observed in natural evolution. Evolution takes place on chromosomes (structures) which are organic blueprints encoding the structure of living beings. A living being is created through the expression of the genes within these chromosomes. Key features of evolutionary theory that are implemented within the GA are -

1. evolution is a process that operate on chromosomes rather than on the living beings they encode.
2. natural selection is the link between chromosomes and the performance of their encoded structures. Processes of natural selection cause those chromosomes that encode successful structures to reproduce more often.
3. the process of reproduction is the point at which evolution takes place.
4. evolution has no memory - information on high performing individuals is carried in their chromosomes and the form of the natural selection process i.e the evaluation function.

Genetic algorithms adapt to the particular problem to be solved because the natural selection between generations results in the propagation of coadapted sequences of genes with a specific value (alleles) within a chromosome. The basic execution cycle at each system time step (epoch) is:

Step 1. From the existing population of structures, build a new mating pool by selecting structures probabilistically based on their fitness (as calculated by the evaluation function).

Step 2. Apply genetic operators (crossover, mutation) to randomly selected pairs from the mating pool, creating offspring structures to be placed into the new population.

Step 3. Replace the existing population with the new population.

Step 1 builds the mating pool using a randomized selection procedure that ensures that the expected number of times a structure is chosen is proportional to that structure's performance (relative to the rest of the population).

Step 2 ensures that new points in the configuration space are searched. The crossover operator works from two structures swapping a random length of components (genes) between two relatively successful parents to form new children that preserve substructures integral to high performing structures. The mutation operator randomly changes a single gene allele in a structure to produce a new child (and help to maintain genetic diversity within the population).

Step 3 creates the next generation ready for evaluation.

The laminate layup problem is a constrained optimization problem in that a large part of the potential configuration space is invalid due to violations in the number of plies with distinct orientations. Two approaches have been developed in the application of GAs to such problems - (a) modification of genetic operators and (b) application of a penalty to structures violating constraints.

2.1. MODIFIED GENETIC OPERATORS

The original theory of genetic algorithms developed by Holland presented search operators that were independent on the problem domain and treated all parts of the structure uniformly. Knowledge-augmented operators guide the genetic algorithm directly towards better structures using domain data that is separate from that encoded in the evaluation function. Such operators have been used in function optimization, pattern recognition, and scheduling problems. Augmented operators ensure that all structures generated through recombination (Step 2) are valid points in the search space of the problem domain. There are two drawbacks to their use - (a) a loss of generality in the GA system and (b) possible violation of the underlying theory governing the behaviour of the GA.

2.2. PENALTY FUNCTIONS

The basic technique for applying penalty functions to the GA is as follows. A linear combination is constructed of a cost function and a penalty function. The cost function is derived from the domain-dependent evaluation function. The penalty function reflects the validity of the structure and must be graded to preserve information within the population and enable the ranking of structures in terms of "incorrectness". General guidelines on the construction of penalty functions stress the need for penalties that are functions of the distance from feasibility and the number of violated
constraints. This technique is simpler to implement than that of modifying operators (by localizing amendments to the evaluation of structures) and preserves the generality and theoretical integrity of the GA.

2.3. REPRESENTATION ISSUES

The representation of each gene within a GA structure is the key step in developing a successful system. The representation should be chosen so that (a) the genetic recombination operators always produce valid structures and (b) so that genes that are physically close correspond to substructures (or building blocks) that should be conserved between generations. In the case of composite laminates where plies are their own genes, (b) is true whilst (a) is not. However the application of a penalty function permits the violation of (a) providing that the function ranks structures correctly and enables the selection procedure to correctly fill the mating pool.

Thus the approach taken this study was to take the most simple ply-to-gene representation and investigate whether a penalty function can be constructed that enables a successful (and robust) optimization of laminate structures to occur. In all experiments each ply was represented by a 2-bit gene encoding one of the permissible orientations. For experiments involving asymmetric laminates of n plies, each structure was 2*n bits in length. For experiments involving symmetric laminates of m plies, each structure was m bits in length with the structure being mirrored into the full laminate before evaluation.

2.4. THE EVALUATION FUNCTION

The evaluation function given below is the 'C' equivalent of that presented in an earlier report. The rationale behind the function is given in that report and is not reiterated here. The use of an equivalent function enables direct comparisons of the GA's performance to be made with that of the random and greedy algorithms.

```c
/* apply evaluation function */
for (i = 1, s = 0; i < PLYS; i++) {
    /* PLYS = total no. of plies */
    d = abs(plys[i-1] - plys[i]);
    /* plys holds ply data */
    c = 1.0 - (double) abs(d - 90) / 90.0;
    s += (100.0 * c * c);
}
```

```c
a = (double) abs (90 - plys[0]) / 90.0;
if (a > 75.0) { a *= a; }
```

```c
b = (double) abs (90 - plys[PLYS - 1]) / 90.0;
```

```c
s += 25.0 * (b < b);
```

```c
for (i = 1, pos = 0, neg = 0;i < PLYS; i++) {
    if (plys[i] - plys[i-1] > 0) pos++;
    if (plys[i] - plys[i-1] < 0) neg++;
}
```

```c
s += (double) abs (pos - neg) * 25 * PLYS) / (double) (PLYS - 1);
```

```c
score = s / (1.25 * (double) PLYS);
/* score = fitness of plys */
```

The maximum score for a complete layup is 100 which corresponds to the worst case. Hence the goal of the GA is to minimize fitness within the population of structures.

2.5. THE PENALTY FUNCTION

Five types of penalty function (P0 - P4) were investigated in this part of the study. In all cases the calculated penalty was added to the existing score to give an adjusted fitness as follows:

```c
fitness = score + penalty_factor * (double) penalty;
```

The penalty_factor in all cases was fixed at 1.0. The value of penalty was a function of the mismatch between actual totals of distinct ply orientations and pre-defined required totals. The basic function was defined by -

```c
/* apply penalty function */
for (i = 0, penalty = 0; i < ORIENT; i++) {
  /* ORIENT = number of */
  match = actual_plys[i] - permitted_plys[i];
  /* possible orientations */
  if (match != 0) penalty += P[abs(match)];
  /* P = penalty function */
}
```

with P defined as follows -

P0 : = 0 i.e. no penalty applied.
P1 : = match if match > 0 else = 0
P2 : = match
P3 : = match2 if match > 0 else = 0
P4 : = match2

3. PENALTY FUNCTION EVALUATION

The trial problem consisted of a layup with PLYS = 10 and ORIENT = 4. The permitted orientations were 0, 30, 45, 90 with specified totals of 3, 1, 3, 3 respectively. The best score for a optimum layup in this test was 5.6; the worst score was 52. This is exactly the same trial problem as previously explored using the random and greedy algorithms.

The GENESIS system was used as the vehicle for implementing the GA system, evaluation and penalty functions. Three different tests were conducted with each penalty function P0 - P5. These tests corresponded to three population strategies (S1 - S3) as follows -

S1 : Elitist; population size = 100.
S2 : Non elitist; population size = 100.
S3 : Non elitist; population size = 50.

Elitism in this context implies that the best individual of any generation is guaranteed to be propagated to the next. Each {P,S} experiment consisted of nine runs of the GENESIS system of 9000 trials.
(structure evaluations) each with a fixed limit of 180 generations. The parameters for each run within an experiment were identical except for a randomly set seed (used by the GENESIS pseudo-random number generator). Two genetic operators - crossover and mutation - were applied throughout the trial at fixed rates.

The behaviour of each \{P,S\} experiment was evaluated against three criteria - (1) the number of incorrect structures generated, (2) the number of optimal structures generated and (3) the fitness of the best structure. In each run, the best ten structures were analysed.

3.1. PENALTY FUNCTIONS VS STRUCTURE VALIDITY

The relationship between penalty /population strategy and structure validity is shown on Figure 1. All results are averaged over 9 runs and the worst case scenario is 10 out of 10 incorrect structures. It can be seen that a no penalty strategy (P0) leads to an underconstrained solution regardless of population strategy whilst the maximum strategy (P4) yields no incorrect structures for the elitist strategy and a maximum of 1 incorrect structure for non elitist strategies.

Intermediate penalty strategies (P1 - P3) are sensitive to population strategies. In all cases an elitist strategy results in fewer incorrect structures. Using a larger population in a non elitist strategy does not significantly effect the result and penalty strategy P2 offers the best overall performance across all population strategies.

3.2. PENALTY FUNCTIONS VS ROBUSTNESS

The relationship between penalty strategy and structure robustness is shown on Figure 2. Robustness is measured in terms of the number of times the GA finds the optimum solution for a specific penalty and population strategy. Applying penalty strategy P0 results in complete failure regardless of population strategy (since there exists invalid structures with 'better' fitness than the valid optimum). In the remaining cases (P1 - P4) the elitist strategy (S1) gives the best performance and the non elitist minimal population strategy (S3) gives the worst performance.

Penalty strategy P4 appears to overconstrain the problem by this criteria whilst strategies P1 - P3 offer approximately equivalent performance. Of particular interest here is the high performance of strategy P2/S2. Here a non elitist population strategy is balanced by the additional constraint imposed by the penalty function to produce an equivalent propagation of optimal structures to the strategy P1/S1.

3.3. PENALTY FUNCTIONS VS STRUCTURE FITNESS

The relationship between penalty strategy and structure fitness is shown on Figure 3. Fitness is recorded as the most fit structure in the best 10 structures averaged over nine test runs. The minimum fitness that can be achieved for valid structures is marked on the Figure with a dashed line (= 5.6). The first point to note is that all population strategies (S1 - S3) yield approximately equivalent results. Penalty strategy P0 produces structures with below minimum fitness indicating underconstraint of the problem. Penalty strategies P1 - P3 offer equal performance with P2 offering the most robust behaviour over varying population strategies. Penalty strategy P4 produces significantly worse performance indicating overconstraint of the problem by this criteria.
second part of the study - that of comparing the GA system with a Knowledge Based Laminate Design System.

4. A KNOWLEDGE BASED APPROACH TO LAYUP DESIGN

Existing analytic methods of optimizing the stacking sequences of composite layups work by uniquely minimizing either the interlaminar normal stresses or the interlaminar shear stresses. Ige$^{2, 14}$ has shown that there is a need in most practical cases to optimize the stacking sequence by simultaneously minimizing both stresses as well as by suppressing the onset of transverse ply cracks. Ige's Knowledge Based System (KBS) attempts to optimize each of the three parameters for a given type of load without compromising the design objectives using a pare-to-optimum method implemented on a simple rule-based KBS.

The three performance constraints are -
1. minimize normal stresses
2. minimize shear stresses
3. delay the onset and control the propagation of 90º ply cracks.

Each of these constraints is split into smaller constraint rules in the rule-base so that
(a) the system is as general as possible
(b) the rule-base can be updated very easily
(c) symbolic computation becomes more efficient.

4.1. SYSTEM OVERVIEW

The KBS operates in the following manner: all available alternatives for an unoccupied ply position are generated and each alternative is evaluated by a set of rules which represent performance constraints (measuring resistance to delamination). The ply alternatives are then ranked, based on how well they satisfy the constraints at that interface level (locally) and also on the laminate level (globally). The highest scoring alternative is chosen as the value for the unoccupied ply position and the process is repeated until the last ply is fixed. A modified depth first search routine is used in the KBS to search for the best choice of ply at all levels and this routine traverses only the best route through the search tree.

4.2. SYSTEM PERFORMANCE

The reliability of the system is a function of the quality of the rules in the rule base. The quality of each solution additionally depends upon the scoring mechanism applied to each ply. In its present implementation the choice of ply is dependent on how well it satisfies a constraint and on how important the constraint is. At present the KBS can design symmetric laminates under four different loading types -
1. uniaxial tension
2. uniaxial compression
3. uniaxial load
4. bending load

5 Artificial Intelligence in Engineering 01/1993

DOI: 10.1016/0954-1810(93)90020-G
To enable comparative tests with the GA system, the KBS has been tested on a trial problem defined by PLYS = 5 and ORIENT = 4 (symmetric layup implies generated laminate has PLYS = 10). The permitted orientations were 0, 45, 90, 135 with specified totals of 2, 1, 1, 1, respectively. Layup designs were produced by the KBS for each of the four loading types as follows -

1. 90, 45, 0, 135, 0
2. 45, 0, 135, 90, 0
3. 0, 45, 135, 90, 0
4. 0, 45, 90, 135, 0

In each case the design produced was optimal. The number of cycles taken to produce each layup is equal to the number of plies in the specification (PLYS). Unlike the design-by-search approach, no intermediate result is offered and the time taken to produce a solution is not dependent on the shape of the configuration space.

5. **MAPPING THE KBS ONTO A GA SYSTEM**

To produce a valid comparison between the KBS and GA systems, a mapping between the two was required in the domains of (a) layup representation, (b) layup evaluation and (c) search constraint. The same genetic structure representation was used as defined in section 2.3, namely direct ply to gene mapping. To ensure an equivalent evaluation across both systems, the KBS rule base was extracted and recoded as an evaluation function to be applied to each structure. Search constraint was implemented using the same approach discussed in section 2.5. Four different types of penalty functions were analysed -

- **P0** : = 0 i.e no penalty applied.
- **P1** : = match
- **P2** : = sum of (match²) across all plies
- **P3** : = (sum of matches across all plies)²

where

match = actual_plys[i] - permitted_plys[i].

The aim of the comparison was to investigate the performance of the KBS rule base designed for layup synthesis when it was applied in a generate and test manner in the GA system.

6. **COMPARATIVE RESULTS**

The GA system was run on the trial problem presented in section 4. Once again the GENESIS system was the implementation vehicle with a non elitist population strategy on a fixed population size of 50. The study consisted of running the GA under each penalty strategy and producing layup designs for each loading type. Each experiment consists of nine runs of the system with 9000 trials each with a fixed limit of 180 generations. The parameters for each run within an experiment were identical (except for the random number seed) and two operators - crossover and mutation - were applied throughout the run at fixed rates.

The behaviour of the system was evaluated against three criteria - (1) the number of incorrect structures generated, (2) the number of optimal structures generated and (3) the rate of system convergence to an optimal solution. In each run, the best ten structures were analysed.

### 6.1. PEnalty Functions VS Structure Validity

The relationship between penalty strategy and structure validity for the four loading types is shown on Figure 4. All results are averaged over 9 runs and the worst case scenario is 10 out of 10 incorrect structures. The no-penalty strategy (P0) leads to an underconstrained solution regardless of the loading type. The behaviour of other penalty strategies (P1 - P3) is dependent on loading type. All strategies cope well with the uniaxial load problem indicating that the configuration space for this problem requires minimal constraint. Penalty P3 (strong) performs best on the uniaxial compression and bending load problems whilst penalty P2 (medium) performs best on the uniaxial tension problem. Penalty P1 underconstrains the problem and is non optimal under all load conditions.

### 6.2. PEnalty Functions VS Robustness

The relationship between penalty strategy and structure robustness is shown on Figure 5. Robustness is measured in terms of the number of times the GA finds the optimum solution for a specific penalty strategy and loading type. Applying penalty strategy P0 results in complete failure across all loading types. In all other cases with the exception of bending load, increasing the constraint produces better performance. In the case of bending load, the medium penalty strategy (P2) appears to offer superior performance to the strong strategy (P3). This is unexpected given the very poor performance of strategy P1 on this problem - indicating the need for strong constraint.
6.3 PENALTIES VS CONVERGENCE RATE

The relationship between penalty strategy and structure fitness convergence is shown on Figure 6. Convergence is measured in terms of the generation in which the optimum solution is found. A generation value of 180 implies that the optimum was never discovered. Penalty P0 fails to produce the optimum solution regardless of loading type since underconstraint results in the production of invalid structures of below optimum fitness. A general trend can be seen in the case of the uniaxial tension, uniaxial compression and bending load results. Increasing constraint appears to improve the rate of convergence with uniaxial tension being the easiest problem to solve. The uniaxial load result bucks this trend however with increasing constraint leading to worse performance. This would tend to confirm the hypothesis mooted in 6.1 - that the uniaxial load problem is easier to solve than the others and its configuration space requires less constraint.

6.4 DISCUSSION

When measured against the three structure performance criteria - validity, robustness, convergence - the penalty strategies can be ranked in order of decreasing severity. A weak penalty strategy (P1) is outperformed under all loading conditions by stronger ones. The difference between a square of sums (P3) and sum of squares (P2) strategy is less marked but the stronger strategy offers more robust performance across most loading types. A point to note in these experiments is that even a weak strategy (P1) is capable of producing valid and optimal structures. This suggests that this layup problem is rather too simple to permit an adequate analysis of the varying strategies. This is a limitation of the KBS (in its current state) rather than the GA system. When the KBS is extended to handle non symmetric layups the problem configuration space will grow rapidly and provide a more challenging environment for the GA system.

CONCLUSION

This research has analysed the feasibility of applying a genetic function optimizer to the problem of laminate layup design. Two different evaluation functions have been studied in the exploration of the layup configuration space - (1) a generate and test function based on a global analysis of the laminate and (2) a synthetic design rule base implementing both local and global (meta) design rules. In both cases the GA has been able to find optimal solutions when harnessed with a penalty function that constrains the search space by penalizing the generation of invalid structures (having incorrect ply orientations).

The problems considered here have been of a simple nature and have not required the use of more subtle representations (e.g. a generative “embryological” representation) or genetic operators (such as the inversion/reordering operator). Future research will consider large laminates with vastly increased configuration spaces. The GENESIS system will be enhanced to include an inversion operator and the tuning of the strongest penalty strategy (P3) investigated by dynamic adjustment of the penalty factor.

REFERENCES

8. Schaffer, J.D. and Morishima, A., An adaptive crossover distribution mechanism for genetic algorithms, in Proc. 2nd


