International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 3, Issue 5(May 2014), PP.77-86

Optimal Design of Laminated Composite Plates by Using Advanced Genetic Algorithm

T.Swathi [1], Mr. K. Sudhakar Reddy [2,] Mr. P.Suresh[3]
Professor, Department Of Mechanical, Faculty Of HITS ,Hyderabad ,India.
Professor, Department Of Mechanical, Faculty Of HITS, Hyderabad,India.
Asso. Professor, Department Of Mechanical, Faculty Of HITS, Hyderabad India.

Abstract: Composite laminate design and optimization requires discrete programming in order to find the correct number of plies with thicknesses, orientation angles, and material types which are usually restricted to a discrete set of values. Genetic algorithms are one of the few optimization tools available that are well suited to such discrete problem solving environments. The main goal of this work was to try and improve the computational efficiency of the GA (its most serious drawback), and demonstrate the GA's ability to be easily adapted to different types of composite laminate design optimization problems. Two different versions of a genetic algorithm, GA-I and GA-II, were developed specifically to accomplish these tasks.

To demonstrate the flexibility of the GA structure, the GA-II algorithm was devised to handle more complex composite laminate configurations constructed from multiple materials. The modified GA utilized two chromosome strings to represent the composite laminate. The first string defined the orientation angle of each ply, and the second string defined a ply's material type. By using two different chromosome strings, only small modifications to the various genetic operators were required. The two chromosome string concept was tested on a multi-objective optimization problem which involved minimizing laminate cost and weight of a simply supported composite plate subjected to various in-plane loading conditions, and numerous constraints. The cost of the laminate was based on the required lay up time and by its material make-up. To achieve high performance laminates at the lowest cost, two materials were allowed in the stacking sequence, one with high strength and cost, the other with lower strength and cost. The optimization formulation was carried out by determining separate cost and weight objective functions. A convex combination of these two objectives was used for laminate fitness, and thus required no additional modifications to the GA. To obtain a Pareto- optimal set of designs, the influence of cost and weight on the overall fitness of a laminate configuration is adjusted from one extreme to the other by adjusting the convex combination accordingly.

The objective of this Paper is to devise a genetic algorithm for stacking sequence design of symmetrically laminated composite plates. Stacking sequence design implies the determination of the number of plies in the laminate as well as their orientation. With this feature, the GA may be used to control laminate weight by adjusting the number of plies in the laminate stacking sequence. The genetic algorithm will not be allowed to adjust the dimensions of the plate throughout the optimization process. Two different versions of a genetic algorithm are explored.

I. COMPOSITE LAMINATES COMPRISED OF ONE MATERIAL

For the GA-I algorithm, one string of genes is used to represent one half of a symmetrically laminated composite plate. The length of the gene string is kept fixed throughout the optimization process. Each gene in the string is represented by an integer value between 0 and 10 and determines whether the ply stack location is empty or occupied with a 3-ply stack which may be oriented at any angle between 0^0 and 90^0, in increments of 10^0, see Figure1. Although the gene string length is fixed, having empty plies makes it possible to change the laminate thickness during the optimization process. Coding the ply orientation angles as consecutive integers is not really necessary, since all the genetic operators could apply directly to the angles. However, the implementation of random choices with given probabilities is both easier to describe and program for integer intervals than for arbitrary sets of objects[1].

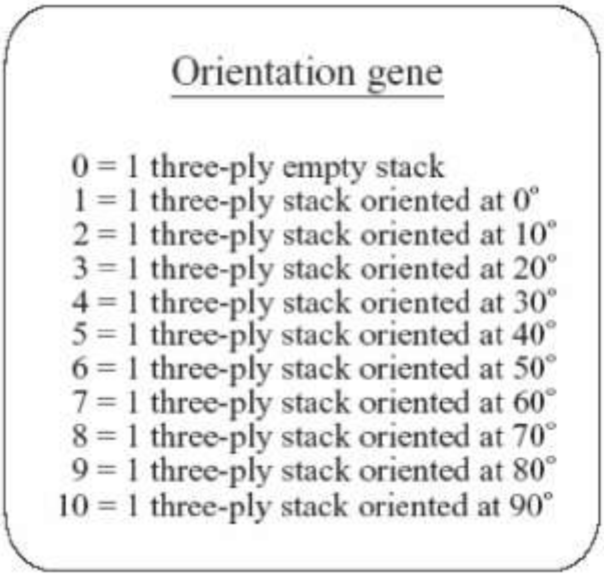
All plies in the stacking sequence have the same prescribed thickness value 0^0, 103^0 , etc.) to keep the number of and are stacked in groups of three (i.e., 03^0 , 203 design variables used and the size of the design space to a minimum. An example of a decoded stacking sequence is given in Figure 5.2, where E represents an empty 3-ply stack. Note that empty stacks are pushed to the outer edge (left end) of the laminate stacking sequence to avoid having voids in the laminate.

1.1 GA Procedure

An initial population of genetic strings with randomly chosen genes is created first. The size of the population used in the present work remains constant throughout the genetic optimization. Various genetic operators are applied at given probabilities to generate new laminates. In order to form successive generations, parents are chosen from the current population based on their fitnesses, as described in the following subsection. The fitness calculation usually involves function values that are determined from separate analysis subroutines or packages. Next, the crossover, mutation, and ply swap operators are applied to create child designs, who are hopefully better suited to their environment than their parents. The child population is then analyzed and ranked. To complete the generation cycle, a selection scheme is implemented which determines which laminates from the child and parent population will be placed in the next generation. One generation after another is created until some stopping criterion is met. A schematic of the genetic algorithm procedure is given in Figure 3.

1.2 Parent Selection

Parent selection is accomplished using a roulette wheel concept. This method of selection differs from other evolutionary algorithms because it gives every member of the population a chance to become a parent (i.e., a non-extinctive breeding procedure).



Orientation gene

0 = 1 three-ply empty stack
1 = 1 three-ply stack oriented at 0°
2 = 1 three-ply stack oriented at 10°
3 = 1 three-ply stack oriented at 20°
4 = 1 three-ply stack oriented at 30°
5 = 1 three-ply stack oriented at 40°
6 = 1 three-ply stack oriented at 50°
7 = 1 three-ply stack oriented at 60°
8 = 1 three-ply stack oriented at 70°
9 = 1 three-ply stack oriented at 80°
10 = 1 three-ply stack oriented at 90°

Figure 1: GA-I code key for laminate stacking sequence.

Coded Orientation: $[0/3/5/2/7]_s$

Decoded Orientation: $[E_3/20^\circ_3/40^\circ_3/10^\circ_3/60^\circ_3]_s$

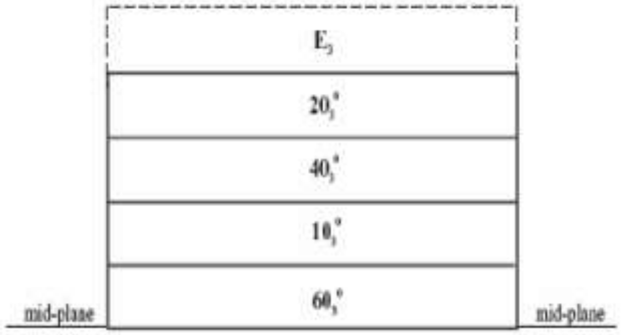


Figure 2: Sample stacking sequence arrangement for GA-I.

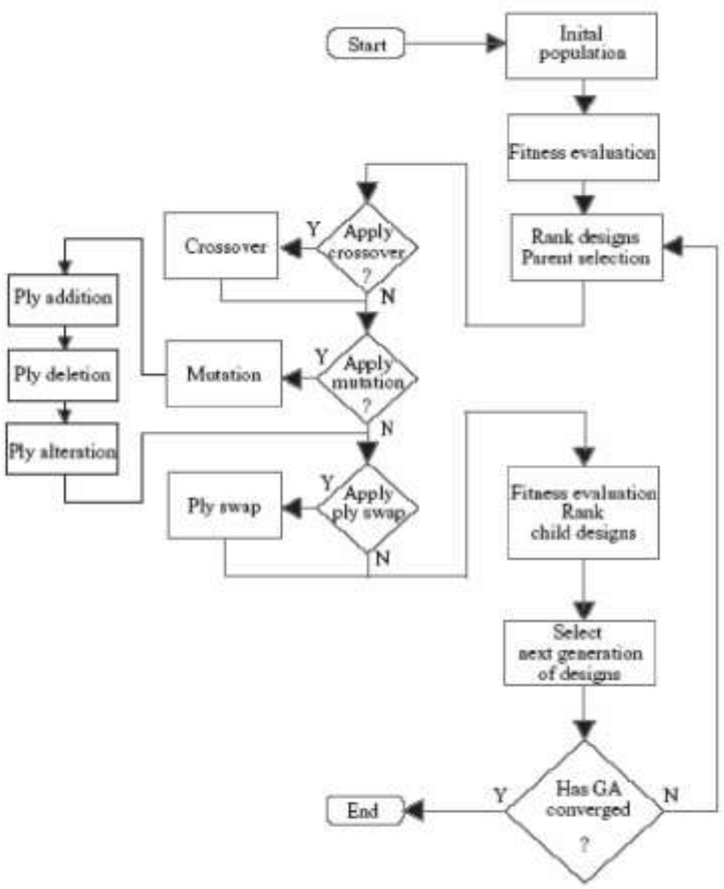


Figure 3: Genetic algorithm procedure.

Before parent selection can begin, all laminates must be ranked from best to worst according to the value of each laminate's objective function [2]. A roulette wheel is implemented where the i^{TH} ranked laminate in the population is given an interval [ĭi-1, ĭi), whose size depends on the population size, P, and its rank, i, in the population:

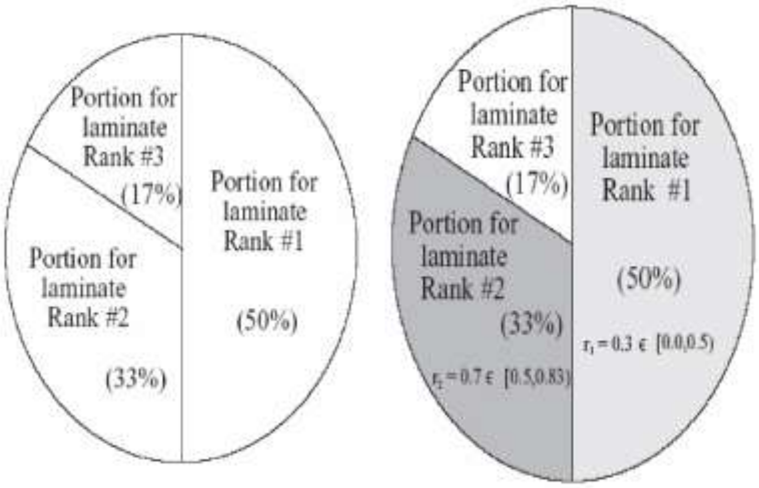
$$\phi_i = \phi_{i-1} + \frac{2(P-i+1)}{P(P+1)},$$

Φ_i=0 and i = 1,……, P. For example, if there are three laminates in a population, the roulette wheel is ivided into three pieces with the best laminate taking 50% of the wheel, the second best taking 33%, and the oorest taking 17%, see Figure 4-a.

A uniformly distributed random number is generated between 0 and 1; laminate i is selected as a parent if the number lies in the interval [ĭi-1, ĭi), Continuing with the above example, if random numbers r1 = 0.3 ò [0, 0.5) and r2 = 0.7 ò [0.5, 0.83) are drawn, then laminate 1 and laminate 2 will become parents of the first child, see the shaded regions in Figure 5.4-b. Parents of a child are required to be distinct laminates from the population [3].

1.2 Crossover

Children are created by combining a portion of each parent's genetic string in an operation called one-point crossover. To determine the crossover point, a uniformly distributed random number is chosen and then multiplied by one less than the maximum number of non-empty genes in the two parents. The integer ceiling value of this product determines the crossover point, see Figure 5.5. The gene string is then split at the same point in both parents. The left piece from parent 1 and the right piece from parent 2 are combined to form a child laminate. To ensure that empty plies are not swapped, all empty plies are pushed to the left side of the coded string (this corresponds to



a) Roulette wheel distribution for 3 laminates. b) Parent selection using random numbers, r1 and r2.

Figure 4: Parent selection using a roulette wheel.

THe outer edge of the laminate). The random crossover point is restricted to fall in the non-empty region of both parent laminates to ensure that the child laminate is unique, see Figure 5 .5. If, during the creation of the child population, crossover is not applied then one of the parent laminates is cloned into the child string. Child laminates are also forced to be distinct from each other and from laminates in the parent population. If a distinct child cannot be found after a prescribed number of iterations, then one of the parents is cloned into the child population also. The crossover process is repeated as many times as necessary to create a new population of laminates [4].

1.3 Mutation

After a child is created, the operations of adding, deleting, or mutating genes occur with small probabilities. These operators make up genetic mutation, and are illustrated in Figure 6. When adding a ply stack, a uniform random number is chosen to determine the orientation. For the design problems considered in this work, outer plies in the laminate will get set up faster because they have a greater influence on the objective function. Thus, added ply stacks are always introduced at the mid-plane of the laminate, see Figure .6-a.

To delete a ply stack, a random number is chosen and the corresponding stack is removed from the stacking sequence by replacing it with a 0 gene. The laminate is then re-stacked so that all empty plies are pushed to the outer edge of the laminate, see Figure 5.6-b.

Gene alteration is shown in Figure 6-c. Each gene in the string switches with a small probability to any other permissible integer value (as defined in Figure 1) except [5].

0 and the value of the gene before ply alteration occurs. Ply alteration does not operate on empty genes either.

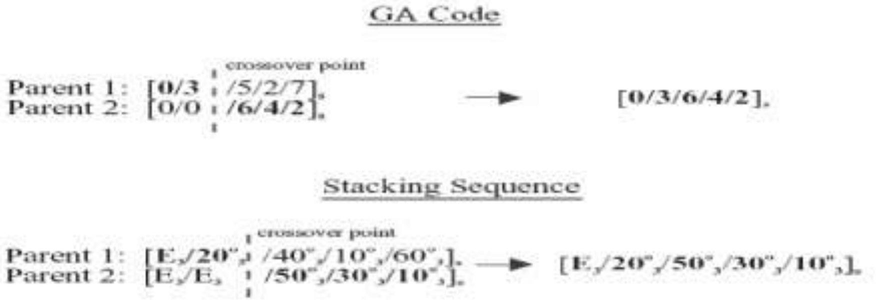


Figure 5: One-point crossover.

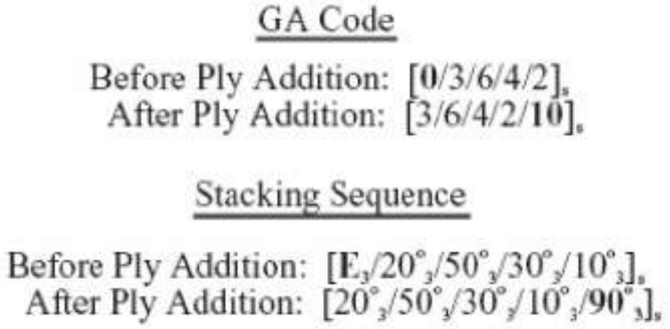
1.4 Ply Swap

In previous works, a permutation operator (see Figure.2) was often used to aid the genetic search but was found to shuffle the digits in the gene string too much [12]. Thus, a less disruptive operator, ply swap, was designed and will be used in favor of permutation for the design problems considered in this work. The ply swap operator is implemented by randomly selecting two genes in the string and switching their positions, see Figure.7. Ply swap can be effective for problems where certain parts of the laminate stacking sequence get set up faster than others [6]. For example, if the optimal stacking sequence for the outer section of the laminate has been determined first (as is the case for laminate design problems which involve bending), the ply swap operator may help the GA determine the optimal orientations for the inner part of the laminate by swapping plies from each section.

1.5 Composite Laminates with Multiple Materials

In this section, modifications to the GA-I algorithm to allow for stacking sequences with multiple materials will be discussed. The second version of the genetic algorithm will be called GA-II. In the previous section, the entire laminate was comprised of one material. Thus, one chromosome consisting of one gene was sufficient to represent the laminate stacking sequence. However, to accommodate two or more materials, each chromosome is expanded to include two gene strings, one for ply orientation and another one for material definition. The representation of genes by integers in each string is maintained. Genes in the first string will once again determine whether the ply location is empty or filled with a ply of prescribed orientation. Corresponding genes in the second string determine the ply material if the ply is present. By employing two gene strings, the number of materials that may be used in the stacking sequence may be changed easily by adjusting the size of the material gene alphabet [7].

In the application of the two material design problem, single ply stacks are used instead of stacks of 3 plies, with ply orientation choices of 0^0, through 90^0 with □ 15^0 increments, see Figure 8. Ply thickness may take one of two prescribed values depending on the material that a ply is comprised of, as shown in the sample stacking sequence of Figure 9.



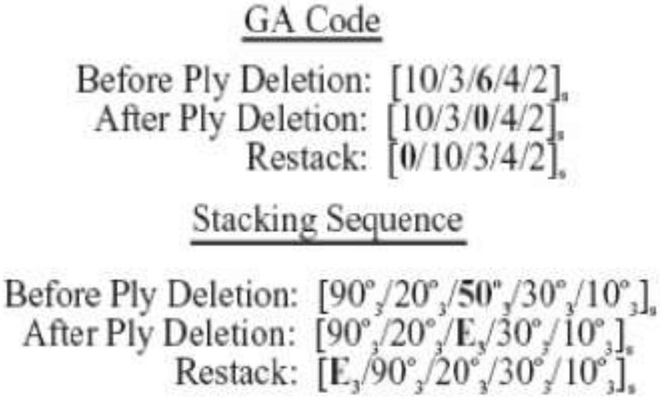
GA Code

Before Ply Addition: $[0/3/6/4/2]_s$
After Ply Addition: $[3/6/4/2/10]_s$

Stacking Sequence

Before Ply Addition: $[E_3/20^\circ_3/50^\circ_3/30^\circ_3/10^\circ_3]_s$
After Ply Addition: $[20^\circ_3/50^\circ_3/30^\circ_3/10^\circ_3/90^\circ_3]_s$

a) Ply Addition (at least 1 empty stack)



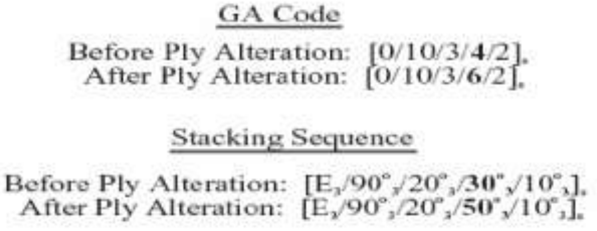
GA Code

Before Ply Deletion: $[10/3/6/4/2]_s$
After Ply Deletion: $[10/3/0/4/2]_s$
Restack: $[0/10/3/4/2]_s$

Stacking Sequence

Before Ply Deletion: $[90^\circ_3/20^\circ_3/50^\circ_3/30^\circ_3/10^\circ_3]_s$
After Ply Deletion: $[90^\circ_3/20^\circ_3/E_3/30^\circ_3/10^\circ_3]_s$
Restack: $[E_3/90^\circ_3/20^\circ_3/30^\circ_3/10^\circ_3]_s$

b) Ply Deletion (at least 2 full stacks)



GA Code

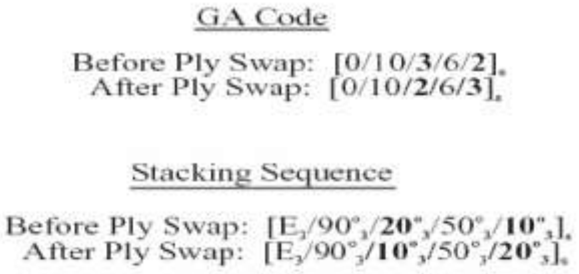
Before Ply Alteration: $[0/10/3/4/2]_s$
After Ply Alteration: $[0/10/3/6/2]_s$

Stacking Sequence

Before Ply Alteration: $[E_3/90^\circ_3/20^\circ_3/30^\circ_3/10^\circ_3]_s$
After Ply Alteration: $[E_3/90^\circ_3/20^\circ_3/50^\circ_3/10^\circ_3]_s$

c) Single Ply-Stack Alteration (filled plies only)

Figure.6: Mutation



GA Code

Before Ply Swap: $[0/10/\mathbf{3}/6/\mathbf{2}]_s$
After Ply Swap: $[0/10/\mathbf{2}/6/\mathbf{3}]_s$

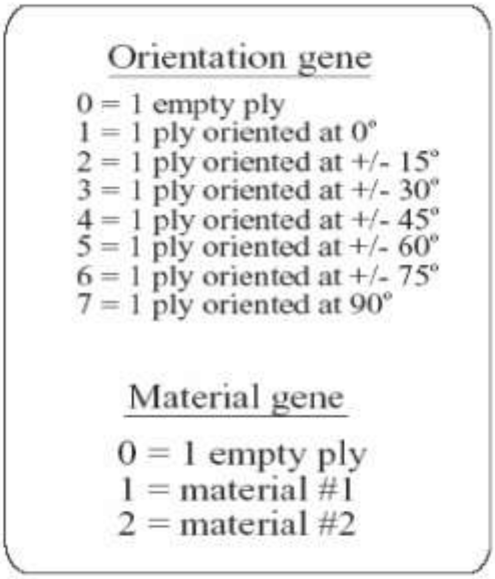
Stacking Sequence

Before Ply Swap: $[E_3/90°_3/\mathbf{20°_3}/50°_3/\mathbf{10°_3}]_s$
After Ply Swap: $[E_3/90°_3/\mathbf{10°_3}/50°_3/\mathbf{20°_3}]_s$

7: Ply swap.

2.1 Decoding the Gene Strings

To incorporate the two material concept into the GA, a more complex decoding procedure was required. Orientation genes that are coded as integer values between 2 and 6 represent either the positive or the negative value of the corresponding ply orientation angle defined in Figure 5.8. For example, a 4 represents either a $+45^0$ or aí45^0 ply in the orientation gene. For the problem considered in Paper 5 laminates will be constrained to have a balanced stacking sequence to simplify analysis procedures. To maintain a balanced laminate or obtain a laminate as close to balanced as possible, the □ θ plies are decoded alternately. For example, the first 4 (starting from the outer edge of the laminate) encountered for a particular material is decoded as a $+45^0$ ply and the next 4 for the same material is decoded as í45^0 and so on. This decoding methodology applies for all plies accept those oriented at 0^0 and 90^0, which are decoded in the normal fashion. Thus, a laminate stacking sequence of □ θ plies is balanced if each + θ ply is matched with a - θ , ply of the same material. If the □ θ plies are balanced for one material but not the other, the laminate is unbalanced.



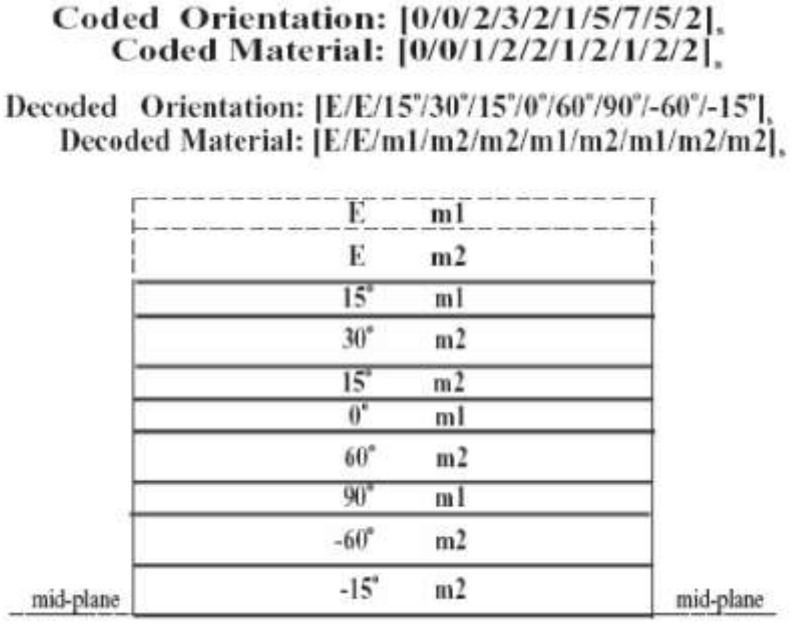
Orientation gene

0 = 1 empty ply
1 = 1 ply oriented at 0°
2 = 1 ply oriented at +/- 15°
3 = 1 ply oriented at +/- 30°
4 = 1 ply oriented at +/- 45°
5 = 1 ply oriented at +/- 60°
6 = 1 ply oriented at +/- 75°
7 = 1 ply oriented at 90°

Material gene

0 = 1 empty ply
1 = material #1
2 = material #2

Figure 5.8: GA-II code key for laminate stacking sequence.



Coded Orientation: [0/0/2/3/2/1/5/7/5/2],
Coded Material: [0/0/1/2/2/1/2/1/2/2],

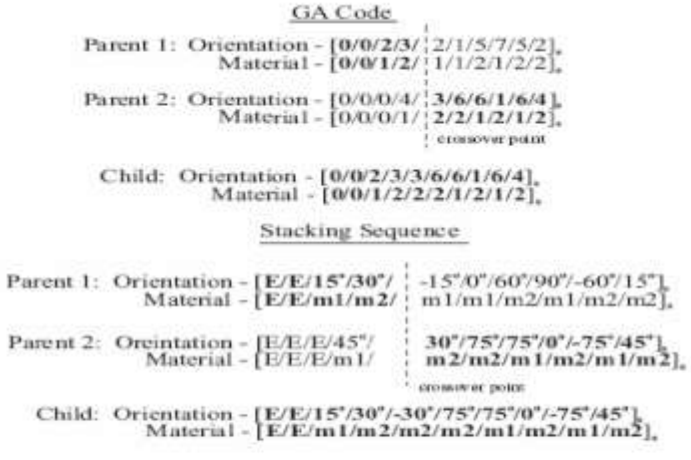
Decoded Orientation: [E/E/15°/30°/15°/0°/60°/90°/-60°/-15°],
Decoded Material: [E/E/m1/m2/m2/m1/m2/m1/m2/m2],

Figure 5.9: Sample stacking sequence arrangement for GA-II.

2.2 Modifications to the genetic operators

The procedure for the GA-II algorithm remains mostly unchanged from the one used in the GA-I version, except for small modifications made to the genetic operators [8]. When a parent is selected for reproduction, both the ply orientation gene string and material gene string are used when creating a child. In the crossover procedure, the orientation and material gene strings are split at the same point in both parents. The left pieces of both the orientation and material gene strings from parent one and the corresponding right pieces from parent 2 are then combined to form a child laminate, see Figure 10. Crossover may also have the effect of changing some +ș plies to íș also. This phenomenon is also seen in Figure 2.10 if one looks at the plies that are coded as 3 in both parents. In the second parent there is only one 3 for material 2 which gets decoded as +30^{0} ply. When the child is created, the ply coded as a 3 that is passed from parent 2 now gets decoded as a í30^{0} ply. This is because parent 1 also passed a ply to the child laminate that was coded as a 3 made of the same material. Since the 30^{0} ply from parent 1 will be closer to the outer edge of the laminate, it will be decoded as +30^{0} where as the second 3, which came form parent 2 gets decoded as í30^{0}.

The procedure for the mutation operator is modified slightly also. Ply addition and deletion are done simultaneously on both the orientation and material gene strings. Added plies are once again introduced at the mid-plane of the laminate. When a ply is added, the corresponding material gene is also added, see Figure 11. When a ply is deleted, it is picked at random with the corresponding material gene also being deleted, see Figure 12. Gene alteration is implemented separately on each gene string, with the same or different probabilities. If a gene is altered in the orientation gene, the corresponding material gene may not necessarily be altered, see Figure 13. Furthermore, when genes are switched in the ply swap operator, both the orientation and material genes are swapped simultaneously, see Figure 14. As in crossover, the other genetic operators may switch the sign on the orientation angle of a ply when they are applied, see for example the plies coded as 3 in the ply alteration procedure depicted in Figure 14.



GA Code

Parent 1: Orientation - [0/0/2/3/ 2/1/5/7/5/2],
Material - [0/0/1/2/ 1/1/2/1/2/2],

Parent 2: Orientation - [0/0/0/4/ 3/6/6/1/6/4],
Material - [0/0/0/1/ 2/2/1/2/1/2],

Child: Orientation - [0/0/2/3/3/6/6/1/6/4],
Material - [0/0/1/2/2/2/1/2/1/2],

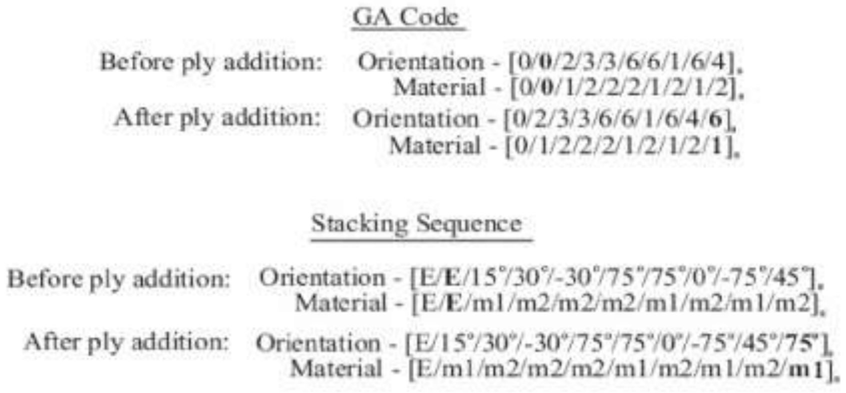
Stacking Sequence

Parent 1: Orientation - [E/E/15°/30°/ -15°/0°/60°/90°/-60°/15°],
Material - [E/E/m1/m2/ m1/m1/m2/m1/m2/m2],

Parent 2: Oreintation - [E/E/E/45°/ 30°/75°/75°/0°/-75°/45°],
Material - [E/E/E/m1/ m2/m2/m1/m2/m1/m2],

Child: Orientation - [E/E/15°/30°/-30°/75°/75°/0°/-75°/45°],
Material - [E/E/m1/m2/m2/m2/m1/m2/m1/m2],

Figure 10: Modified crossover operator.



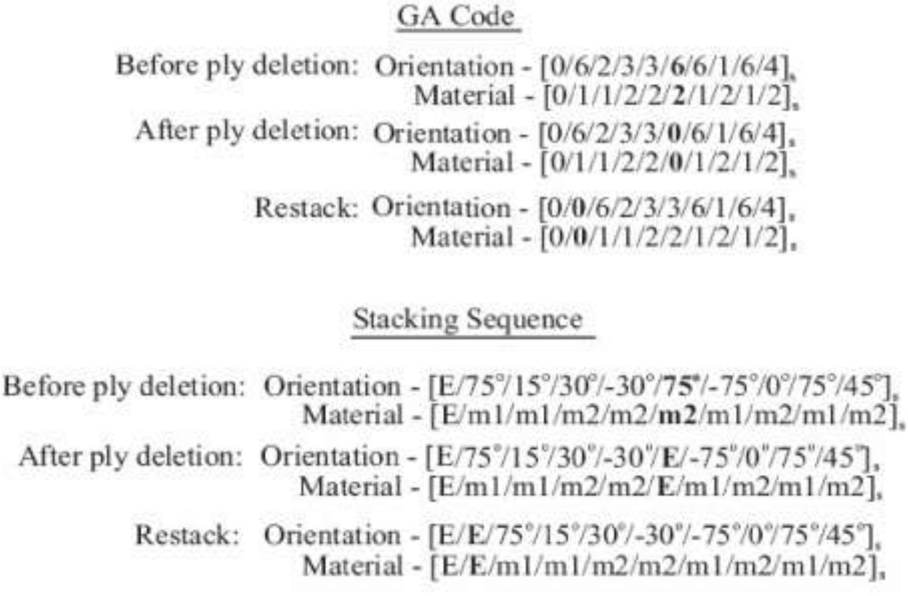
GA Code

Before ply addition: Orientation - [0/0/2/3/3/6/6/1/6/4],
Material - [0/0/1/2/2/2/1/2/1/2],
After ply addition: Orientation - [0/2/3/3/6/6/1/6/4/6],
Material - [0/1/2/2/2/1/2/1/2/1],

Stacking Sequence

Before ply addition: Orientation - [E/E/15°/30°/-30°/75°/75°/0°/-75°/45°],
Material - [E/E/m1/m2/m2/m2/m1/m2/m1/m2],
After ply addition: Orientation - [E/15°/30°/-30°/75°/75°/0°/-75°/45°/75°],
Material - [E/m1/m2/m2/m2/m1/m2/m1/m2/m1],

Figure 11: Modified mutation operator - ply addition



GA Code

Before ply deletion: Orientation - [0/6/2/3/3/6/6/1/6/4],
Material - [0/1/1/2/2/2/1/2/1/2],
After ply deletion: Orientation - [0/6/2/3/3/0/6/1/6/4],
Material - [0/1/1/2/2/0/1/2/1/2],
Restack: Orientation - [0/0/6/2/3/3/6/1/6/4],
Material - [0/0/1/1/2/2/1/2/1/2],

Stacking Sequence

Before ply deletion: Orientation - [E/75°/15°/30°/-30°/75°/-75°/0°/75°/45°],
Material - [E/m1/m1/m2/m2/m2/m1/m2/m1/m2],
After ply deletion: Orientation - [E/75°/15°/30°/-30°/E/-75°/0°/75°/45°],
Material - [E/m1/m1/m2/m2/E/m1/m2/m1/m2],
Restack: Orientation - [E/E/75°/15°/30°/-30°/-75°/0°/75°/45°],
Material - [E/E/m1/m1/m2/m2/m1/m2/m1/m2],

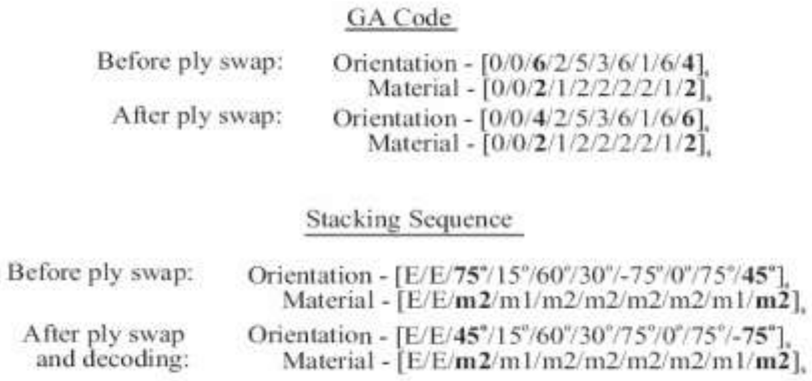
Figure 5.12: Modified mutation operator - ply deletion.

III. OTHER IMPLEMENTATION CONCERNS

Aside from the modifications addressed for the inclusion of multiple material in the laminate stacking sequence, the genetic procedures for both the GA-I and GA-II algorithms are largely the same. However, there are some general implementation concerns pertaining to both versions of the algorithm that need to be discussed and are the focus of this section [9].

3.1 Selecting a Stopping Criterion for the GA

The first issue is the stopping criterion for the genetic algorithm. The genetic search may be stopped after a prescribed number of iterations with no improvement of the top design in the population. This stopping criterion is well suited for measuring the search if an effort to improve the efficiency of the GA is being made. A simpler stopping criterion is to use an upper bound on the total number of function evaluations conducted by the GA. The second stopping criterion may be preferred when conducting a number of independent searches, and simplifies the calculation of the statistics of the GA at the end of the search since each optimization run will have the same number of generations. The first stopping criterion will be used in Paper



GA Code

Before ply swap: Orientation - [0/0/**6**/2/5/3/6/1/6/**4**],
Material - [0/0/**2**/1/2/2/2/2/1/**2**],

After ply swap: Orientation - [0/0/**4**/2/5/3/6/1/6/**6**],
Material - [0/0/**2**/1/2/2/2/2/1/**2**],

Stacking Sequence

Before ply swap: Orientation - [E/E/**75°**/15°/60°/30°/-75°/0°/75°/**45°**],
Material - [E/E/**m2**/m1/m2/m2/m2/m2/m1/**m2**],

After ply swap and decoding: Orientation - [E/E/**45°**/15°/60°/30°/75°/0°/75°/**-75°**],
Material - [E/E/**m2**/m1/m2/m2/m2/m2/m1/**m2**],

3.2 Fitness Calculation

The weight of the laminate can be implicitly or explicitly defined in the objective function for a laminate. Although the thinnest laminates will yield the best performance, they are heavily penalized if the material fails under the given loading condition [10]. Thus, the laminates that yield the best performance without failing the material strength constraint will automatically be the lightest (i.e.,have the fewest number of plies).

For the multi-objective optimization problem presented in Paper 6, two objective functions will be utilized. The first objective function will explicitly contain the weight of the laminate by counting the total number of plies in the stacking sequence. A second function will contain information about the manufacturing and material cost of the laminate. The physical weight and cost the of the laminate are then adjusted using information pertaining to the buckling and strength constraint satisfaction of the laminate [11]. The objective functions are then scaled by the corresponding objective functions of a nominal design to ensure that the cost and weight of the laminate are represented accordingly. The overall fitness of the laminate is obtained as a convex combination of the two objective functions. The convex combination can then be adjusted to allow cost and weight to contribute to the fitness calculation in any desired manner [12]. An in-depth discussion is given in Paper 6.

REFERENCE

[1]. Alliot, J., Gruber, H., Joly, G., and Schoenauer, M., “Genetic Algorithms for Solving Air Traffic Control Conflicts." Proceeding of the 9th Conference on AI Applications, 1993.

[2]. Davis, L., “Job Shop Scheduling with Genetic Algorithms." Proceedings of the First International Conference on Genetic Algorithms, 1985, pp. 136-140.

[3]. Giffer, B. and Thompson, G. L., “Algorithms for Solving Production Scheduling Problems." Operations Research, Vol. 8, 1969, pp. 90-99.

[4]. Jones, R.M., “Mechanics of Composite Materials." Hemisphere, New York, 1975.

[5]. “Boeing 777 roll out" Aviation Week and Space Technology, April 11, 1994, pp. 36-58.

[6]. Holland, J. H., “Adaptation in Natural and Artificial Systems." The University of Michigan Press, Ann Arbor, MI, 1975.

[7]. Goldberg, D. E., “Genetic Algorithms in Search, Optimization, and Machine Learning." Addison-

Wesley Publishing Company, Inc., Reading, MA, 1989.

[8]. Le Riche, R., “Optimization of Composite Structures by Genetic Algorithm." Ph.D. Dissertation, Virginia Polytechnic Institute and State University, 1994.

[9]. G¨urdal, Z., Haftka, R. T., and Hajela, P., “Design and Optimization of Laminated Composite Materials." To be published.

[10]. Le Riche, R., Haftka R. T., \Optimization of Stacking Sequence Design for Buckling Load aximization by Genetic Algorithm." Proceedings of the 33rd AIAA/ASME/ASCE/AHS/ASC SDM Conference, Dallas, TX, 1992.

[11]. Orvosh D. and Davis L., “Shall We Repair? Genetic Algorithms, Combinatorial Optimization, and Feasibility Constraints." Proceedings of the 5th International Conference on Genetic Algorithms, University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, 1993, pp. 650.

[12]. Nagendra, S., Haftka, R. T., and G¨urdal, Z., “Stacking Sequence Optimization of Simply Supported Laminates with Stability and Strain Constraints. " AIAA Journal, Vol. 30, No. 8, 1992, pp. 2132{2137.