JOURNAL OF AIRCRAFT
Vol. 36, No. 1, January–February 1999

Nongradient Methods in Multidisciplinary Design Optimization—Status and Potential

Prabhat Hajela*
Rensselaer Polytechnic Institute, Troy, New York 12180-3590

A number of multidisciplinary design optimization (MDO) problems are characterized by the presence of discrete and integer design variables, over and beyond the more traditional continuous variable problems. In continuous variable design problems, the design space may be nonconvex or even disjointed. Furthermore, the number of design variables and constraints may be quite large. The use of conventional gradient-based methods in such problems is fraught with hazards. First, these gradient-based methods cannot be used directly in the presence of discrete variables. Their use is facilitated by creating multiple equivalent continuous variable problems; in the presence of high dimensionality, the number of such problems to be solved can be quite large. Second, these methods have a propensity to converge to a relative optimum closest to the starting point, and this is a major weakness in the presence of multimodality in the design space. This paper primarily focuses on the use of nontraditional optimization methods in such problems, broadly classified today as soft computing strategies. These methods include techniques such as simulated annealing, genetic algorithms, Tabu search, and rule-based expert systems. It also examines issues pertinent to using these methods in MDO problems.

Introduction

THE field of multidisciplinary optimization has witnessed a healthy growth over the past decade.[1–4] This activity has been largely motivated by a recognition that the design and development of a complex system, which typically includes analysis input from many different disciplines, can no longer be conducted in isolation. An increased level of complexity within subdisciplines, and the need to extract the advantages of a synergistic design process, dictate the need for a more comprehensive strategy. Generic solution strategies for multidisciplinary systems[4,5] have evolved more recently, and are applicable to the problem of multidisciplinary design. In general, the multidisciplinary design optimization (MDO) problem is characterized by the presence of a large number of design variables and constraints, and analysis from various contributing disciplines that are not completely independent; one- or two-way couplings may exist between two or more disciplines. In an approach where the coupled multidisciplinary problem is treated as a single, large-scale optimization problem, the following difficulties have been identified.

1) The dimensionality of the design space may increase to a degree that obtaining reliable solutions to the optimization problem is rendered questionable when using traditional mathematical programming methods. Furthermore, there is a diminished capacity to evaluate the acceptability of solutions in higher-dimensionality design spaces.

2) For an iterative analysis environment that is typical of optimization, the presence of coupling between disciplines would introduce an inner loop of iteration (analysis iteration) that adds to the computational costs.

3) Although not specific to multidisciplinary design problems alone, in many instances the design space can be nonconvex or even disjointed. Such characteristics call for the use of nontraditional search techniques that do not have a propensity to seek the nearest relative optimum from a nominal starting solution.

Received Aug. 17, 1997; revision received April 15, 1998; accepted for publication April 15, 1998.

*Professor, Mechanical Engineering, Aeronautical Engineering, and Mechanics. Associate Fellow AIAA.

The problems just identified do not disappear by simply reverting to nongradient methods for optimal search. Indeed, some problems, such as higher computational costs, are further compounded in adopting these techniques. Nevertheless, these search methods have received increased attention, primarily because of their lack of dependence on gradient information, a more robust approach to handling discrete and integer design variables, and for an enhanced ability to locate the globally optimal solution.

Exhaustive enumeration and random search methods are among the simplest and most robust nongradient strategies that can be used in searching for an optimal design. These methods are applicable to a broad class of problems, and are not limited by discontinuous design spaces as are the majority of gradient-based techniques derived from mathematical programming principles. A similar statement can also be made about a multitude of zero-order methods in mathematical programming such as the Nelder–Mead simplex method,[6] complex method of Box,[7] the Hooke–Jeeves pattern search,[8] Rosenbrock's method,[9] and Powell's conjugate direction method.[10] However, as stated earlier, there is usually a stiff computational requirement associated with the use of these methods, particularly as the design problem grows in size. The issue of computational costs is also a concern in methods such as random walk and random walk with direction exploitation, which represent an improvement over simple enumerative techniques. Recently, emerging computational paradigms that have their philosophical basis in physical processes such as natural evolution and laws of thermodynamics, use random sampling as a tool to guide a highly exploitative search process. These methods have emerged as potent tools for locating optimal designs in problems that are generally regarded as difficult. The computational requirement associated with the use of these methods in problems with increasing dimensionality continues to be excessive, necessitating their use in conjunction with function approximations.

While the issue of handling design problems with a mix of continuous, discrete, and integer design variables, nonconvex or disjoint design spaces, and unavailability of gradient information, is somewhat addressed through the use of simulated annealing (SA) and genetic algorithm (GA) approaches, there continues to be a need for adapting these techniques to large-

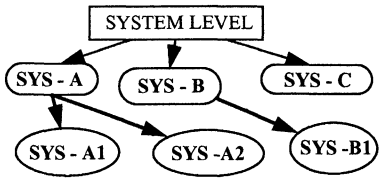

Fig. 1 Top-down hierarchical system.

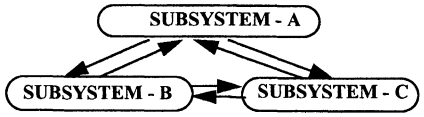

Fig. 2 Nonhierarchical system.

scale problems. There has been some progress in extending GAs to large-dimensionality problems,[11] where all design variables are simultaneously considered. An alternative approach borrowed from the use of standard mathematical programming search algorithms is the use of problem decomposition strategies. In this approach, a large-scale problem is represented by two or more smaller subproblems that are weakly coupled. The form of coupling may be hierarchic in nature, where there is a well-ordered flow of information from one level to another (Fig. 1). More generally, however, such hierarchy is difficult to establish, and the majority of design problems are represented by subproblems with multiple one- and two-way couplings (Fig. 2). While decomposition allows for reduced dimensionality, there are attendant problems with the approach. In addition to identifying a topology for decomposition, there is a need for a method to account for the interactions between the temporarily decoupled subproblems. Given the absence of gradients that are generally used to coordinate subproblem solutions,[12] alternative strategies must be investigated.

The development of design systems like ENGINEOUS,[13] and its more recent form iSIGHT,[14] which are a combination of algorithmic and heuristic procedures, are typical of efforts that have grown out of a recognition of inadequacies of traditional design techniques. Such methods augment traditional search techniques with rule-based expert systems, changing parameters of the algorithm in an opportunistic manner dependent upon the progress of the solution procedure. Such methods do suffer from a problem typical of most rule-based systems, the rule base is defined at the outset and does not have the capacity to learn new rules from the problem domain. Computational paradigms that use GA search procedures to develop machine learning capabilities offer a potential to circumvent some of these problems. Other quasiheuristic methods such as Tabu search have also received recent attention in the design literature.

This paper first presents a very basic overview of zero-order methods of mathematical programming that are applicable to the search phase of the MDO problem. A similar treatment is then presented for techniques that use random sampling as an engine to direct the search process. In this overview, a distinction is made between methods that are useful for local search vs those that are configured to search for globally optimal solutions. The discussion then turns toward those additional requirements that must be met if these methods are to be used in MDO, particularly the issues of function approximations and use in large-scale problems. The subject of response surfaces and artificial neural networks is briefly discussed in this context. Some newly emerging concepts, such as modeling of the biological immune system[15] and machine learning,[16,17] are then discussed, particularly in reference to their implications in obtaining solutions to MDO problems.

Zero-Order Methods for Local Search

This class of methods can be subdivided into deterministic and stochastic methods, and these will be treated in order.

Deterministic Search Methods

Zero-order methods require only function evaluations, and a commonly used approach in this class of methods, applicable to problems with continuous variables, is the Nelder–Mead simplex strategy.[6] The figure obtained by joining a set of $n + 1$ points in an n-dimensional space is referred to as a simplex; in a two-dimensional space, the simplex would be a triangle. Values of the objective function at each of the $n + 1$ points, also referred to as vertices, are first computed. The basic step in the approach is referred to as the reflection step, in which the vertex with the worst objective function value is reflected onto the opposite face, creating a new simplex. This step follows an assumption that such reflection will get the search pointed away from the worst point in a direction with a more favorable objective function value. With a fixed-size simplex, a cycling among vertices may occur, and is typically fixed by expanding or contracting the size of the simplex. This process is repeated until no further improvement in the objective or constraint functions are obtained.

Other commonly used methods in this class of zero-order solution techniques are the complex method, the pattern search of Hookes and Jeeves,[8] the Rosenbrock method,[9] and Powell's method.[10] These methods are predicated on learning about the shape of the design space through a series of exploratory searches, and assist in locating the optimal point in a fixed number of steps. Powell's method of conjugate directions, for example, uses the conjugacy property to select search directions.[10] This property assures that the optimal point of a n-dimensional quadratic function can be located by searching in n or fewer directions that are conjugate to one another. While a detailed explanation of the conjugacy property is beyond the scope of this paper, it suffices to state that two vectors, $\boldsymbol{S}^i$ and $\boldsymbol{S}^j$, are considered conjugate if $(\boldsymbol{S}^i)^T\nabla^2 F(\boldsymbol{S}^j) = 0$, where $\nabla^2 F$ is the Hessian matrix of the function F that is to be minimized. The Hessian matrix is never actually computed in practice. Instead, the approach is to use the n-orthogonal directions that would result from univariate moves in an n-variable problem as the first set of search directions. A conjugate direction is then constructed as

$$\boldsymbol{S}^{n+1} = \sum_{i=1}^{n} \alpha_i^* \boldsymbol{S}^i \tag{1}$$

and an optimal step size in this direction is then computed. This newly constructed search direction then displaces the first search direction $\boldsymbol{S}^i$, and Eq. (1) is used to compute the next conjugate direction. Even though all engineering design problems are not quadratic, a large fraction of these problems can be approximated as quadratic functions close to the optimum. The method of Hooke and Jeeves[8] and the Rosenbrock's method[9] are similar in concept, in that exploratory searches are used to assess the shape of the design space. It uses information derived in this way to accelerate convergence of the search process.

Status and Potential

These methods have been widely used in unconstrained optimization, and work best in problems with relatively few variables. They are, however, local search techniques, and will locate the nearest relative optimum from the starting point. With the exception of the simplex method, the efficiency of other techniques mentioned earlier is dependent on how well the objective function can be represented through a quadratic approximation. The use of these methods in the presence of design constraints is accommodated by converting the constrained problem into a sequence of unconstrained problems,

also referred to in the literature as the penalty function approach. This strategy, however, adds further to the computational costs.

Random Search Methods

Random search methods are generally easy to implement, and are useful in problems involving nonconvex design spaces, discontinuities in gradients, and problems with a mix of continuous, discrete, and integer design variables. A common denominator in each of these methods is the use of random numbers to assist in the exploration of the design space, and a number of different variants may be considered.

Random-Walk

Approach

Consider the unconstrained optimization problem where an objective function $F(X)$ is to be minimized or maximized over the design variables $X \in R^n$, where the components of X given as X_i are restricted to be within prescribed lower and upper bounds $X_i^L \leq X_i \leq X_i^U$. In the random walk approach, the design is modified at the $t+1$ step as $X^{t+1} = X^t + \alpha S^i$, where α is a step-size parameter, and S^i is an established direction of search. The search direction is simply a unit random vector generated at the ith stage, by using a random number generator to create a total of n pseudorandom numbers $r_1, r_2, \ldots, r_n$ between -1 and $+1$. The vector S^i can then be obtained as follows:

$$S^i = \frac{1}{R}\begin{bmatrix} r_1 \\ r_2 \\ \bullet \\ \cdot \\ r_n \end{bmatrix}, \quad \text{where} \quad R = \left[\sum_n (r_i^2)\right]^{1/2} \tag{2}$$

The new design point is evaluated and used to compute the corresponding objective function value. If this new value is lower than the current best (for minimization problems), the current design is replaced with the newly computed design point, and the process is repeated. If, however, a poorer value of the objective function is obtained, a new search direction is computed until a better design is identified, or an arbitrary large number of new search directions have been evaluated without an improvement in the objective function value. In case of the latter, the step size is reduced, and the process is repeated. When an improved design point cannot be identified, even after reducing the step size to a prescribed small value, the process is terminated. A further improvement of the random walk search technique has been demonstrated by requiring that each favorable search direction be exploited to its maximum capacity. In any search direction where the objective improves, a one-dimensional search is conducted to obtain an optimal value of the step size in that direction.

Random search methods are clearly dependent on the random number generation sequence. A pseudorandom number generator is typically used for this purpose, which requires a seed number to be specified by the user. Given this dependence, it would be prudent to run the search for many different values of the random number seed. The reader should recognize, however, that this adds to the computational requirements of the search strategy.

Status and Potential

The random walk method is generally considered to be well developed. It is easy to implement and use for mixed variable problems, but severely limited in application; the approach would be considered untenable for even problems with moderate dimensionality. Also, with the exception of random enumeration methods, the approach still retains some characteristics of a local search.

Methods for Global Search

As in the previous section, it is worthwhile to distinguish between deterministic and stochastic search techniques that feature attributes of global search. Global optimization methods have received considerable attention in the optimization literature, and are generally categorized according to the problem to which they are applicable. Among the most important classes of global optimization problems are differential convex optimization, minmax problems, bilinear and biconvex programming, and quadratic programming problems. In general, there has been greater focus on applying these methods to unconstrained optimization problems or to constrained problems with simple bound constraints; the subject of global optimization with general nonlinear functions and constraints remains a field for further work. A good reference to global optimization methods is available in Ref. 18; however, this source leans heavily toward deterministic methods. Reference 19 provides a collection of real-world applications in global optimization.

Deterministic Global Search

Among the deterministic methods for the solution of unconstrained problems or those with simple bound constraints, partitioning methods have been used with some success. These methods rely upon a priori knowledge of how rapidly the function can vary on the basis of the Lipschitz constant,[19] or on the availability of an analytic formulation of the objective function.[20] The latter are referred to in the literature as interval methods. The basic design of interval methods does not require gradient information; however, the efficiency of these methods can be enhanced. Another deterministic global search strategy that has been widely used is that of sequential quadratic programming.[21] This approach, however, is based on the approximate representation of the objective function as a quadratic function, constrained by linear equality and inequality constraints. The solution guarantees a globally optimal solution, but only to the approximate problem. Hence, the quality of solution is linked to how well the sequence of quadratic programming problems represents the original problem.

The reader is also alerted to strategies for global optimization that are referred to as clustering methods.[22] These can be viewed as a modified form of standard multistart procedures, where local searches are conducted from several points distributed over the search domain. These methods attempt to overcome the principal deficiency of multistart methods, namely repeated identification of the same local minimum, by carefully selecting points at which the local search is initiated. Sample points in the search domain are selected and transformed to group them around the local minima. Clustering methods are then used to identify groups that represent neighborhoods of local optima; redundant searches can be avoided by simply starting from a local point within each cluster. Because many function evaluations are required to identify clusters, the method is numerically expensive. Although the method has been shown to be effective on low-dimensionality problems, most of the development and application work has been in the context of unconstrained optimization problems. Also, although not a requirement, it is generally more efficient to conduct the local searches using methods that are based on gradient information.

Status and Potential

While the field of deterministic global optimization has been very active, the focus has resided in looking at unconstrained optimization problems. The usual methods of converting constrained problems into a sequence of unconstrained problems would be applicable, although at a substantial increase in computational costs. Also, the deterministic methods of global search either require explicit functional forms of the objective and/or constraint functions, or, in many instances, the availability of gradient information. Increased use of these methods

in multidisciplinary design problems will require significant new focus in handling constrained problems, and in strategies that alleviate the required computational costs. The most promise appears to be in developing hybrid methods that combine characteristics of local and global search in an advantageous manner.

Stochastic Global Search

We next focus attention on some recently emerging global optimization methods that have received considerable attention in the design community. These methods use stochastic sampling as a tool to guide highly exploitative search techniques that are considerably more efficient than pure random sampling or random-walk methods.

Simulated Annealing

Approach

Simulated annealing ideas are derived from principles of statistical mechanics. In any physical system, the probability $p_T(\phi)$ that at the given equilibrium temperature T, the system is in a state ϕ is given by the Boltzmann distribution

$$p_T(\phi) = \exp\left[-\frac{E(\phi)}{KT}\right] \Big/ \sum_{\Phi} \exp\left[-\frac{E(\phi)}{KT}\right] \tag{3}$$

where K is the Boltzmann constant, $E(\phi)$ is the energy of the state, and Φ represents all possible states that the system can assume at that temperature. The process of annealing reduces the temperature from some elevated point in a series of controlled steps, so as to attain a minimal energy state for the system. At a given temperature, random variations of the system state are considered. If a state results in a lower energy level, it is immediately accepted. If, however, a higher-energy state results from the variation, it is only accepted with a probability p, defined by the following ratio:

$$p = \frac{p_T(\phi_2)}{p_T(\phi_1)} = \exp\left\{-\left[\frac{E(\phi_2) - E(\phi_1)}{KT}\right]\right\} \tag{4}$$

Here, $p_T(\phi_1)$ and $p_T(\phi_2)$ are probabilities of states ϕ_1 and ϕ_2, respectively, as obtained from the Boltzmann distribution, and state ϕ_2 corresponds to a higher energy level than state ϕ_1. Note that with decreasing temperatures, this probability becomes increasingly lower, indicating that higher-energy level states are more likely to be rejected at lower temperatures.

It is precisely this concept[23] that is central to Kirkpatrick et al.'s[24] proposal to use a simulation of the annealing process for optimization. Consider the objective function in an optimization problem as the energy level corresponding to a particular design, where the design is analogous to the state of the system. For a high value of a parameter T, which is similar to the temperature in Eq. (3), several random variations of the design are considered, and the objective function corresponding to each variation is computed. If a variation results in an improved value of the objective function, it is immediately accepted. If this is not the case, and the objective function value increases, then the probability that this design is accepted is given by Eq. (4). The fact that there is always a finite probability of going through a higher energy state enroute to a final lower energy configuration, is the mechanism that allows for an escape from a relative minimum. A finite number of random variations are considered at a given temperature to allow the attainment of an equilibrium state. The number of random variations at each temperature and the rate at which the temperature is lowered is collectively referred to as an annealing schedule.

The initial value of the temperature T_0, and the rate at which this is lowered, has a profound influence on the performance of the algorithm. This temperature must be set high enough so that at least initially, all proposed states are accepted, and the process is simply a random walk. Typically, T_0 should be set so that the ratio μ of accepted moves to proposed moves should be about 0.95. To determine the number of random design variations at each temperature, a simple approach would be to abandon further random trials when the objective function value does not change for a prescribed number M of design changes. Finally, the rate at which the temperature must be lowered in the simulated annealing process is important; in practice, it turns out that the very slow annealing schedule is often unnecessary. However, this is highly problem dependent, and some amount of numerical experimentation with a particular class of problems is generally recommended.

The approach has been studied extensively, both for its mathematical basis[25-28] and in a number of design applications.[29-37] Recent applications of significance in the multidisciplinary design context have been in image restoration and reconstruction in medical diagnostics,[38] and in airfoil design.[39] The latter deals with the use of SA algorithms, coupled with computational fluid dynamics based analysis, in designing airfoils for drag minimization at subsonic and supersonic speeds, as well as in alleviating the intensity of the sonic boom.

Status and Potential

The SA approach has been well-tried and tested in the design community. The scope and size of design problems has been relatively modest, and additional effort is required in this regard. The approach is readily applicable to problems with discrete and integer design variables. However, it is primarily set up for handling unconstrained function optimization. Extensions to constrained problems are usually done through the penalty function formulation. A number of variations of the basic SA approach are being considered, including SAs with multistart strategies, which advocates multiple shorter runs of the SA algorithm as an approach of enhancing efficiency and global search capability. Similar research efforts are being directed at implementing SAs in a parallel computing environment.

Genetic Algorithms

Approach

This algorithm is motivated by the hypothesized natural process of evolution in biological populations, where genetic information stored in chromosomal strings evolves over generations to adapt favorably to a static or changing environment. In Holland's[40] original work, GAs were characterized by bit string representations of possible solutions to a given problem, and by transformations used to vary and improve these coded solutions. The algorithm is based on an elitist reproduction strategy, where members of the population that are deemed most fit are selected for reproduction, and are given the opportunity to strengthen the chromosomal makeup of the progeny generation. This approach is facilitated by defining a fitness function or a measure indicating the goodness of a member of the population in a given generation during the evolution process. A good starting point for someone interested in pursuing this field would be the book by Goldberg.[41]

To represent designs as chromosome-like strings, different schemes may be adopted. As an example, the design variable values can be in the decimal form,[42] with a fixed number of significant digits. The decimal numbers can be stacked head-to-tail to create a string that represents the total design. Similarly, integer numbers[43] can be used to represent discrete values of design variables. Alternatively, and perhaps the most widely practiced approach is one where each design variable is converted into its binary equivalent, and thereby mapped into a fixed length string of 0s and 1s. In this case, the numerical precision with which the design variable is represented is determined by the string length. A number of such strings constitute a population of designs, with each design having a corresponding fitness value F_v. This fitness value could be the objective function $F(X)$ for a function maximization problem.

In the case of function minimization, at least two different alternatives can be considered, as follows:

$$F_v = 1/F(X), \quad F_v = F^* - F(X) \tag{5}$$

where F^* is an arbitrary large value that is greater than the largest $F(X)$ in the population. For constrained optimal design problems, an exterior penalty function formulation can be adapted to transform a constrained optimization problem into an unconstrained one.

Once a chromosome-like representation of designs is available, simulations of genetic evolution and adaptation can be invoked. An artificial gene-transformation mechanism is composed of three principal components, namely, selection, crossover, and mutation.

Selection. The selection process is one that biases the search toward producing more fit members in the population and eliminating the less fit ones. One simplistic approach to selecting members from an initial population to participate in the reproduction is to assign each member a probability of selection on the basis of its fitness. If $F_v(i)$ is the fitness measure of the ith member, it can be assigned a probability of selection

$$p_i = F_v(i) \Big/ \sum_{NPOP} F_v(i) \tag{6}$$

where *NPOP* is the population size. A new population pool of the same size as the original is created, but has a higher average fitness value. Note that no new designs are created in this process; the less fit ones are simply eliminated and additional copies of the more fit designs are brought into the population.

Crossover. The crossover process allows for an exchange of design characteristics among members of the population pool with the intent of improving the fitness of the next generation. While there are a number of different ways in which the crossover operation can be implemented, a widely practiced approach is the two-point crossover. In this approach, two mating parents are selected at random; the random number generator is invoked to identify two sites on the strings, and the strings of 0s and 1s enclosed between the chosen sites are swapped between the mating strings. An illustration of the crossover transformation for two mating strings, each with 20 bits is as follows.

Parent1 = 11001<u>0</u>0100<u>1</u>0011010

Parent2 = 01011<u>1</u>0001<u>1</u>1001000

Child1 = 110011000110011010

Child2 = 010110010011001000

The crossover sites on the parent strings are identified by the underlined bits. A probability of crossover p_c is defined to determine if crossover should be implemented for the chosen pair of mating strings. An approach for implementing crossover when using real number strings is described in Ref. 42.

Mutation. Mutation safeguards the genetic search process from a premature loss of valuable genetic information during reproduction and crossover. The idea behind mutation is simply to choose a few members from the population pool according to a probability of mutation p_m, and to switch a 0 to 1 or vice versa at a randomly selected mutation site on the chosen string.

Evolutionary programming (EP) and evolutionary strategies (ES) are two other procedures that have been motivated by natural evolution.[44,45] These methods, like the GA, use transformation operators such as selection, recombination (crossover), and mutation. However, the degree to which each operator is emphasized in the algorithm is where the principal differences can be found.

In EP, evolution of populations is done primarily through selection and mutation; recombination is not directly implemented although the contention is that the mutation operation is flexible enough to achieve the characteristics of recombination. All N members of the population are considered as parents, and mutated to produce N children. Using a probabilistic survival scheme, the best N of the $2N$ members then go over to the next generation. Real valued numbers are often used to represent the designs, and the mutation is normally distributed with a zero expectation.

The ES approach was developed and most strongly practiced in Europe. In this approach, a population of designs is evaluated and a uniform random selection done to select the parents. In a standard recombinative ES, pairs of parents produce children via recombination, which are then further perturbed through mutation. The number of children may be greater than N, and either the N best children are selected to replace all parents, or the best N out of the children and parents retained as the next generation of parents. The emphasis is again given to mutation operation, by adaptively selecting the mutation rate for each variable that is normally distributed with a zero expectation. Note, however, that recombination is included, and this is in contrast to the EP. Design representations, however, are based on real numbers like in the EP approach, and averaging recombination is introduced to produce children with different characteristics than the parents.

Status and Potential

These methods are well entrenched in the research literature, and have been successfully applied to a number of combinatorial optimization problems. Further progress will hinge upon adaptation of the strategies in large-scale problems, and on more robust approaches to treatment of constrained optimization problems. Additional comments on these aspects, primarily in the context of the GA, are presented in the following section.

Enhancements to GA

The basic GA approach as described here is merely the tip of the iceberg. In fields of computer and system science, GAs have been actively researched. The mathematical basis of the approach has been explored in a number of publications.[46–49] The focus in this paper is to highlight those modifications to the basic GA approach have been proposed as a means to ease the applicability of the method to MDO problems. These include direct schemes by which to account for design constraints,[50,51] increased efficiency in the presence of long chromosomal strings,[11] applications in multicriterion design,[52,53] and adaptation into a decomposition-based design strategy.[54,55] We examine each of these issues in subsequent sections of the paper.

Constraint Handling

If the traditional GA formulation is to be used in constrained optimization, the scalar fitness function is typically formulated as a composite of the objective and constraint functions. A review of the literature pertaining to constraint handling in GA is presented in Ref. 50. This study established the sensitivity of the penalty function approach to the choice of user-specified weighting constants, and which resulted in biasing the search toward suboptimal designs. Another strategy to account for design constraints that assumed that the constraints were linear and that the design space was convex, was explored in Ref. 51.

An alternative to the penalty function approach is based on strategies that adapt useful features of the feasible designs into the infeasible population. The process of adaptation is through the use of an expression operator, which like the crossover and mutation operations in genetic search, is probabilistic in nature. Feasible and infeasible design strings are brought together as a diploid pair, and the expressed gene at each site results in

an expressed chromosome that replaces the infeasible design. A similar process of adaptation is also at work in another strategy that is based on immune network simulation.[15] Both of these schemes, which enhance the effectiveness of constraint handling in GAs, can be described as belonging to a category of gene-repair methods,[56] wherein some faulty genes that contribute to constraint violation are replaced with minimal change in the objective function value.

GAs in Multicriterion Design

In a number of MDO design problems, the statement of the optimization problem calls for allocation of resources in a manner that satisfies multiple, and sometimes conflicting criteria. A commonly adopted approach is to treat one of the multiple criteria as the scalar objective function for the problem, and to formulate appropriate design constraints to accommodate the requirements on the other criteria. While the apparent simplicity afforded by this approach is quite attractive, affective arguments can be made against its use. At a philosophical level, one can always contend that there is a natural separation of criteria and constraints in any design problem. Additionally, when one formulates criteria as constraints, the ability to learn about the extent of the feasible set is seriously compromised. In other words, a multiple criterion approach offers a solution in which a tradeoff pattern emerges, indicating how improvement in any one criterion would adversely affect another criterion. Finally, it is also known that the treatment of criterion as constraints does not yield the same optimal design as when solving the multicriterion problem.

While a number of gradient-based search algorithms have been developed to address the multicriterion problem, our interest here is in nongradient methods applicable to problems with a mix of continuous, discrete, and integer design variables. The GA approach has been adapted in this problem in some recent studies. In Ref. 52, the ability of GAs to simultaneously discover multiple relative optima through the sharing function approach,[57] and an approach referred to as vector-evaluated GA (VEGA[58]), are exploited to solve multicriterion structural design problems. Another adaptation of the GA in multicriterion problems is available in Ref. 59. The GA-based simulation of the biological immune system has also been used in the solution of the multicriterion design problem.[59]

GAs in Large-Scale Problems

It is most critical to recognize that GAs essentially search for an optimal design among a discrete set of design alternatives. Even continuous variables are treated as discrete variables with the precision of representation given by the binary string length. Long string lengths obtained by concatenating the string representations of a large number of design variables, and/or considering a very fine precision in design variable representation imply a larger number of design alternatives, which, in turn, requires that the population size be proportionately increased.

A multistage search,[11] wherein the granularity of the GA-based search is varied through a successive increase in the precision with which a design space is represented, provides one alternative to working with large population sizes. In this approach, a relatively smaller population is first used to identify promising regions of the design space; the design space itself is represented in a coarse granular manner in this early stage. Once the promising regions of the search space are identified, a biased search with higher precision of design-space representation is conducted within those regions. Similar ideas have been studied under the banner of dynamic parameter encoding.[60]

Another approach that assigns significance to the previous generations of evolution in genetic search is referred to as directed crossover.[11] In theory, if a binary string of length L is used for representing the design, a population size that is in proportion to the string length would have to be selected. If however, only a smaller fraction, $isig*L$ of the bits in the string were really significant to the search process, the population size could be reduced accordingly. The primary motivation behind the directed crossover strategy, therefore, is to identify significant bit position on the string, and to constrain the crossover operation to these bit locations. The process is initiated in the usual manner with random selection of crossover sites on the mating strings, and with no preference allocated to any particular site. After the crossover operation, the fitness change of each mating pair is recorded. Use these fitness changes, a crossover gain is assigned to each bit involved in the crossover. The crossover gains are accumulated over a few initial cycles of evolution, and then used to perform two-point or multipoint crossover operations.

Finally, it is worthwhile to point out that preliminary studies have indicated a role for immune network modeling as an approach to enhance the convergence characteristics of GAs in large-scale design problems.[61]

Parallel GAs

The implementation of GAs in parallel has been the subject of considerable research. At the very outset, it is important to make the distinction between use of multiprocessor computers to make the analysis involved in GAs more efficient, and the use of such computers to give new slant to the GA procedure itself. The latter, termed parallel genetic algorithm (PGA)[62,63] has two major differences from the traditional GA.

1) The population is divided into a number of sectors, and mating is restricted to members in a particular sector; minimal migration of members from one sector to another is permitted. Also, as a result of some overlap among the sectors, information flows out to the whole population through a diffusion process.

2) Each individual may improve its fitness during its lifetime, e.g., by local hill-climbing. After such local hill-climbing, it mates with an individual in its vicinity, with the offspring being subjected to further hill-climbing. The child may then replace the parent, depending on fitness value.

The primary difference between PGA and GA is that the former uses few intelligent and active members in the search as opposed to the latter that uses more, but passive, members to conduct the search. In fact, the power of the PGA stems from a combination of the processing speed of parallel hardware and software speed on the inherent parallelism available in the GA.

While the mechanics of implementing these search strategies are simple, there are some key differences from traditional methods to which a few of the strengths may be attributed. First, these methods work on function evaluations alone and do not require function gradients; gradients may direct the search toward a local optimum. The methods are easily applied to problems where the design space consists of a mix of continuous, discrete, and integer variables. Finally, it is important to recognize that methods inspired by natural evolution proceed from several points in the design space to another such set of design points. Because the information for this update is drawn from several points distributed through the design space, these methods have a better probability of locating global minima as opposed to those schemes that conduct a point-to-point search.

The usefulness of stochastic search methods in MDO problems is severely limited without the use of global function approximations. Given that these methods are primarily based on the use of function information only, the use of response surface-based approximations is a viable option. The use of these response surfaces, however, requires that the order of the surface be first specified. Neural network-based function approximations have also been extensively explored in this context, and are briefly discussed in what follows.

Function Approximations—Response Surfaces and Neural Networks

Response surfaces are obtained by fitting a chosen-order polynomial model to given experimental or numeric data. When building a multiple regression or response surface, care must be exercised in determining which terms participate in the regression model. For example, with two variables x_1 and x_2, a quadratic response variation y can be written as follows:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_4 x_1^2 + \beta_5 x_2^2 + \beta_6 x_1 x_2 \quad (7)$$

where β are the constants to be determined using the given data. It is conceivable that not all terms in Eq. (7) may contribute to modeling the actual response. If these terms are arbitrarily forced into the regression, the generalization capabilities of the regression would be compromised; a number of techniques are available for identifying the terms that should be included to obtain an optimal fit.

The data used in the regression are obtained by sampling the design space at a predetermined number of points. The collection of data by designed experiments that places emphasis on satisfactory distribution over the domain of interest, allows for designs of increasing order to be built sequentially, provides an internal estimate of error, and requires the least number of experimental runs, has been proposed.[64,65] The response surface approach has been used in a number of optimization studies.[66,67] Several references on the subject are available in Ref. 68, which itself deals with the subject of response surface use in aircraft design. References 69 and 70 detail applications of the response surface approach in multidisciplinary design of the high-speed civil transport. The principal drawback of using the approach is that the user must specify the order of the fit. Further, as problem dimensionality increases, response surface models are imprecise and very difficult to generate.

We next turn to the use of another global-approximation paradigm, namely, the artificial neural network. While a detailed description of neural networks is beyond the scope of this paper, some general introductory statements are in order. As shown in Fig. 3, the network architecture consists of a layer of artificial neurons to which the external stimuli are presented, a series of hidden layers of artificial neurons, and a layer of neurons at which the output is available. The input neurons do not process the input stimulus; they simply serve as fan-out points for connections to neurons in successive layers. Neurons in each layer are connected to all neurons in adjacent layers; there is an interconnection weight associated with this connection that defines the strength of the connection. Also associated with each artificial neuron is what is referred to as an activation function (sigmoid function or step function). The weighted sum of all inputs to a particular neuron are processed through this nonlinear activation function to produce a neuron output, which then feeds into all neurons of the next layer.

The presence of the hidden layer, and the nonlinear activation functions, enhance the ability of the networks to learn nonlinear relationships between the presented input and output quantities. This learning or training in these networks simply requires the determination of all interconnection weights of the network and characteristics of all activation functions, so that the network accurately produces the desired output for each of the input patterns used in the training. Once such a trained network is established, it responds to a new input within the domain of its training by producing an estimate of the output response. To this extent, it serves as a function approximation tool that provides inexpensive function information in the stochastic sampling-based search procedures.[71] The trained weights of this network can also be used to identify dependencies among design variables and design objectives/constraints.[16,72] This weight analysis may be considered as a smeared global sensitivity over the range of network training, and identifies the importance of any input component on an output quantity of interest. Such an analysis can be used to systematically partition the design space in a decomposition-based design approach. The use of GAs in such a strategy is explored in a later section.

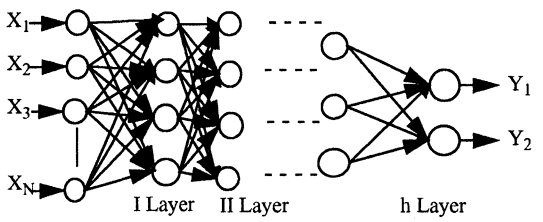

Fig. 3 Back propagation network.

The use of neural networks may be considered as a response surface approach, where the order of the polynomial fitting function does not have to be specified. In fact, the neural network is a special form of response surface where the response function is a nested squashing function; the interconnection weights correspond to the regression parameters in a typical regression model. The polynomial regression models are good for linear mappings, but difficult, if not impractical for large dimensionality, nonlinear mappings. If, for example, a matrix inversion approach is used to solve for the regression parameters, then with *NPAR* parameters, the modeling capacity would be approximately proportional to the square root of the memory size. The neural network approach has no such restriction, and the modeling capacity would be proportional to the actual memory size. In other words, if a polynomial model using inverse methods is used to derive a mapping, it could afford at most *NPAR* parameters, whereas the back propagation neural network would allow the use of (*NPAR* × *NPAR*) parameters.

GAs in Decomposition-Based Design

An alternative approach to adapting GA search for large-scale design problems is based on partitioning the problem into an appropriate number of subproblems; a reasonable approach for partitioning is one where balanced subsets of design variables are assigned to different subproblems, and where each subproblem would be responsible for meeting the system-level design objectives and for satisfying constraints most critically affected by the design variables of that subproblem. A formal manner of partitioning the problem has been presented in Ref. 54, where causality in the design domain is obtained through the use of trained neural networks.

Consider the design problem to be formulated in terms of a design variable vector $\boldsymbol{X}$. Also, let the design constraints $\boldsymbol{g}_j(\boldsymbol{X})$ belong to the global constraint set $\boldsymbol{G}$. The vector $\boldsymbol{X}$ and constraint set $\boldsymbol{G}$ are said to define a system level problem that is formulated as follows:

$$\begin{aligned} &\min \text{ or } \max \quad F(X) \\ &\text{subject to} \quad G \equiv \{g_j(X), j = 1 \ldots NCON\} \leq 0 \end{aligned} \quad (8)$$

Assume further that the best topology for decomposing the problem domain resulted in three subproblems A, B, and C, and the design variables and constraints for each of these subproblems are denoted by X_A, X_B, X_C, and g_A, g_B, and g_C, respectively. The objective function $F(\boldsymbol{X})$ for each of the subproblems is the same, and is the system level objective function. The system level problem of Eq. (8) is now represented by the following three subproblems:

$$\begin{aligned} &\min \text{ or } \max \quad F(X) \\ &\text{subject to} \quad g_i(X_i)0,\ X_k = \text{const} \\ &i, k = A, B, \text{ or } C, \quad i \neq k \end{aligned} \quad (9)$$

The GA strategy in each subproblem works with shorter string lengths, and, hence, smaller population sizes are required in each subproblem. The principal challenge in this approach is that the constraint sets identified for a particular subproblem, are not completely independent of the design variables that may have been assigned to another subproblem. One strategy that allows for subproblem coordination is based on the simulation of a biological immune system.

Immune Network Modeling

In biological immune systems, foreign cells and molecules, denoted as antigens, are recognized and eliminated by type-specific antibodies. This pattern recognition capability is impressive, given that the genome contains about 10^5 genes, and the immune system must use segments of these to construct antibodies for all possible antigens that are likely to be encountered. In biological systems, this recognition problem translates into a complex geometry-matching process. For the process to effectively use the available genetic resources in this pattern-recognition process, there must be considerable organization and cooperation among the gene libraries.

This process can be simulated using the GA approach, and has been the subject of considerable study.[61,73-76] Here, a matching function that measures the similarities between antibodies and antigens substitutes for the fitness function of the GA. Specialist antibodies adapted to specific antigens, or generalists, that provide the best match to a number of different antigens, can be developed in this simulation. The use of binary string structures to represent both the antigens and antibodies provides a seamless integration into the proposed GA-based design optimization. Various levels of complexity can be introduced in developing a numerical measure of the degree of match, including the number of matches on a bit-by-bit basis and the length of contiguously matched strings. A simple numerical measure Z defined as

$$Z = \sum_{i=1}^{N_{\text{string}}} t_i \tag{10}$$

can be used, where N_{string} is the length of strings in the binary representation of design variables, and $t_i = 1$ if there is a match at the ith location of the two strings, and is 0 otherwise (Fig. 4). A larger value of Z indicates a higher degree of match between the two strings over the entire antigens. Using a traditional GA simulation, a population of antibodies can be evolved to cover a specified pool of antigens, with Z used as the fitness measure for this simulation.

In using this approach to account for interactions among temporarily decoupled subproblems, the motivation is to adapt changes in design variables from other subproblems into a particular subproblem with a prescribed frequency in the artificial evolution process. This adaptation must proceed under two guiding principles:

1) Updating the design variables of other subproblems must not simply involve introducing the best design from those subproblems, but rather an average of the best few designs. In this regard, a generalist antibody would be developed that is the best representation of a number of good designs.

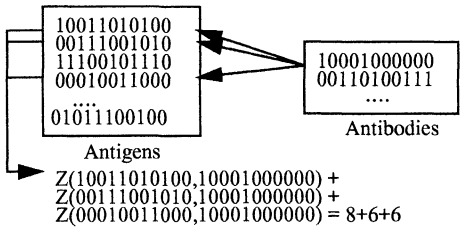

Fig. 4 Matchscore computation.

2) The updating scheme of step 1 would also be influenced by requiring that the difference among the objective function values from the different subproblems be minimized.

While the requirement of step 2 can be explicitly included into the fitness function for the immune network simulation, it is not required, as this requirement is implicitly satisfied by the adaptation process in the first step. In fact, this is a much more cost-effective strategy as it precludes the computation of any objective function other than the matching function in the immune network simulation.

The proposed decomposition-based design procedure using an immune network system may be summarized as follows. The stringlike chromosome structure representing the design contains a definition of all design variables. After partitioning of the design variable vector for each subsystem has been performed, genetic evolution is carried out in each subsystem in parallel, with the fitness function described in terms of the system level objective function. It should be noted that only that subsection of the chromosome string corresponding to the design variables for that particular subsystem is changed. This process can be carried out for a fixed number of generations, and then a predetermined fraction of fit strings from each subsystem introduced as antigens to all other subsystems. An immune system simulation is then performed, and serves as a correction step that introduces compatibility between the different subpopulations, so that the process eventually converges on the desired design. The approach has been used in the multidisciplinary design of rotor blades,[54] and in a dual structure-control optimization problem.[77]

Heuristic Methods in Design

With an increased acceptance of optimization methods in industry, heuristic techniques have assumed importance in the research community. This section provides an overview of heuristic methods that may be considered as an advancement over the traditional rule-based expert systems.

Tabu Search

Approach

Tabu search is an adaptive strategy that was primarily designed for combinatorial problems. The principal attributes of the method is its ability to continue exploration with a hill-descending search algorithm, even in the absence of improvements in the objective function, and its ability to avoid local optima that it may have found but later rejected. It uses short-term memory of recent solutions and strategies to impose move restrictions.

Assume that a cost function $c(x)$ is to be minimized, and x is a member of X that may include constraints on x. A neighborhood $N(x)$ is defined in which an improved solution is searched. In general, the solution is found from a modified neighborhood $MN(x)$, which contains those solution points that are allowed or are non-Tabu. A Tabu list is thus maintained, and while initially, this list would be a subset of the neighborhood points, it is ultimately replaced by a number of previously identified elite solutions. It is this Tabu list that prevents cycling of solutions as well as avoids convergence to local optima. The list is dynamic and incorporates issues related to both short-term, and mid-to-long-term memory. Tabu Search also uses memory to modify the cost function, $c(x)$; the history of moves is therefore used to encourage or penalize emergence of solutions. In problems where the neighborhood is large or costly to examine, a subset of candidates representative of $MN(x)$ is created. Randomization may or may not be used as a search tool.

The Tabu list is not completely binding, and moves on this list are allowed if aspiration–criteria so warrants it. One such aspiration criterion could be the best-solution criterion; here, if a solution on the Tabu list yields a better solution than one encountered before, it is accepted. Other aspiration criteria include aspiration by objectives, aspiration by search direction,

aspiration by influence, etc. A more detailed description of this approach is available in Refs. 78 and 79. Applications of this approach to scheduling problems in manufacturing are available in Refs. 80 and 81.

Status and Potential

Technique is comparatively new and many variations are still being explored. It is somewhat difficult to even find a single accepted implementation of the approach. Like the SA and GA, it provides an approach to overcome the problem of relative optima.

Hybrid Expert-System/Numerical Optimization

In design problems, where different subset of design variables are dominant in different regions of the search space, the problem complexity can be controlled by dynamically selecting and retaining only the dominant variables at any given stage of the search. A rule-based search coupled with numerical optimization has been used in one such problem involving turbine blade design,[82] with the design engineer implementing a static rule base.

Consider a design problem where the objective is to maximize the turbine performance of a three-stage power turbine, using as design variables the stage reactions, and the blade and vane Zwiefel numbers. Constraints included in this problem pertain to upper and lower bounds on turning and swirl at stage exit for each of the three stages. Examples of typical rules directed at the objective function and at constraint satisfaction are as follows:

Goal: Increase Efficiency

Weight: 1.0

Conditions: Current design is in Cluster 1

Actions: Increase Stage-3 Blade-Zwiefel

Decrease Stage-2 Blade-Zwiefel

Decrease Stage-2 Reaction

Increase Stage-1 Blade-Zwiefel

Goal: Increase Stage-3 Swirl

Weight: 1.0

Conditions: Current Design is in Cluster 1

Actions: Decrease Stage-3 Blade-Zwiefel

Decrease Stage-3 Vane-Zwiefel

Increase Stage-2 Blade-Zwiefel

It is clear, however, that rules would have different strengths in different parts of the design domain, and such information can be derived from causal relations extracted from trained neural networks, as described in Ref. 72. Using this data, rules were generated for each cluster, which were then executed in a prioritized manner based on the dominance established through the weight analysis. An example of the modified rule is as follows:

Goal: Increase Efficiency

Weight: 1.0

Conditions: Current Design is in Cluster 1

Actions: Increase Stage-3 Blade-Zwiefel

Decrease Stage-2 Blade-Zwiefel

Increase Stage-3 Reactions

Decrease Stage-1 Blade-Zwiefel

Increase Stage-1 Reaction

This approach was shown to significantly enhance the efficiency of the computational process[15] involved in obtaining an optimized design.

Classifier Systems in Design

A solution strategy that exhibits a significant degree of adaptivity is one based on the use of classifier systems. Classifier systems, first described by Holland,[83] have been studied extensively in the field of computer science.[84,85] Their proposed adaptation in MDO problems is a novel concept, and targets the shortcomings in a traditional rule-based system. Here, a set of rudimentary rules initially defined by the user (could be random) are adaptively changed as the solution progresses. Information from the computational domain is used in this adaptation process.

A classifier system is generally divided into two parts: a set of rules or classifiers, and a message list. The message list contains at least one input from the external environment and also provides the framework for the rules to interact (any rules generated internally are posted here as well). Hence, the message list is dynamic in nature, constantly evolving as the system changes. The classifier rules are made up of three distinct segments: conditions, actions, and strength. To facilitate the use of GAs in this approach, all rules are coded as binary strings. The conditions allow the classifier to read the message list by searching for matches between the condition and the message list. If a match is found, the action is posted to the message list. Some rules have the special role of affecting external action (effectors). The strength is a number associated with each rule designed to indicate its value to the systems, and it forms the basis for learning. If a rule helps bring about useful responses, it gains strength. Similarly, an ineffective rule is weakened and perhaps ultimately purged from the system. The usefulness of a rule is measured by the performance of the system based on external outputs. When a useful output takes place, the responsible rules are rewarded, and their strength is proportionately increased. The strength of the proposed approach would be to introduce new rules into the system based on principles of genetic search—the use of genetic operators to produce new rules is the creative element of the approach.

Applications of this approach to a problem of truss optimization, and determining how data must be distributed in a design domain to generate acceptable function approximations (polynomial response surfaces), are presented in Refs. 16 and 86. Additional studies are also being conducted to examine the applicability of this approach in decomposition-based design, and in the multidisciplinary design of turbine blades.[87]

Conclusions

The paper provides a general overview of nongradient methods in the optimal design of multidisciplinary systems, with a focus on techniques that are broadly described as soft-computing methods. The methods discussed include heuristic strategies, those based on random sampling, or a combination of the two. They have a clear role in handling design optimization problems with nonconvex or disjoint design spaces, and where the design space is a mix of continuous, discrete, or integer design variables. Additional studies are required in making these methods more amenable to large-scale design problems, and in reducing the effort involved in problems where each function evaluation is computationally demanding.

Acknowledgments

The partial support received under DARPA Primary Contract 70NANB6H0074 is gratefully acknowledged. The author would like to thank S. Nagendra of General Electric Corporate Research and Development for his many suggestions.

References

[1]Sobieszczanski-Sobieski, J., "Multidisciplinary Design Optimization: An Emerging New Engineering Discipline," *Advances in Struc-*

tural Optimization, edited by J. Herskovits, Kluwer, Dordrecht, The Netherlands, 1995, pp. 483–496.

[2]Tolson, R. H., and Sobieszczanski-Sobieski, J., "Multidisciplinary Analysis and Synthesis: Needs and Opportunities," AIAA Paper 85-0584, 1985.

[3]Abdi, F., Ide, H., Levine, M., and Austel, L., "The Art of Spacecraft Design: A Multidisciplinary Challenge," *2nd NASA/Air Force Symposium on Recent Advances in Multidisciplinary Analysis and Optimization*, NASA CP-3031, 1988, pp. 1137–1156.

[4]Hajela, P., Bloebaum, C. L., and Sobieszczanski-Sobieski, J., "Application of Global Sensitivity Equations in Multidisciplinary Aircraft Synthesis," *Journal of Aircraft*, Vol. 27, No. 12, 1990, pp. 1002–1010.

[5]Bloebaum, C. L., "Formal and Heuristic Decomposition Methods in Multidisciplinary Design," Ph.D. Dissertation, Univ. of Florida, Gainesville, FL, 1991.

[6]Nelder, J. A., and Mead, R., "A Simplex Method for Function Minimization," *Computer Journal*, Vol. 7, No. 4, 1965, pp. 308–313.

[7]Box, M. J., "A New Method of Constrained Optimization and Comparison with Other Methods," *Computer Journal*, Vol. 8, No. 1, 1965, pp. 42–52.

[8]Hooke, R., and Jeeves, T. A., "Direct Search Solution of Numerical and Statistical Problems," *Journal of the Association of Computing Machinery*, Vol. 8, 1961, pp. 212–229.

[9]Rosenbrock, H. H., "An Automated Method for Finding the Greatest or Least Value of a Function," *Computer Journal*, Vol. 3, No. 4, 1960, pp. 175–184.

[10]Powell, M. J. D., "An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives," *Computer Journal*, Vol. 7, No. 4, 1964, pp. 303–307.

[11]Lin, C.-Y., and Hajela, P., "Genetic Search Strategies in Large Scale Optimization," *Proceedings of the AIAA/ASME/ASCE/AHS/ASC 34th Structures, Structural Dynamics, and Materials Conference* (La Jolla, CA), AIAA, Washington, DC, 1993, pp. 2437–2447.

[12]Sobieszczanski-Sobieski, J., "A Linear Decomposition Method for Large Optimization Problems—Blueprint for Development," NASA TM 83248, 1982.

[13]Powell, D. J., Skolnick, M. M., and Tong, S. S., "Interdigitation: A Hybrid Technique for Engineering Design Optimization Employing Genetic Algorithms, Expert Systems, and Numerical Optimization," *Handbook of Genetic Algorithsm*, edited by L. Davis, Van Nostrand Reinhold, New York, 1991, pp. 312–331.

[14]*iSIGHT Developer's Manual*, Engineous Software, Inc., Morrisville, NC, 1997.

[15]Hajela, P., and Lee, J., "Constrained Genetic Search Via Schema Adaptation: An Immune Network Solution," *Structural Optimization*, Vol. 12, No. 1, 1996, pp. 11–15.

[16]Goel, S., and Hajela, P., "Classifier Systems in Design Optimization," AIAA Paper 97-1572, 1997.

[17]Goel, S., and Hajela, P., "Identification of Parameter Coupling in Turbine Design Using Neural Networks," *Journal of Propulsion and Power*, Vol. 12, No. 3, 1966, pp. 503–508.

[18]Horst, R., and Pardalos, P. M. (eds.), *Handbook of Global Optimization*, Kluwer, Dordrecht, The Netherlands, 1995.

[19]Pinter, J. D., *Global Optimization in Action*, Kluwer, Dordrecht, The Netherlands, 1995.

[20]Hansen, E., *Global Optimization Using Interval Analysis*, Marcel Dekker, New York, 1992.

[21]Biggs, M. C., "Constrained Minimization Using Recursive Quadratic Programming: Some Alternative Subproblem Formulations," *Towards Global Optimization*, edited by L. C. W. Dixon and G. P. Szego, North–Holland, New York, 1975, pp. 341–349.

[22]Torn, A., and Zilinskas, A., *Global Optimization, Lecture Notes in Computer Science*, No. 350, Springer–Verlag, Heidelberg, Germany, 1989.

[23]Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E., "Equations of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, Vol. 21, No. 6, 1953, pp. 1087–1091.

[24]Kirkpatrick, S., Gellat, C. D., Jr., and Vecchi, M. P., "Optimization by Simulated Annealing," *Science*, Vol. 220, No. 4598, 1983, 671–680.

[25]Lundy, M., and Mees, A., "Convergence of an Annealing Algorithm," *Mathematical Programming*, Vol. 34, No. 1, 1986, pp. 111–124.

[26]Johnson, D. S., Aragon, C. R., McGeoch, L. A., and Schevon, C., "Optimization by Simulated Annealing: An Experimental Evaluation. Part I. Graph Partitioning," *Operations Research*, Vol. 37, No. 99, 1990, pp. 865–893.

[27]Siarry, P., Berthiau, G., Durdin, F., and Haussy, J., "Enhanced Simulated Annealing for Globally Minimizing Functions of Many-Continuous Variables," *ACM Transactions on Mathematical Software*, Vol. 23, No. 2, 1997, pp. 209–228.

[28]Aarts, E., and Korst, J., *Simulated Annealing and Boltzmann Machine—A Stochastic Approach to Combinatorial Optimization and Neural Computing*, Wiley, New York, 1989.

[29]Chen, G.-S., Bruno, R. J., and Salama, M., "Optimal Placement of Active/Passive Members in Truss Structures Using Simulated Annealing," *AIAA Journal*, Vol. 29, No. 8, 1991, pp. 1327–1334.

[30]Kincaid, R. K., and Padula, S. L., "Minimizing Distortion and Internal Forces in Truss Structures by Simulated Annealing," *Proceedings of the AIAA/ASME/ASCE/AHS/ASC 32nd Structures, Structural Dynamics, and Materials Conference* (Long Beach, CA), AIAA, Washington, DC, 1990, pp. 327–333.

[31]Balling, R. J., "Optimal Steel Frame Design by Simulated Annealing," *Journal of Structural Engineering*, Vol. 117, No. 6, 1991, pp. 1780–1795.

[32]Szewczyk, Z., and Hajela, P., "Neural Network Approximations in a Simulated Annealing Based Optimal Structural Design," *Journal of Structural Optimization*, Vol. 5, No. 1, 1993, pp. 159–165.

[33]Ogot, M. M., and Alag, S., "An Effective Mixed Annealing/Heuristic Algorithm for Kinematic Mechanical Design," *Journal of Mechanical Design* (to be published).

[34]Ogot, M. M., and Gilmore, B. J., "A Kinematic Based Approach for the Robust Design of Mechanical Assemblies," *Journal of Engineering Design and Automation* (to be published).

[35]Inhwan, H., Gilmore, B. J., and Ogot, M. M., "The Incorporation of Arc Boundaries and Stick/Slip Friction in a Rule-Based Simulation Algorithm for Dynamic Mechanical Systems with Changing Topologies," *Journal of Mechanical Design*, Vol. 115, No. 3, 1993, 423–434.

[36]Lee, S. J., Gilmore, B. J., and Ogot, M. M., "Dimensional Tolerance Allocation of Stochastic Dynamic Mechanical Systems Through Performance and Sensitivity Analysis," *Journal of Mechanical Design*, Vol. 115, No. 3, 1993, pp. 392–402.

[37]Yip, P. P. C., and Pao, Y. H., "A Guided Evolutionary Simulated Annealing Approach to the Quadratic Assignment Problem," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 24, No. 9, 1994, pp. 1385–1388.

[38]Geman, S., and Geman, D., "Stochastic Relaxation, Gibbs Distributions and Bayesian Restoration of Images IEEE Transactions on Pattern Analysis and Machine Intelligence," PAMI-6, 1984, pp. 721–741.

[39]Aly, S., Marconi, F., Ogot, M., Pelz, R., and Siclari, M., "Stochastic Optimization Applied to CFD Shape Design," AIAA Paper 95-1647, June 1995.

[40]Holland, J. H., *Adaptation in Natural and Artificial Systems*, Univ. of Michigan Press, Ann Arbor, MI, 1974.

[41]Goldberg, D. E., "Genetic Algorithms in Search, Optimization, and Machine Learning," Addison–Wesley, Reading, MA, 1989.

[42]LeRiche, R., "Optimization of Composite Structures by Genetic Algorithms," Ph.D. Dissertation, Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, Oct. 1994.

[43]LeRiche, R., and Haftka, R. T., "Optimization of Laminate Stacking Sequence for Buckling Load Maximization by Genetic Algorithms," *AIAA Journal*, Vol. 31, No. 5, 1993, pp. 951–956.

[44]Box, G. E. P., "Evolutionary Operation: A Method of Increasing Industrial Productivity," *Applied Statistics*, Vol. 6, No. 2, 1957, pp. 81–101.

[45]Fogel, L. J., Owens, A. J., and Walsh, M. J., *Artificial Intelligence Through Simulated Evolution*, Wiley, New York, 1966.

[46]Goldberg, D. E., and Segrest, P., "Finite Markov Chain Analysis of Genetic Algorithms," *Proceedings of the 2nd International Conference on Genetic Algorithms*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987, pp. 1–8.

[47]Goldberg, D. E., "Sizing Populations for Serial and Parallel Genetic Algorithms," *Genetic Algorithms and Their Applications: Proceeding of the 3rd International Conference on Genetic Algorithms*, 1989, pp. 70–79.

[48]DeJong, K. A., "Analysis of the Behavior of a Class of Genetic Adaptive Systems," Ph.D. Dissertation, Univ. of Michigan, Ann Arbor, MI, 1975.

[49]DeJong K. A., "Adaptive System Design: A Genetic Approach," *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-10 (9), 1980, pp. 566–574.

[50]Hajela, P., and Yoo, J., "Constraint Handling in Genetic Search Using Expression Strategies," *AIAA Journal*, Vol. 34, No. 11, 1996, pp. 2414–2420.

[51]Michalewicz, Z., and Janikow, C. Z., "Handling Constraints in

Genetic Algorithms,'' *Proceedings of the 4th International Conference on Genetic Algorithms*, Morgan Kauffman, Palo Alto, CA, 1991, pp. 151–157.

[52]Hajela, P., and Lin, C.-Y., ''Genetic Search Strategies in Multicriterion Optimal Design,'' *Structural Optimization*, Vol. 4, No. 2, 1992, pp. 99–107.

[53]Osyczka, A., and Kundu, S., ''A New Method to Solve Multicriterion Optimization Problems Using the Genetic Algorithm,'' *Structural Optimization*, Vol. 10, No. 2, 1995, pp. 94–99.

[54]Lee, J., and Hajela, P., ''GA's in Decomposition Based Design—Subsystem Interactions Through Immune Network Simulation,'' *Proceedings of the AIAA/NASA/USAF/ISSMO 6th Conference on Multidisciplinary Analysis and Optimization* (Bellevue, WA), AIAA, Washington, DC, 1996, pp. 1717–1726.

[55]Lee, J., and Hajela, P., ''Parallel Genetic Algorithm Implementation in Multidisciplinary Rotor Blade Design,'' *Journal of Aircraft*, Vol. 33, No. 5, 1996, pp. 962–969.

[56]Back, T., *Evolutionary Algorithms in Theory and Practice*, Oxford Univ. Press, Oxford, England, UK, 1996.

[57]Goldberg, D. E., and Richardson, J., ''Genetic Algorithms with Sharing for Multimodal Function Optimization,'' *Genetic Algorithms and Their Applications: Proceedings of the 2nd International Conference on Genetic Algorithms*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987, pp. 41–49.

[58]Schaffer, J. D., ''Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms,'' Ph.D. Dissertation, Vanderbilt Univ., Nashville, TN, 1984.

[59]Yoo, J., and Hajela, P., ''Immune Network Modeling in Multicriterion Design of Structural Systems,'' AIAA Paper 98-1911, 1998.

[60]Schraudolph, N. N., and Belew, R. K., ''Dynamic Parameter Encoding for Genetic Algorithms,'' *Machine Learning*, Vol. 9, No. 1, 1992, pp. 9–21.

[61]Hajela, P., Yoo, J., and Lee, J., ''GA Based Simulation of Immune Networks—Applications in Structural Optimization,'' *Journal of Engineering Optimization*, Vol. 29, No. 1–4, 1997, pp. 131–149.

[62]Manderick, B., and Spiessens, P., ''Fine-Grained Parallel Genetic Algorithm,'' *Proceedings of the 3rd International Conference on Genetic Algorithms*, edited by H. Schaffer, Morgan–Kauffmann, Palo Alto, CA, 1989, pp. 428–433.

[63]Muhlenbeim, H., Schomisch, M., and Born, J., ''The Parallel Genetic Algorithm as a Function Optimizer,'' *Parallel Computing*, Vol. 17, No. 627, 1991, pp. 619–632.

[64]Peterson, R., *Design and Analysis of Experiments*, Marcel Dekker, New York, 1985.

[65]Box, G., and Drapper, N., *Empirical Model-Building and Response Surfaces*, Wiley, New York, 1987.

[66]Venter, G., Hajtka, R. T., and Starner, J., Jr., ''Construction of Response Surfaces for Design Optimization Applications,'' *Proceedings of the AIAA/NASA/USAF/ISSMO 6th Conference on Multidisciplinary Analysis and Optimization* (Bellevue, WA), AIAA, Reston, VA, 1996, pp. 548–564.

[67]Wang, B., ''A New Method for Dual Response Surface Optimization,'' *Proceedings of the AIAA/NASA/USAF/ISSMO 6th Conference on Multidisciplinary Analysis and Optimization* (Bellevue, WA), AIAA, Reston, VA, 1996, pp. 1805–1814.

[68]Giunta, A. A., ''Aircraft Multidisciplinary Design Optimization Using Design of Experiments Theory and Response Surface Modeling,'' Ph.D. Dissertation, Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, May 1997.

[69]Giunta, A. A., Narducci, R., Burgee, S., Grossman, B., Mason, W. H., Watson, L. T., and Haftka, R. T., ''Variable-Complexity Response Surface Aerodynamic Design of an HSCT Wing,'' *Proceedings of the AIAA 13th Applied Aerodynamics Conference* (San Diego, CA), AIAA, Washington, DC, 1995, pp. 994–1002.

[70]Crisafulli, P., et al., ''Response Surface Approximations for Pitching Moment Including Pitch-Up in the MDO Design on an HSCT,'' *Proceedings of the AIAA/NASA/USAF 6th Multidisciplinary Analysis and Optimization Symposium* (Bellevue, WA), AIAA, Reston, VA, 1996, pp. 1308–1322.

[71]Hajela, P., and Lee, J., ''Role of Emergent Computing Techniques in Multidisciplinary Design,'' *Proceedings of the NATO Advanced Research Workshop on Emergent Computing Methods in Engineering Design* (Nafplio, Greece), edited by D. Grierson and P. Hajela, Springer–Verlag, Berlin, 1996, pp. 162–187.

[72]Hajela, P., and Szewczyk, Z., ''Neurocomputing Strategies in Structural Design—On Analyzing Weights of Feedforward Neural Networks,'' *Structural Optimization*, Vol. 8, No. 4, 1994, pp. 236–241.

[73]Smith, R. E., Forrest, S., and Perelson, A. S., ''Searching for Diverse Cooperative Populations with Genetic Algorithms,'' Dept. of Computer Science, Univ. of New Mexico, TR CS92-3, Albuquerque, NM, 1992.

[74]Dasgupta, D., and Forrest, S., ''Tool Breakage Detection in Milling Opeartions Using a Negative Selection Algorithm,'' Dept. of Computer Science, Univ. of New Mexico, TR CS95-5, Albuquerque, NM, 1995.

[75]Farmer, J. D., Packard, N. H., and Perelson, A. S., ''The Immune System, Adaptation, and Machine Learning,'' *Physica D*, Vol. 22, 1986, pp. 187–204.

[76]Varela, F., Coutinho, A., Dupire, B., and Vaz, N. M., ''Cognitive Networks: Immune, Neural, and Otherwise,'' *Theoretical Immunology, Part Two*, edited by A. S. Perelson, Addison–Wesley, Reading, MA, 1988.

[77]Lee, J., and Hajela, P., ''GA's in Decomposition Based Design—Subsystem Interactions Through Immune Network Simulation,'' *Structural Optimization*, Vol. 14, No. 4, 1997, pp. 248–255.

[78]Glover, F., ''Tabu Search—Part I,'' *ORSA Journal on Computing*, Vol. 1, No. 3, 1989, pp. 190–260.

[79]Glover, F., ''Tabu Search—Part II, *ORSA Journal on Computing*, Vol. 2, No. 1, 1990, pp. 4–32.

[80]Dell'Amico, A. M., and Trubian, A. M., ''Applying Tabu Search to the Job-Shop Scheduling Problem,'' *Annals of Operations Research*, Vol. 41, No. 1/4, 1993, pp. 231–252.

[81]Porto, A. S. C. S., and Ribeiro, A. C. C., ''A Tabu Search Approach to Task Scheduling on Heterogeneous Processors Under Precedence Constraints,'' *International Journal of High-Speed Computing*, Vol. 7, No. 2, 1995.

[82]Goel, S., Gregory, B., and Cherry, D., ''Knowledge-Based System for Preliminary Aerodynamic Design of Aircraft Engine Turbines,'' SPIE Applications of Artificial Intelligence: Knowledge-Based Systems in Aerospace and Industry, Orlando, FL, April 1993.

[83]Holland, J. H., ''Outline for a Logical Theory of Adaptive Systems,'' *Journal of the Association of Computing Machinery*, Vol. 9, 1962, pp. 297–314.

[84]Holland, J. H., ''Properties of the Bucket Brigade Algorithm,'' *Proceedings of the 1st International Conference on Genetic Algorithms and their Applications*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1985, pp. 1–7.

[85]Forrest, S., ''Implementing Semantic Network of Structures Using the Classifier System,'' *Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1985, pp. 80–92.

[86]Hajela, P., and Lee, J., ''A Machine Learning Based Approach to Improve Global Function Approximations for Design Optimization,'' *Proceedings of the International Symposium on Optimization and Innovative Design Japan Society of Mechanical Enginners*, 1997.

[87]Goel, S., ''Machine Learning Paradigms in Design Optimization—Applications in Turbine Aerodynamic Design,'' Ph.D Dissertation, Rensselaer Polytechnic Inst., Troy, NY, Feb. 1998.