Pakistan Journal of Applied Sciences 1(3): 289-294, 2001

Genetic Algorithms in Optimization and Computer Aided Design

Reza Farshadnia
Department of Mathematics, University of Kashan, Kashan, Iran

Abstract: Developments in computational models of evolutionary processes have led to the realization of powerful, robust and general optimization and adaptive systems collectively called evolutionary algorithms. In this paper, we consider one member of this class of algorithms, the genetic algorithm and describe the features and characteristics that are particularly apprcpriate for applications in control systems engineering. The versatility and robust qualities of the algorithm are considered and a number of application areas described. Some prospective future directions are also identified.

Key words: Evolutionary algorithms, genetic algorithms, optimization, adaptive control

Introduction

There has been widespread interest from the control community in applying genetic algorithms (GAs) Holland (1992) to problems in control systems engineering. Based on computational models of natural biological evolution, GAs are members of the class of evolutionary algorithms Spears *et al.* (1993) that also includes evolutionary programming Fogel *et al.*, (1966), evolution strategies Rechenberg (1973) and genetic programming Koza (1991). Compared to traditional search and optimization methods, such as calculus-based and enumerative strategies, the GA is robust global and generally more straightforward to apply in situations where there is little or no a priori knowledge about the process to be controlled. For the control engineer GAs and evolutionary algorithms in general, present opportunities to address some classes of problems that are not. amenable to efficient solution through the application of conventional techniques. In recent years, GAs have been applied to a broad range of activities in control systems engineering, including combinatorial and parametric optimization, robust control analysis, multiobjective design, process scheduling and adaptive control.

Searching from a population of solution estimates, the GA is an adaptive and robust search method, which in control systems engineering may be used as an optimization tool or as the basis of a more general adaptive or learning system. Starting with an introduction to the basic GA, this paper considers some of the variations that are possible from the simple model. Important characteristics, relevant to applications in control engineering, are then identified, including the use of parallelism. A range of recent control applications are surveyed and the benefits, or disadvantages, of using the GA are considered. Finally, the paper speculates on likely promising areas for future developments.

Genetic Algorithms: Genetic algorithms are stochastic global search methods that mimic the metaphor of natural biological evolution. GAs operate on a population of potential solution estimates, individuals, applying the principle of survival of the fittest to produce better and better approximations to a solution. At each epoch, or generation, of the algorithm, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals from which they were created, just as occurs in natural adaptation.

Fig. 1 shows an outline of the simple GA described by Goldberg (1989). Individuals are encoded as strings, chromosome, composed over some particular alphabet(s) so that the genotypes (chromosome values) are uniquely mapped onto the decision variables, phenotypes. The first step of a GA is therefore to initialize the population by randomly sampling the chromosome space. Once this has been accomplished, the initial population may be evaluated by converting the chromosomes to their phenotypic values and evaluating some objective function. The objective function can be seen as characterizing an individual's performance in the problem domain and is used in the selection phase as the basis for defining the relative fitness of an individual.

```
procedure GA {
        t = 0;
      initialize P(t);
     evaluate P(t);
     while not finished do {
            t = t + 1;
            select P(t) from P(t-l );
            reproduce pairs in P(t);
            mutate P(t);
            evaluate P(t);
            }
                  }
```

Fig. 1.: A simple genetic algorithm

During the reproduction phase, each individual is assigned a fitness value derived from its raw performance measure, given by the objective function, and this is used to bias the selection process. Highly fit individuals, relative to the whole population, have a high probability of being selected for reproduction and less fit individuals a correspondingly.

Lower probability of being selected. The selected individuals are then modified through the application of genetic operators to produce the next generation. Genetic operators manipulate the characters (genes) of the chromosomes directly, using the assumption that certain individuals' gene codes, on average, produce fitter individuals. Genetic operators may be divided into two main categories:

Recombination: Exchanges genetic information between pairs, or larger groups, of individuals in order to create new chromosomes. It is not necessarily applied to all groups of individuals selected for reproduction, allowing some chromosomes to survive into successive

generations unaltered.

Mutation: Causes individuals' genetic representations to be altered according to some probabilistic rule. Generally, mutation is applied with a much lower probability than recombination and is usually viewed as a background operator that ensures that the probability of searching any particular subspace of the search space is never zero.

After recombination and mutation, the individual strings are then decoded and evaluated by the objective function and so the process continues through subsequent generations. In this way, the average performance of individuals in a population is expected to increase as fit individuals are preserved and reproduced with one another and the less fit individuals die out. The GA is terminated when some criterion is satisfied, for example, a certain number of generations is reached, or a mean deviation in the average performance of the population, or when some point in the search space is encountered.

Many variations of the simple GA have been developed. Selection algorithms have been introduced that reduce the stochastic errors associated with roulette wheel methods, bounding the actual number of offspring that an individual produces to the expected number derived from its relative fitness measure Baker (1987). Ranking methods Whitley (1989) have been introduced as an alternative to proportional fitness, which reduces the bias of the selection algorithm to highly fit individuals and help avoid premature convergence. Variations of recombination operators have been developed, such as multiple point and reduced surrogate crossover Booker (1987) for binary-coded GAs and intermediate and line recombination Muhlenbein and Voosen (1993) for real-valued representations. Further parameterization of the GA has been introduced through the introduction of generation gaps and variable recombination and mutation rates. In the simple GA populations are non-overlapping, which is known not to be the case in biological systems. The concept of a generation gap establishes how many individuals are replaced in a population at each generation. If only a single offspring is produced at each generation then the GA is said to be steady-state Whitley (1989). Parameterization of the recombination and mutation rates allows the amount of disruption during reproduction to be controlled with generations, without introducing a bias towards the length of representation used Smitendorf *et al.* (1992).

Genetic Algorithm Issues for Control: As the GA does not require derivative information or a formal initial estimate of the solution region and because of the stochastic nature of the search mechanism, GAs are capable of searching the entire solution space with more likelihood of finding the global optimum than conventional optimization methods. Indeed, conventional methods usually require the objective function to be well behaved, whereas the generational nature of GAs can tolerate noisy and discontinuous function evaluations.

A number of considerations commonly arising in control engineering problems and the way in which these are treated through the application of GAs, are discussed below.

Representation: Continuous decision variables may be handled either directly through real-valued representations and the appropriate genetic operators or by using binary representation schemes and standard genetic operators. In the case of binary representations, real values can be approximated to the necessary degree with a fixed point binary scheme. In most control problems, however, it is the relative precision of the parameters that is significant rather than absolute precision. In such cases, the logarithm of the parameter may be encoded, reducing the number of bits and hence memory usage Smitendorf *et al.* (1992). Alternatively, a direct floating point representation may be used. Discrete decision variables can be encoded using either binary or n-ary representation. When functions can be expected to be locally monotonic with respect to such variables, the use of Gray coding is known to better exploit such monotonicity Caruana and Schaffer (1988). This consideration also applies to binary representations of continuous decision variables. In cases where a mixture of discrete and continuous decision variables is to be used, it is feasible to use a mixed representation provided care is taken to ensure that the genetic operators used function correctly over the set of encoding chosen. However, encoding all of the parameters using a single binary representation can simplify the operation of the GA.

Scale: The concept of fitness is central to all evolutionary algorithms. Given that many optimization problems are characterized by a real-valued objective function, these values must be converted into a nonnegative fitness value if they are to be handled correctly by the GA. Early work on GAs concentrated on the use of offsetting objective function values so that selection could be based directly on an individual's performance within a population Goldberg (1989). The use of scaling retains an individual's relative performance and also attempts to bias the selective pressure towards better individuals while still allowing relatively unfit individuals the potential to reproduce. Alternatively, by discarding the relative differences between individual's raw performance and considering them only according to their ranking in a population, a constant selective pressure may be applied throughout the evolutionary process Whitley (1989). Off setting and scaling can result in more and more individuals receiving fitnesses with relatively small differences as the population converges. The rank-based methods maintain a constant selective pressure towards good individuals throughout convergence and are claimed to bring a number of other advantages, such as computational efficiency and reduced bias. Additionally, rank-based schemes offer a convenient mechanism for considering multiple objectives simultaneously; this is discussed in the next section.

Constraints: Most control engineering problems are subject to constraints. For example, actuator have finite limits on their operation and control loops are required to be stable. GAs can handle constraints in a number of ways. The most efficient and direct method is to embed these constraints in the coding of the individuals. Where this is not feasible, penalty functions may be used to ensure that invalid individuals have fitness levels that reflect that they are low performers. However, appropriate penalty functions are not always easy to design for a given problem and may effect the efficiency of the search Richardson *et al.* (1989). An alternative approach is to consider constraints as design objectives and recast the problem as a multiobjective one (see, for example, Fonseca and Fleming (1995).

Adaptation: The vast majority of applications of GAs in control have concentrated on their use as function optimizers. However, GAs have been shown to be well suited to tracking time-varying systems Dasgupta and

McGregor (1992), that is, systems in which the optimum fitness or fitness criterion changes over time. Such changes typically occur as a result of a change in the external environment (e.g., a change in operating conditions) or of system changes (e.g., wear of mechanical components). The GA has the advantage over many conventional methods of being able to respond to such changes by exploiting the diversity of individuals in the current population and evolving towards the new performance measure. If there is insufficient diversity in the population, then new material can be readily introduced by randomly reinitializing some members of the population.

Software: Although there exist many good public-domain genetic algorithm packages, such as GENESYS Grefenstette (1990) and GENITOR Whitley (1989), none of these provide an environment that is immediately compatible with existing tools in the control domain. The MATLAB Genetic Algorithm Toolbox aims to make GAs accessible to the control engineer within the framework of an existing CACSD package. This allows the retention of existing modeling and simulation tools for building objective functions and allows the user to make direct comparisons between genetic methods and traditional procedures. By building GAs into a standard CACSD package, GAs can be made available to control engineers as a powerful tool to complement those already in use. The Neural Works Professional II/Plus neural network software from NeuralWare Inc. now comes with a genetic reinforcement learning system that augments the standard training procedures using an EA to avoid getting stuck in local optima. It can be expected that many CACSD and CAE packages will be supplied with GA tools as a standard feature in the near future.

Parallelism: Apart from the obvious benefits of speedup in execution time, parallel GAs have a higher degree of robustness and, typically, require fewer function evaluations to reach optimal solutions than a comparable, so- called, panmictic GA. In addition, parallel GAs may be made fault tolerant by exploiting the process replication inherent in parallel implementations Goldberg (1989). Employing a distributed population structure with local selection and reproduction and some form of genetic mobility may enhance the performance of the GA over one where the population is treated globally. These benefits may be realized even when the GA is implemented on a sequential machine. A full discussion of parallel GAs can be found in Chipperfield and Fleming (1994).

Control Applications of Evolutionary Algorithms: The application of GAs to control engineering can be broadly classified into two main areas: offline design and analysis and online adaptation and tuning. In offline applications, the GA can be employed as a search and optimization engine, for example to select suitable control laws for a known plant to satisfy given performance criteria or to search for optimal parameter settings for a particular controller structure. In online adaptation GAs may be used as a learning mechanism to identify characteristics of unknown or non-stationary systems or for adaptive controller tuning for known or unknown plants.

In the remainder of this section, some recent applications of GAs in control are considered where conventional methods have been found unsuitable, problematic, or unavailable.

Control System Design: One of the most common applications of GAs is that of parametric optimization. In control system design, many tasks can be cast within an optimization framework. In particular, in the design of control systems through parameter optimization, the final solution can be sensitive to the initial solution estimate or there may be a number of combinations of parameter settings that produce the desired control action. Conventional optimizers may therefore find only suboptimal solutions as the search is forced to consider only small regions of the search space influenced by the initial estimate. GAs are capable of sampling the entire solution space and can therefore be expected to produce solutions that are more global in nature. Additionally, evolutionary approaches have the potential to find solutions in many different areas of the search space simultaneously. Thus more information regarding the nature of the design problem may be elicited during the search, potentially yielding a more informed design process.

GAs have been shown to be an effective strategy in the off-line design of control systems through parametric optimization by a number of practitioners. Krishnakumar and Goldberg (1992) and Bramlette and Cousin (1989) have demonstrated how genetic optimizers can be used to derive superior controller structures in aerospace applications in less time (in terms of function evaluations) than other methods such as LQR and Powell's gain set design. Varsek *et al.* (1993) have shown how GAs may be used in the selection and tuning of controller structures, and in robotics Gleghocn *et al.* (1988) have demonstrated how GAs may be used for path-planning problems in both stationary and non-stationary environments

Multiobjective Optimization: Problems in control engineering very seldom require the optimization of a single objective function. Instead there are usually a number of competing design objectives that are required to be satisfied simultaneously.

Conventionally, members of the Pareto-optimal solution set are sought through solution of an appropriately formulated nonlinear programming problem. A number of approaches are currently employed, including the ε - constraint, weighted sum, and goal attainment methods Hwang and Masud (1979). However, such approaches require precise expression of a usually not well understood set of weights and goals. If the trade-off surface between the design objectives is to be better understood, repeated application of such methods will be necessary. In addition, nonlinear programming methods cannot handle multimodality and discontinuities in function space well and can thus only be expected to produce local solutions. Multiobjective GAs Fonseca and Fleming (1994) evolve a population of solution estimates, thereby conferring an immediate benefit over conventional MO methods. Using rank-based selection and Niching techniques, it is feasible to generate populations of non-dominated solution estimates without directly combining objectives in some way. This is advantageous because the combination of non-commensurate objectives requires precise understanding of the interplay between those objectives if the optimization is to be meaningful. The use of rank-based fitness assignment permits different non-dominated individuals to be sampled at the same rate, thereby according the same preference to all Pareto- optimal solutions. GAs have the potential to become a powerful method for multiobjective optimization. Including the control engineer in the design process as a decision

maker, the GA may be guided, through the progressive articulation of preferences, to particular areas of interest in the search space. The trade-offs between design criteria and their interactions can be examined closely and the engineer's knowledge and experience can be employed to make informed decisions on the basis of design requirements rather than the properties of the objective functions.

Robust Control: One approach to the design of robust control systems is through eigenstructure assignment. Patton and Liu (1994) have demonstrated a hybrid approach combining GAs and gradient-based optimization. Their scheme has been applied to the design of an aircraft lateral control system and employs a real-valued representation in the minimization of a cost function based on a combination of the sensitivity and complementary sensitivity functions of the closed-loop system. In the evaluation stage of the GA, each individual is improved by one step of the DFP algorithm, and the resulting individual is evaluated according to the cost function. The remainder of the GA operates in the usual manner for a real-valued population. It is claimed that a GA approach takes full advantage of the freedom provided by the eigenstructure assignment to find a stabilizing controller that minimizes the performance index. In another approach, Dakev *et al.* (1995) employ a GA in the loop-shaping design procedure to find suitable weighting functions for a robust MIMO controller for a critical system. The GA employs a structured representation Dasgupta and McGregor (1992) that allows a wide range of weighting functions to be searched and suitable parameters to be identified simultaneously. The design problem considered was an EMS suspension system for a magnetically levitated vehicle and was based on finding suitable weighting functions to shape the open-loop transfer functions while satisfying the $H \infty$ optimization of a normalized co-prime factorization of the nominal plant description and explicit closed-loop performance criteria. Hill-climbing techniques Whidborne *et al.* (1994) had previously been proposed as a potential solution for such "mixed optimization" problems; however, the GA-based approach was found to be advantageous if the weighting space was large and beyond the coverage of conventional approaches. In particular, it should be noted that the GA-based approach has the potential to find controllers satisfying $H \infty$ criteria of lower order than those found using conventional approaches.

System Identification: Many problems in control-engineering, signal processing, and machine learning can be cast as a system identification problem where the task is to determine a suitable model from a given set of input-output data. The resulting model can then be used for the prediction and control of a "black-box" system. Although there exist many tried and tested methods for linear system identification, in practice most real-world control systems are, to some extent, nonlinear. The extra complexity associated with nonlinear system identification, particularly when there is no initial information or model structure detail, has to some degree contributed to the relative lack of attention to this particular field. One successful approach to this problem is the NARMAX method Chen *et al.* (1989) to find a suitable set of nonlinear terms for the system in question. However, the complexity and the combinatorial growth in the search space means that exhaustive search is not always feasible.

Fonseca *et al.* (1993) have implemented nonlinear system identification using the NARMAX approach and GA- based subset selection Lucasius and Kateman (1992). The GA is used to select a fixed number of terms from a set of possible nonlinear terms, and the NARMAX method is used to identify the parameters of those terms. Because the GA operates on a population of solution estimates, a family of low-variance models is produced that can be assessed according to different criteria before a final model is chosen. Compared with the conventional approaches, which search the term space iteratively, building a more and more complex model, the GA-based approach conducts a global and robust search of the model space. Thus, the GA has the potential to be more effective in identifying a suitable model structure, and hence more general in nature.

The genetic identification strategy may be made more effective by allowing the search to be conducted over a variable number of terms. Rather than using a minimum descriptor-length-based objective function, we can cast the problem as a multiple-objective one, thereby considering the number of terms and the variance simultaneously. This approach is preferable, as the impact of the number of terms in the model will not obscure the residual performance of any model directly. To accommodate a population with individuals consisting of a variable number of terms, a structured representation may be appropriate Dasgupta and McGregor (1992). Using a hierarchical chromosome representation, individual genes can be used to turn on or off specific terms in a model formulation. This approach has been applied with some success to the related problem of FIR filter design by Roberts and Wade Robert and Wade (1993). In a similar vein, the group method of data handling allows the representation of terms as a tree structure. Sub-trees can then be exchanged among individuals during reproduction without the need to re-evaluate that part of the tree. Iba *et al.* (1993) have demonstrated how this approach may be applied to nonlinear systems identification by considering time-series prediction and pattern-matching problems.

System Integration: In addition to the problem of designing controllers to provide optimal performance, the controllers must also be integrated into the whole system. For example, in gas turbine engine controller implementations, besides control algorithm issues there are other important hardware realization and implementation concerns. These include design considerations such as the placement, weight, and interface types for transducers and actuators and the design for redundancy, fault tolerance, and fault diagnosis policies. All of these issues bear on the optimization of the integration, and it is desirable to consider how they interact with each other as a single design process. A typical modern gas turbine engine has over 160 interfaces combined with a mixture of data busses and dedicated wiring harnesses to convey discrete and continuous data signals. Placement of components is dictated by the environment, electronics cooling requirements, reliability and safety considerations, and a number of other criteria. The goal of systems integration is therefore to consider the integration of the control system with the plant and interface requirements with other interacting systems, such as the aircraft, to lead to improvements in safety, efficiency, weight, reliability, and operating costs.

The systems integration problem is common to many industrial applications and can be posed within a

multiobjective framework. However, it has generally been found to have properties that render it unsuitable for conventional optimization techniques. For example, the decision variables are a mixture of discrete and continuous parameters and the design objectives cover a broad range of measures from controller performance to mechanical considerations. Additionally, the objective functions can, in general, be expected to be highly nonlinear and exhibit a high degree of interaction with one another. The use of genetic based methods has the potential to search such complex objective function surfaces, incorporating the designer's knowledge in both the formulation and the solution of the problem (see, for example, Parmee and Denham (1994).

Real-Time and Adaptive Control: Two major problems have been encountered in attempts to use GAs in real-time control applications: generational execution time and ensuring the production of satisfactory control laws at each generation. The problem of generational execution time may be addressed by either parallel GAs, incremental GAs Whitley (1989), or micro GAs Karr (1991). Incremental GAs produce only one or two offspring at each generation, and therefore have the advantages of reduced generation cycle time and memory requirements, but may not produce new or satisfactory individuals at each generation. Micro GAs, employing a very small population, have the potential to produce more offspring but suffer from a lack of genetic diversity and tend to result in a more local search. The task of producing a satisfactory control law at each generation is the harder problem to address. For example, a relatively insensitive controller may have a large number of suitable parameter settings that result in a satisfactory control law. Thus, it is possible that the GA could produce successive control laws that resulted in unacceptably large changes to the controller settings, possibly introducing stability problems. Nonetheless, a number of successful schemes based on GAs have been developed. Porter and Jones ((1992) have developed a genetic approach to tuning digital PID controllers that is claimed to be much simpler to implement than previously applied constrained optimization techniques, even for the case of simple plants. Other adaptive approaches have used GAs to implement classifier systems that learn production rules online and provide performance measures for these rules according to how well they control the plant, Jong (1980). In these GA classifier systems, individuals are a coding of a set of production rules that represent the control action to be applied to the plant under given circumstances. A related approach to the online use of GAs is that of fuzzy control. Karr (1991 and 1992) demonstrates how GAs may be used to design both adaptive and non-adaptive fuzzy logic controllers for a dynamic system. In this approach, the GA is used to optimize the membership functions of the controller. It is argued that the control rules tend to remain constant, even across a wide range of conditions, and that the membership functions should be adapted to the present situation. However, Linkens and Nyongesa (1992) see the optimization of membership functions as an offline task, and a fuzzy classifier system is used to acquire and modify good sets of rules online. In their scheme, adaptive control is provided by the constant adaptation of the rule set to meet the changing dynamics of the problem, without an explicit identification of a plant model.

Conclusion

This study has presented a broad overview of GAs in control systems engineering, describing many of the features and characteristics of GAs that are important and relevant in the implementation of these algorithms for control engineering applications. In addition, a number of specific application areas have been considered in some detail. Clearly, GAs have become an established search and optimization procedure in control engineering and elsewhere.

Offline implementations are a particularly rewarding application area for GAs. In control systems design, and multiobjective optimization in particular, GAs offer the opportunity to address problems that were hitherto not susceptible to efficient solution by conventional methods. Such problems may be addressed directly through GAs and, by including the control engineer in the optimization process as a decision maker, afford the opportunity for the designer to guide the search while learning about the problems' trade-offs. This consideration is particularly important when the objective functions are not well understood or behave poorly.

The ability of the GA to solve complex combinatorial optimization problems efficiently is also an important consideration. For example, in systems integration this allows the control engineer to consider such issues as sensor/ actuator placement and connectivity requirements in the same process as the controller design. Future developments in the field of decision support and concurrent engineering can expect to benefit from such advances in GA applications.

Online applications of GAs present a number of significant challenges that will need to be addressed before their use becomes widespread. When it is possible to identify a suitable system model, a conventional GA can be applied in some adaptive or tuning capacity in a relatively straightforward manner. However, very few systems can be expected to exhibit the high degree of robustness required for direct manipulation by an GA. Instead, methods are beginning to emerge that either limit the control action that the GA is capable of, or bound the scope of the GA's search while maintaining an acceptable level of performance. In the future, learning systems, which represent an important growth area in control, can expect to receive more attention. Such systems, including artificial neural networks, classifier systems, and adaptive fuzzy controllers, may be coupled with GAs to expedite their learning. Future developments in these areas are likely to involve GAs as a central component.

References

A.J. Chipperfield and P.J. Fleming, 1994. Parallel genetic algorithms, forthcoming in Handbook of parallel and distributed computing, ed. A.Y. Zomaya Chipperfield, P.J. Fleming, and, C.M. Fonseca, Genetic algorithm tools for control systems engineering, Proc.1andt Int. Conf. Adaptive Computing in Engineering Design and Control, Plymouth Engineering Design Centre, UK, 128-133.

A. Roberts and G. Wade, 1993. A structured GA for FIR filter design, Proc.IEE/IEEE Workshop on Natural Algorithms in Signal Processing, 16/1-16/8.

A. Varsek, T. Urbacic and B. Filipic, 1993. Genetic algorithms in controller design and tuning, IEEE Trans. Sys. Man and Cyber.,23: 1330-1339.

B. Porter and A.H. Jones, 1992. Genetic tuning of digital PID controllers, Electronics Letters, 2.8: 843-844.

C.L. Karr, 1991. Design of an adaptive fuzzy logic controller using a genetic algorithm, Proc. 4th Int. Conf. Genetic Algorithm 450-451.

C.-L. Hwang and A.S.M. Masud, 1979. Multiple objective decision making: Methods and application.: A state of the art survey (Berlin, Germany: Springer-Verlag).

C.M. Fonseca and P.J. Fleming, 1994. Multi bjective optimal controller design with genetic algorithms, Proc. IEE Control '94: 745-749.

C.M. Fonseca and P.J. Fleming, 1995. An overview of evolutionary algorithms in multiobjective optimization, Evolutionary Computing, 3: 1-16.

C. L. Karr , 1992. An adaptive system for process control using genetic algorithms, Int. Symp. on Artificial Intelligence in Real-Time Control, IFAC/ IFIP /IMACS preprints, Delft, Netherlands, 585-590.

C.M. Fonseca, E.M. Mendes, P.J. Fleming, and S.A. Billings, 1993. Non-linear model term selection with genetic algorithms, Proc.IEE/IEEE Workshop on Natural Algorithms in Signal Processing, 27/1-27/8.

C.B. Lucasius and G. Kateman, 1992. Towards solving subset selection problems with the aid of genetic algorithms, in Parallel problem solving from nature 2, ed. R. Manner and B. Manderick, (Amsterdam, Holland: Elsevier Science Publishers, 239-247.

D.A. Linkens and H.O. Nyongesa, 1992, A real time genetic algorithm for fuzzy control, IEE Colloquium on Genetic Algorithms for Control Systems Engineering, Digest No. 1992/106.

D. Dasgupta and D.R. McGregor, 1992. Nonstationary function optimization using the structured genetic algorithm, Parallel Problem Solving from Nature t., ed. R. ManDer and B. Manderick, (Amsterdam, Holland: Elsevier Science Publishers,) 145-154.

D.E. Goldberg,, 1989. Genetic algorithm. in search, optimization and machine learning (Reading, MA: Addison-Wesley)

D. Whitley, 1989 The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best, Proc. 3rd Int. Conf. Genetic Algorithm., 116-121.

H. Iba, T. Kurita, H. de Garis and T. Sato, 1993. System identification using structured genetic algorithms, Proc. 5th Int. Conf. Genetic Algorithm., 219-286.

H. Muhlenbein and D. Schlierkamp- Voosen, 1993. Predictive models for the breeder genetic algorithm: I. Continuous parameter optimization, Evolutionary Computation, 1: 25-49.

I.C. Parmee and M.J. Denham, 1994. The integration of adaptive seard1 teclmiqus with current engineering design practice, Proc. 1st Int. Conf. Adaptive Computing in Engineering Design and Control, Plymouth Engineering Design Centre, UK, 1-13.

I.R. Koza, 1991. Evolving a computer program to generate random numbers using the genetic programming paradigm, Proc. 4th Int. Conf. on Genetic Algorithm., Morgan Kaufmann, La lolla, CA, 37-44.

I. E. Baker, 1987. Reducing bias and inefficiency in the selection algorithm, Proc. 2nd Int. Conf. on Genetic Algorithm., 100-101.

J. J Holland, 1992. Adaptation in natural and artificial system. (Cambridge, MA: MIT Press, (2nd Ed.).

J.F. Whidborne, I. Postlethwaite and D.-W. Gu, 1994. Robust controller design using H loop-shaping and the method of inequalities, IEEE Trans. on Cont. Syst. Technology, 9: 455-461.

J.T. Richardson, M.R. Palmer, G. Liepins and M. Hilliard, 1989. Some guidelines for genetic algorithms with penalty functions, Proc.3rd Int. Conf. Genetic Algorithm., 191-197.

J.J. Grefenstette, 1990. A user's guide to GENESIS Version 5.0, Technical Report, Navy Centre for Applied Research in Artificial Intelligence, Washington D.C., USA.

K. De Jong, 1980. Adaptive system design: A genetic approach, IEEE Trans. Syst. Man and Cyber., SMC-JO9: 566-514.

K. Krishnakumar and D.E. Goldberg, 1992. Control system optimization using genetic algorithms, J. Guidance, Control and Dynamic., 15: 735-740.

L. J. Fogel, A.J. Owens and M.J. Walsh, 1966. Artificial intelligence through simulated evolution (N. Y: Wiley).

L. Rechenberg,. 1973. Evolution strategy: Optimierung Technischer System nach Prinzipien der Biologischen Evolution (Stuttgart: Frommann-Holzboog,).

L. Booker, 1987. Improving seard1 in genetic algorithms, in Genetic algorithm. and simulated annealing, ed. L. Davis, (San Mateo, CA: Morgan Kaufmann), 61-73.

M.F. Bramlette and R. Cusin, 1989. A comparative evaluation of search methods applied to parametric design of aircraft, Proc. 3rd Int. Conf. on Genetic Algorithms, 213-218.

N .V. Dakev, J .F .Whidborne and A.J. Chipperfield, 1995. H 00 design of an EMS control system for a Maglev vehicle using evolutionary algorithms, Proc. 1st IEE/IEEE Int. Conf. on Genetic Algorithm. in Engineering System.: Innovation. and Application and, Sheffield, UK, 226-231.

R.J. Patton and G.P. Liu, 1994. Robust control design via Eigenstructure assignment, genetic algorithms and gradient-based optimization, IEE Proc.-Control Theory Appl., 141.

R.A. Caruana and I.D. Schaffer, 1988. Represmtation and hidden bias: Gray vs. binary coding, Proc. 6th Int. Conf. Machine Learnin9: 153-161.

S. Chen, S.A. Billings and W. Luo, 1989. Orthogonal least squares methods and their application to non-linear system identification, Int. J. Control, 50: 1873-1896.

T.F. Gleghocn, P.T. Baffes and L. Wang, 1988. Robot path planning using a genetic algorithm, Proc. 2nd Annual Workshop on Space Operation., Automation and Robotics, 383-390.

W .M. Spears, K.A. De jong, T. Back, D.B. Fogel and H.de Garis, 1993. An overview of evolutionary computation, Machine learning: ECML-93 European conf. on machine learning, lecture note. in artificial intelligence, .667: 442-459.

W .M. Spears and K.A. De long, 1991. On the virtues of parameterised uniform crossover, Proc. 4th Int. Conf. on Genetic Algorithm., 230-236.

W .E. Smitendorf, O. Shaw, R. Benson and S. Forrest, 1992. Using genetic algorithm. for controller design: Simultaneous stabilization and Eigenvalue placement in a region, Tedmical Report No. CS92-9, Deptt. Computer Sci., College Engg., Univ. New Mexico, USA.