GENETIC ALGORITHMS IN CONTROL SYSTEMS ENGINEERING

P. J. Fleming and **R. C. Purshouse**
Department of Automatic Control and Systems Engineering, University of Sheffield, UK

Keywords: Genetic algorithms, control systems engineering, evolutionary computing, genetic programming, multiobjective optimization, computer-aided design, controller design, robust control, H-infinity control system design, linear quadratic Gaussian control, linear time-invariant (LTI) system, model-based predictive control, PID controller, system identification, fault diagnosis, system reliability, stability analysis, robotics, structural control, scheduling, optimization, hybrid control, neural control, fuzzy control, intelligent systems.

Contents

Summary

Genetic algorithms (GAs) are global, parallel, stochastic search methods, founded on Darwinian evolutionary principles. Many variations exist, including genetic programming and multiobjective algorithms. During the 1990s GAs have been applied in a variety of areas, with varying degrees of success within each. A significant contribution has been made within control systems engineering. GAs exhibit considerable robustness in problem domains that are not conducive to formal, rigorous, classical analysis. They are not limited by typical control problem attributes such as ill-behaved objective functions, the existence of constraints, and variations in the nature of control variables. Genetic algorithm software tools are available, but there is no 'industry standard'. The computational load associated with the application of the genetic algorithm has proved to be the chief impediment to real-time application of the technique. Hence, the majority of applications that use GAs are, by nature, off-line. GAs have been used to optimize both structure and parameter values for both controllers and plant models. They have also been applied to fault diagnosis, stability analysis, robot path-planning and combinatorial problems (such as scheduling and bin-packing). Hybrid approaches have proved popular, with GAs being integrated in fuzzy logic and neural computing schemes. The GA has been used as the population-based engine for multiobjective optimizers. Multiple, Pareto-optimal, solutions can be represented simultaneously. In such schemes, a decision-maker can lead the direction of future search. Interesting future developments are anticipated in on-line applications and multiobjective search and decision-making.

1. Introduction

The *genetic algorithm* (GA) has arisen from a desire to model the biological processes of natural selection and population genetics, with the original aim of designing autonomous learning and decision-making systems. Since its introduction, and subsequent popularization, the GA has been frequently utilized as an alternative optimization tool to conventional methods. The correctness of the GA as an abstraction of natural evolution has been challenged but this issue should not be of undue concern to the engineer, who is using the GA for its robust search and optimization properties.

Several analogous algorithms have been proposed in the literature, such as *evolution strategies* (ES) and *evolutionary programming* (EP). These, together with GAs, have been classified under the umbrella group of *evolutionary algorithms* (EAs).

This article describes how the genetic algorithm methodology can be applied to problems in control systems engineering. The suitability of the GA towards various types of problem is discussed, and methods for incorporating the characteristics of control problems, such as constraints on actuator performance, are outlined.

The application of GAs to control can broadly be classified into two distinct areas: off-line design and on-line optimization. Off-line applications have proved to be the most popular and successful. On-line applications tend to be quite rare because of the difficulties associated with using a GA in real-time and directly influencing the operation of the system. GAs have been applied to controller design and to system identification. In each case, either the parameters or the structure can be optimized, or – potentially – both. Other applications include fault diagnosis, stability analysis, sensor-

actuator placement, and other combinatorial problems. This article considers examples from the literature for each class of problem.

The article concludes by offering future perspectives on the direction of EA research, with particular attention to issues of concern to the control engineer.

2. What are Genetic Algorithms?

2.1. Overview

Genetic algorithms (GAs) are global, parallel, search and optimization methods, founded on Darwinian principles. They work with a population of potential solutions to a problem. Each individual within the population represents a particular solution to the problem, generally expressed in some form of genetic code. The population is evolved, over generations, to produce better solutions to the problem. A schematic of the algorithm is shown in Figure 1.

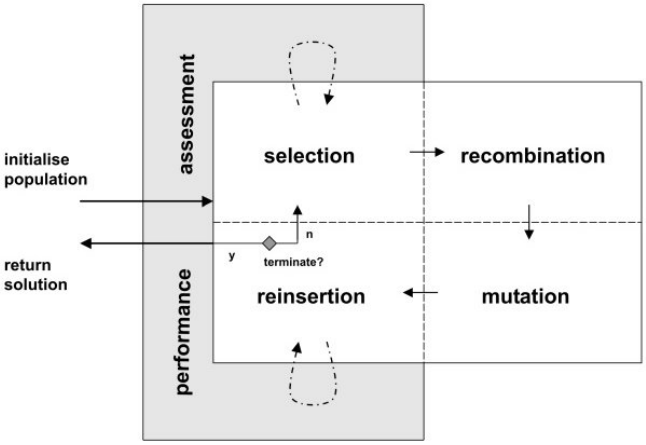

Figure 1. Schematic of a standard genetic algorithm

Each individual within the population is assigned a fitness value, which expresses how good the solution is at solving the problem. The fitness value probabilistically determines how successful the individual will be at propagating its genes (its code) to subsequent generations. Better solutions are assigned higher values of fitness than worse performing solutions.

Evolution is performed using a set of stochastic genetic operators, which manipulate the genetic code. Most genetic algorithms include operators that select individuals for reproduction, produce new individuals based on those selected, and determine the composition of the population at the subsequent generation. *Crossover* and *mutation* are two well-known operators.

Figure 2. Single-point crossover

The crossover operator involves the exchange of genetic material between chromosomes (parents), in order to create new chromosomes (offspring). Various forms of this operator have been developed. The simplest form, single-point crossover, is illustrated in Figure 2. This operator selects two parents, chooses a random position in the genetic coding, and exchanges genetic information to the right of this point, thus creating two new offspring.

Figure 3. Binary mutation operator

The mutation operator, in its simplest form, makes small, random, changes to a chromosome. For a binary encoding, this involves swapping gene 1 for gene 0 with small probability (typically around one percent) for each bit in the chromosome, as shown in Figure 3.

Once the new generation has been constructed, the processes that result in the subsequent generation of the population are begun once more.

The genetic algorithm explores and exploits the search space to find good solutions to the problem. It is possible for a GA to support several dissimilar, but equally good, solutions to a problem, due to its use of a population.

Despite the simple concepts involved, the development and analysis of genetic algorithms is quite complicated. Many variations have been proposed since the first GA was introduced. Rigorous mathematical analysis of the GA is difficult and is still incomplete.

Genetic algorithms are robust tools, able to cope with discontinuities and noise in the problem landscape. Inclusion of domain-specific heuristics is not a pre-requisite, although it may improve the performance of a GA. They have proved useful at tackling problems that cannot be solved using conventional means.

2.2. Landscapes

The genetic algorithm seeks to maximize the mean fitness of its population, through the iterative application of the genetic operators previously described. The fitness value of a solution in the GA domain corresponds to a *cost* value in the problem domain. An explicit mapping is made between the two domains. ‘Cost’ is a term commonly

associated with traditional optimization problems. It represents a measure of performance: namely, the lower the cost, the better the performance. Optimizers seek to minimize cost. Hence, it is evident that, by maximizing fitness, the GA is effectively minimizing cost. Raw performance measures must be translated to a cost value. This process is usually straightforward for single objective problems, but becomes more complicated in the multiobjective case.

Every possible decision vector has an associated cost value and fitness value. The enumeration of all such vectors leads to the *cost landscape* and *fitness landscape* for the problem. For a problem with two decision variables, the cost and fitness landscapes will each be three-dimensional. An example is given in Figure 4. In general, if a problem has n decision variables (is n-dimensional), then the corresponding landscapes will be n+1 dimensional. The nature of a cost landscape depends on the chosen mapping from the vector of raw performance measure to the scalar cost value. The nature of the scalar fitness value subsequently depends on the translation from cost to fitness.

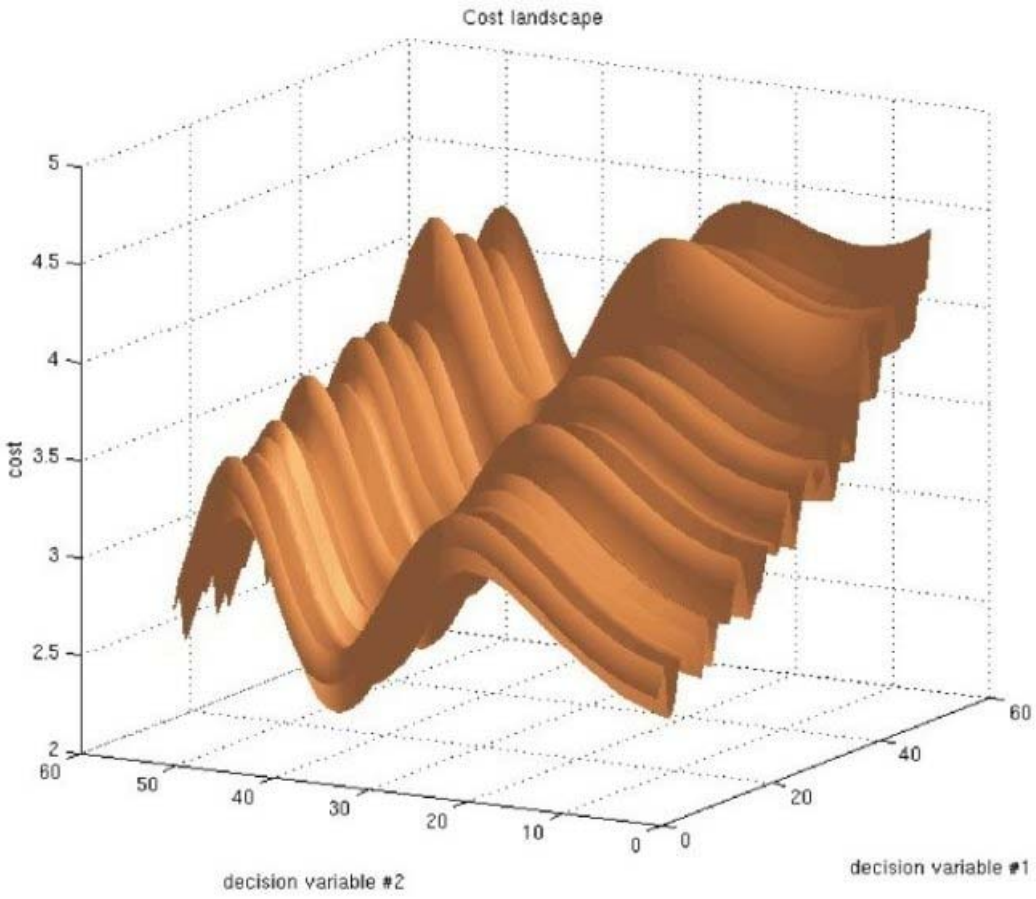

Figure 4. A multimodal cost landscape

2.3. Diversity

Many variations on the standard genetic algorithm, as presented by Goldberg in 1989, can be found in the literature. Modifications have been motivated by a desire to improve the performance of the GA, and to adapt it to particular problem domains. It may be

more helpful or appropriate to regard evolutionary computing as a general problem-solving methodology, rather than a specific parameter-less tool.

Virtually every aspect of the GA has been exposed to experimentation. As a note of caution, the results of such changes are often inconclusive and are frequently based on limited empirical testing. A very brief summary of key developments is presented below:

- **Population –** The size of the population has been of standard concern to both theorists and implementers. A population of between twenty and one hundred chromosomes is normally sufficient for most applications. The encoding of potential solutions to form chromosomes has also been the subject of intense research. Binary, or its *Gray* variant, encoding is the traditional approach, but direct floating-point representations of design parameters are becoming increasingly popular.

- **Fitness assignment –** Techniques for the conversion of the raw performance of a potential solution to a GA fitness value have also received much attention. Fitness is often taken as absolute, prior to normalization using the population average but, alternatively, ranking techniques may be used. The main aims are to prevent premature convergence (early in the search), and to prevent directionless search late in the search. Ranking is, arguably, the most effective method of achieving this.

- **Selection –** The standard roulette wheel selection method is known to produce biased results, leading to a phenomenon known as *genetic drift*. Two central aims in the development of alternatives are to eliminate statistical bias and to achieve potential parallelism. Other selection methods have been proposed, such as tournaments between two individuals (which achieves good parallelism), and stochastic universal sampling (which is unbiased). Note that a trade-off has been shown to exist between the two aims cited above.

- **Genetic manipulation –** Genetic operators have been subject to intensive discussion, over both the composition and purpose of the various operators. Some researchers have abandoned recombination, whilst others regard the effect of mutation as minimal. Essentially, recombination tends to direct the search to superior areas of the search space, whilst mutation acts to explore new areas of the search space and to ensure that genetic material cannot be irretrievably lost. Choice of genetic operators must be made together with choice of representation.

- **Iteration –** GAs evolve a population over a number of generations. The exact number depends on the speed with which convergence can be achieved, and is dependent on the interplay between the GA construction and the type of problem under consideration. The make-up of each new generation must be chosen. Typically, this will include offspring produced as a result of genetic operators acting on old individuals, some remnants of the past population, and possibly a few randomly generated individuals (see Figure 5). The exact proportion of each

tends to vary from implementation to implementation, but offspring usually dominate the new population. The ratio of offspring to population size is termed the *generational gap*. It may be a static value, or may vary dynamically during the course of a run. *Elitism* is the term given to describe the deliberate introduction of good past individuals into the new population.

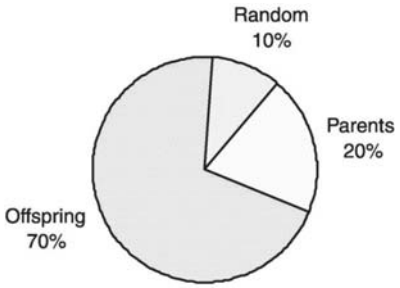

Figure 5. Typical composition of the new generation

- **New operators –** Various new operators have been introduced to address problems discovered in application. For example, *fitness sharing* can be used to encourage *niching* behavior (sub-population formation at different, comparatively optimal, landscape peaks). *Mating restriction* can be applied in cases where crossover between largely different solutions is unlikely to create good offspring (poor offspring are commonly described as *lethals*).

A particular area of interest is the endeavor to incorporate further parallelism within the GA methodology in order to improve the efficiency of the algorithm. Three main categories of parallel GA (PGA) can be defined, namely *global*, *migration*, and *diffusion* algorithms. Global PGAs treat the entire population as a single breeding unit and aim to exploit the inherent parallelism of the algorithm. Farmer-worker systems are a typical implementation, in which the workers carry out performance evaluations, or conduct genetic operations. In a migration-based PGA, the population is distributed amongst semi-isolated groups. From time to time, migration of individuals within the groups occurs. Diffusion PGAs are based on a local neighborhood selection mechanism. The population is treated as a single, continuous, structure. Breeding is restricted to adjacent individuals. This type of scheme tends to give rise to clusters of individuals of similar genetic material and fitness, known as 'virtual islands'. Overviews and comparisons of parallel genetic algorithms are available in the literature.

GAs have also been utilized as a component of hybrid problem-solving tools, including elements such as hill-climbing, simulated annealing, *neural networks*, Bayesian belief networks, and *fuzzy logic*.

Two key developments that have arisen from the GA are *genetic programming* (GP) and *multiobjective evolutionary algorithms* (MOEAs). General introductions are provided in the following subsections.

2.3.1. Genetic Programming

Genetic programming represents a major variation on the GA. It was developed in the

early 1990s, with the original purpose of generating and evaluating entire computer programs. The algorithm fundamentally resembles a GA but application of the operators requires special care. GP evaluates and manipulates variable length structures, in contrast to the generally fixed length chromosomes of a GA. The structures are composed of functions and terminals (potential inputs) that are defined in a library, prior to the execution of the GP. The maximum depth of any structure is, usually, also pre-defined.

GP uses a parse tree structure that is very similar to the Lisp programming language. However, the GP approach admits any problem for which the solution can be represented as a structure. Applications have involved manipulation of structures such as neural networks, system block diagrams, circuits, and equations. A single-point crossover operator for a block diagram representation is shown in Figure 6. A simple mutation operator is illustrated in Figure 7. This operator swaps, with low probability, a block within a particular solution for one from the library of possible blocks. GP embodies an entire field of research in its own right. Much literature on the subject is available, including synopses of trends and applications.

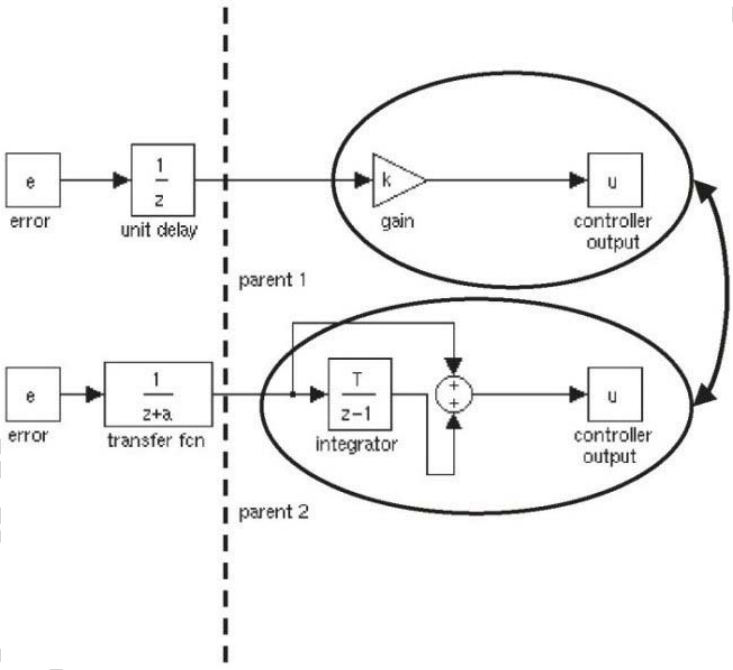

Figure 6. Single-point crossover for GP using block diagrams

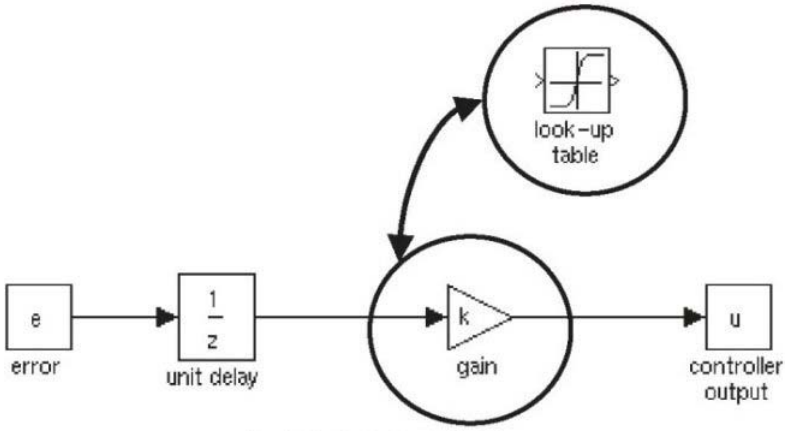

Figure 7. Mutation for GP using block diagrams

2.3.2. Multiobjective Evolutionary Algorithms

Real-world problems usually involve the simultaneous consideration of multiple performance criteria. These objectives are often non-commensurable and are often in conflict with one another. *Trade-offs* exist between some objectives, where advancement in one objective will cause deterioration in another. It is very rare for problems to have a single solution; rather, a family of *non-dominated* solutions will exist. These *Pareto-optimal* (PO) solutions are those for which no other solution can be found which improves on a particular objective without detriment to one or more other objectives. The concept of Pareto optimality is illustrated in Figure 8.

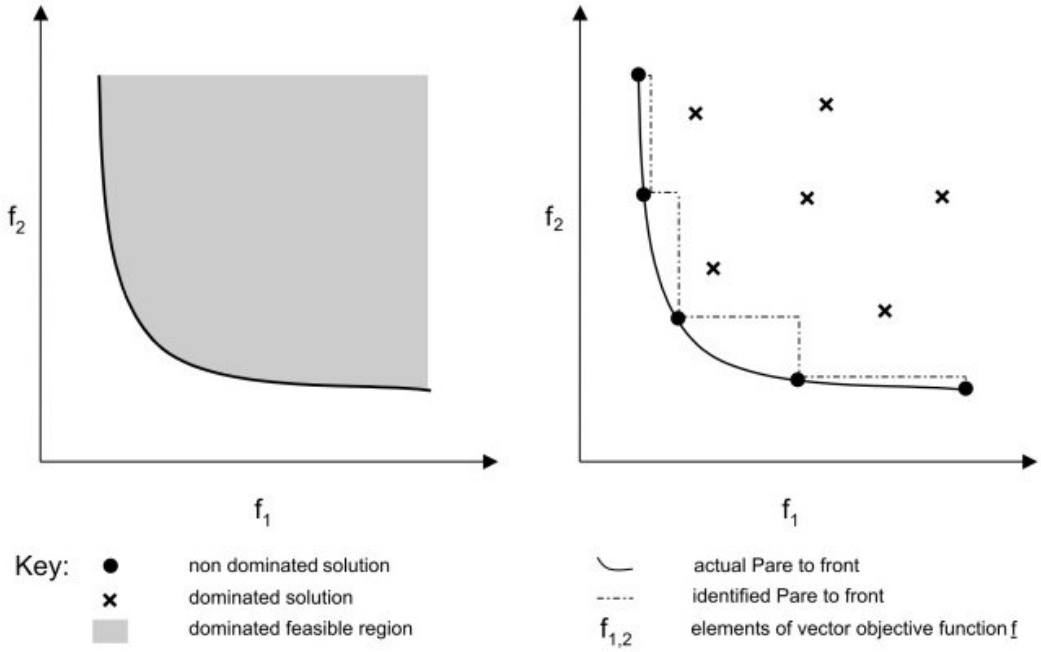

Figure 8. Pareto optimality

Evolutionary algorithms are a suitable technique for multiobjective optimization. Due to their population-based nature, they are capable of supporting several different solutions simultaneously. The robustness of the GA in the face of ill-behaved problem landscapes

increases the value of their utility. Research into multiobjective evolutionary algorithms is still in its infancy, and is likely to prove a highly fruitful line of investigation in the coming years. One of the first approaches to utilize the concept of Pareto optimality was Fonseca and Fleming's multiobjective genetic algorithm (MOGA), which tends to be the favored approach for control engineers. Several excellent surveys of MOEA (multiobjective evolutionary algorithm) development are available; MOGA is currently regarded as a "classic" MOEA with relatively few opportunities for improvement.

In the past, multiobjective problems have been cast as, effectively, single objective problems by constructing a utility function describing the relative importance of each objective. For example, in linear quadratic regulator design, the competing objectives of error and control size have in the past been combined as a weighted-sum of quadratic measures. The cost function is defined prior to the optimization procedure.

This requires in-depth information concerning the various trade-offs and valuation of each individual. This data is not commonly fully available in practice. Even then, such a procedure returns only a single solution (one point on the trade-off surface) per optimizer run. While multiple runs can be made with different settings for the weights, there is no guarantee that a uniform spread of objective weightings will provide a good spread of solutions. By contrast, evolutionary algorithms, due to their population-based nature, are capable of supporting several different solutions simultaneously.

Thus, in a single optimizer run, the decision-maker is provided with an indication of the trade-offs within the problem. Furthermore, the weighted-sum approach is unable to identify non-convex parts of the trade-off surface, potentially missing important areas for compromise. By contrast, the GA selection operator can be used to identify degrees of Pareto optimality, thus enabling objectives to be handled individually. Hence, the requirement for a forced combination of objectives and the need for *a priori* information are both avoided. Indeed, this kind of search can help to identify the existence and nature of specific trade-offs.

The central theme of MOEA research to date has been the search for a problem's *Pareto-front* (the set of non-dominated solutions). This set can be quite large and, hence, preference information may usefully be incorporated in order to direct the search to useful parts of the trade-off surface. Incorporation of designer preferences within a MOEA-based tool is a crucial area for further research.

MOEAs can be applied to a wide range of design problems, encompassing many different fields. For example, a MOGA has been applied to the optimization of radiotherapy treatment planning, in which the objectives are to deliver a high dose to the target area, whilst sparing the organs at risk, and minimizing the dose to other healthy tissue.

They have also been applied to engineering design problems such as supersonic wing-shape optimization and automotive engine design. Control-related applications are described in Section 4, many of which extend design capabilities of GA search methods based on single objectives.

-
-
-

Bibliography

Conference Publications

Annual Genetic Programming Conference (GP), (1996) – present, proceedings published by the *Massachusetts Institute of Technology* (MIT) Press. [An annual conference covering the field of genetic programming and now incorporated into the Genetic and Evolutionary Computation Conference (GECCO)]

Congress on Evolutionary Computation (CEC), (1999) – present, published by the *Institute of Electrical and Electronics Engineers* (IEEE) Press. [An annual international conference covering all forms of evolutionary computation]

Genetic ALgorithms in Engineering Systems: Innovations and Applications (GALESIA), (1995) – present, published by the *Institution of Electrical Engineers* (IEE). [A series of international conferences, held every two years, and now incorporated into the Congress on Evolutionary Computation]

Genetic and Evolutionary Computation COnference (GECCO), (1999) – present, proceedings published by Morgan Kaufmann. [An annual international conference that incorporates both the Genetic Programming Conference and the International Conference on Genetic Algorithms]

International Conference on Evolutionary Computation (ICEC), (1994 – 1998), proceedings published by IEEE Press [An annual conference on evolutionary computation, now part of GECCO]

International Conference on Genetic Algorithms (ICGA), (1985) – present, proceedings published by Morgan Kaufmann. [An annual meeting that brings together an international community from academia, government, and industry with an interest in evolutionary computation, now part of GECCO]

Parallel Problem Solving From Nature (PPSN), (1990) – present, published in the *Lecture Notes in Computer Science* series. [A series of international conferences, held every two years, based on paradigms taken from natural processes]

Papers and books

Ahmad, M., Zhang, L., and Readle, J. C., (1997), *On-line genetic algorithm tuning of a PI controller for a heating system*, GALESIA '97 – Genetic Algorithms in Engineering Systems: Innovations and Applications, pp510-515. [This paper represents one of the few examples in which a GA has been used on-line, if only for a demonstration-level system]

Chipperfield, A. J. and Fleming P. J., (1995), *Parallel genetic algorithms*, Chapter 39, Parallel and Distributed Computing Handbook (Ed. Zomaya, A.), McGraw-Hill, pp1034-1059. [A broad overview and comparison of parallel genetic algorithms]

Chipperfield, A. and Fleming P., (1996), *Multiobjective Gas Turbine Engine Controller Design Using Genetic Algorithms*, IEEE Transactions on Industrial Electronics, Vol. 43, No. 5, pp1-5, October 1996. [This article uses MOGA to select controller structure and suitable parameters for a multivariable control system for a gas turbine engine]

Coello Coello, C. A., (1999), *A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques*, Knowledge and Information Systems. An International Journal, Vol. 1, No. 3, pp269-308, August 1999. [A thorough survey of MOEA activity]

Darwin, C., (1859), *The Origin of Species*, John Murray, London.

Dimopoulos, C. and Zalzala, A. M. S., (2000), *Recent Developments in Evolutionary Computation for Manufacturing Optimization: Problems, Solutions, and Comparisons*, IEEE Transactions on Evolutionary Computation, Vol. 4, No. 2, pp93-113, July 2000. [A broad review of the use of evolutionary algorithms for manufacturing optimization]

Dorigo, M. and Columbetti, M., (1994), *Robot shaping: developing autonomous agents through learning*, Artificial Intelligence, Vol. 71, pp321-370. [A comprehensive article describing the behavioral control of a robot, within which GAs are used to evolve classifier systems]

Fonseca, C. M. and Fleming, P. J., (1998), *Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms – Part I: A Unified Formulation* and *Part II: Application Example*, IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans, Vol. 28, No. 1, pp26-37 and pp38-47, 1998. [Part I describes a scheme for the progressive articulation of designer preferences, which is incorporated into MOGA; Part II describes the application of MOGA to the multiobjective optimization of a gas turbine engine]

Goldberg, D. E., (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Reissue, Addison-Wesley Publishing Company. [The standard genetic algorithm textbook that was followed by an explosion of interest in the field of GAs]

Haas, O. C. L., Burnham, K. J., and Mills, J. A., (1997), *On improving physical selectivity in the treatment of cancer: a systems modelling and optimisation approach*, Control Engineering Practice, Vol. 5, No. 12, pp 1739-1745. [Multiobjective optimization of radiotherapy treatment planning using MOGA]

Holland, J. H., (1975), *Adaptation in Natural and Artificial Systems*, Ann Arbor: The University of Michigan Press. [The classic text that introduced the concept of genetic algorithms]

Koza, J. R., (1992), *Genetic Programming – On the Programming of Computers by Means of Natural Selection*, The MIT Press, Cambridge, Massachusetts. [The classic text that introduced genetic programming]

Koza, J. R., Keane, M. A., Yu, J., Bennett III, F. H., Mydlowec, W., (2000), *Automatic Creation of Human-Competitive Programs and Controllers by Means of Genetic Programming*, Genetic Programming and Evolvable Machines, Vol. 1, pp121-164. [This article uses genetic programming for the automatic synthesis of the parameter values and the topology of controllers]

Kristinsson, K. and Dumont, G. A., (1992), *System Identification and Control Using Genetic Algorithms*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 22, No. 5, pp1033-1046, September/October 1992. [This seminal paper presents methods for the application of GAs to system identification]

Lennon, W. K. and Passino, K. M., (1999), *Intelligent Control for Brake Systems*, IEEE Transactions on Control Systems Technology, Vol. 7, No. 2, pp188-202, March 1999. [In this article, GAs are applied on-line in a model-reference adaptive control scheme for a (simulated) vehicle braking system]

Linkens, D. A. and Nyongesa, H. O., (19950, *Genetic algorithms for fuzzy control – Part 1: Offline system development and application*, IEE Proceedings – Control Theory and Applications, Vol. 142, No. 3, pp161-176 and pp177-185, 1995. [A review of the off-line application (Part 1) and on-line application (Part 2) of GAs to fuzzy control]

Linkens, D. A. and Nyongesa, H. O., (1996), *Learning systems in intelligent control: an appraisal of fuzzy, neural and genetic control applications*, IEE Proceedings – Control Theory and Applications, Vol. 143, No. 4, pp367-386, July 1996. [A comprehensive appraisal of fuzzy control and neural control]

Marrison, C. I. and Stengel, R. F., (1997), *Robust Control System Design Using Random Search and Genetic Algorithms*, IEEE Transactions on Automatic Control, Vol. 42, No. 6, pp835-839, 1997. [This article uses a GA to explore the space of possible compensator parameters within a scheme that utilizes stochastic robustness analysis]

Michalewicz, Z., (1996), *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd Ed., Springer-Verlag, Berlin. [A GA textbook, with in-depth chapters on various aspects of the algorithm such as the use of floating-point encodings of decision variables]

Michalewicz, Z. and Fogel, D. B., (2000), *How to Solve It: Modern Heuristics*, Corrected Second

Printing, Springer-Verlag, Berlin. [This book describes approaches to problem-solving and includes many popular heuristic algorithms]

Mühlenbein, H. and Schlierkamp-Voosen, D., (1993), *Predictive Models for the Breeder Genetic Algorithm*, Evolutionary Computation, Vol. 1, No. 1, pp25-49. [This article introduces the Breeder Genetic Algorithm, which has been shown to perform very well on benchmark GA problems]

Oliveira, P., Sequeira, J., and Sentieiro, J., (1991), *Selection of Controller Parameters using Genetic Algorithms*, Engineering Systems with Intelligence. Concepts, Tools, and Applications, Kluwer Academic Publishers, Dordrecht, Netherlands, pp431-438. [This article represents an early application of GAs to PID parameter optimisation]

Onnen, C., Babuška, R., Kaymak, U., Sousa, J. M., Verbruggen, H. B., and Isermann, R., (1997), *Genetic algorithms for optimization in predictive control*, Control Engineering Practice, Vol. 5, Iss. 10, pp1363-1372. [This article reports on the application of GAs to the determination of an optimal control sequence in model-based predictive control]

Painton, L. and Campbell, J., (1995), *Genetic Algorithms in Optimization of System Reliability*, IEEE Transactions on Reliability, Vol. 44, No. 2, pp172-178, June 1995. [In this article, a GA-based technique is developed to improve overall system reliability by means of component-level choices]

Rodríguez-Vázquez, K., Fonseca, C. M., and Fleming P. J., (1997), *Multiobjective Genetic Programming: A Nonlinear System Identification Application*, Late Breaking Papers at the 1997 Genetic Programming Conference, pp207-212. [In this paper, multiobjective genetic programming is used to simultaneously optimize seven objectives for the purposes of NARMAX model structure identification]

Schroder, P., Green, B., Grum, N., and Fleming, P. J., (2001), *On-line evolution of robust control systems: an industrial active magnetic bearing application*, Control Engineering Practice, Vol. 9, No. 1, pp37-49, 2001. [On-line tuning of a controller, whilst the plant is off-line, using MOGA]

Schwefel, H.-P., (1995), *Evolution and Optimum Seeking*, John Wiley, New York. [The classic text that introduced the concept of ES]

Spears, W. M., De Jong, K. A., Bäck, T., Fogel, D. B., and de Garis, H., (1993), *An Overview of Evolutionary Computation*, Machine Learning: ECML-93, European Conference on Machine Learning, Lecture Notes in Artificial Intelligence, No. 667, Ed. Brazdil, P.B., pp442-459. [This paper compares and contrasts the different types of evolutionary algorithm; namely, genetic algorithms, evolution strategies, and evolution programming].

Veldhuizen, D. A. Van and Lamont, G. B., (2000), *Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art*, Evolutionary Computation, Vol. 8, No. 2, pp125-147. [A thorough survey of MOEA activity]

Vlachos, C., Williams, D., and Gomm, J. B., (1999), *Genetic approach to decentralised PI controller tuning for multivariable processes*, IEE Proceedings – Control Theory and Applications, Vol. 146, No. 1, pp58-64, 1999. [This paper reports on the application of a GA to the tuning of decentralized PI controllers for multivariable processes]

Wolpert, D. H. and Macready, W. G., (1995), *No Free Lunch Theorems for Search*, Technical Report SFI-TR-95-02-010, The Santa Fe Institute, New Mexico. [This report argues the case for the *No Free Lunch Theorem*, which states that if one algorithm performs better than another on a particular problem then the converse will be true for a different problem]

Biographical Sketches

Peter Fleming is Professor of Industrial Systems and Control at the University of Sheffield and Director of the Rolls-Royce University Technology Centre for Control and Systems Engineering. He is Vice-President of the International Federation of Automatic Control and the Editor of *International Journal of Systems Science*. His research interests in control systems engineering, optimization and evolutionary computing have led to close links with industries in sectors such as aerospace, power generation, food processing, pharmaceuticals and manufacturing.

Robin Purshouse received the MEng degree in control systems engineering from the University of

Sheffield in 1999. He subsequently held the position of IT Consultant at Logica plc, assigned to government-funded projects within the UK defence sector, before returning to Sheffield as a postgraduate research student. He is a Member of the Institution of Electrical Engineers and of the Institute of Electrical and Electronics Engineers. Robin's core research interests are multiobjective optimization (including search mechanisms, decision-making and visualization) and metaheuristics such as evolutionary algorithms.