Multi-objective evolutionary design of robust controllers on the grid

SHENFIELD, Alex <http://orcid.org/0000-0002-2931-8077> and FLEMING, Peter

Multi-objective Evolutionary Design of Robust Controllers on the Grid ⋆

Alex Shenfield * **Peter J. Fleming** **

* *Manchester Metropolitan University, Manchester, M1 5GD (e-mail: a.shenfield@mmu.ac.uk).*
** *University of Sheffield, Sheffield, S1 3JD (e-mail: p.fleming@sheffield.ac.uk).*

Abstract:
The emerging paradigm of grid computing provides a powerful platform for the solution of complex and computationally expensive problems. An example of this is the multi-objective evolutionary design of robust controllers, where each candidate controller design has to be synthesised and the resulting performance of the compensated system evaluated by computer simulation. This paper introduces a grid-enabled framework for the multi-objective optimisation of computationally expensive problems, before using the framework in the multi-objective evolutionary design of a robust lateral stability controller for a real-world aircraft using H_∞ loop shaping.

Keywords: Aerospace Stability Control, H_∞ Controller Design, Multiobjective Evolutionary Algorithms, High Performance Computing

1. INTRODUCTION

Modern aircraft consist of many complex subsystems, all of which require robust and reliable control. These systems are often multi-variable, consisting of multiple inputs and multiple outputs, and frequently the desired responses of a subsystem (such as rise time and overshoot) are in conflict. Coupling evolutionary multi-objective optimisation techniques with conventional controller design methods such as H_∞ or LQG control can provide the engineer with a powerful tool for addressing such problems (Fleming and Purshouse, 2002). However, such methods are frequently computationally expensive, requiring many thousands of controller designs to be evaluated.

Grid computing offers one potential solution to the computationally expensive nature of this evolutionary controller design process. The grid computing paradigm aims to provide "a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities" (Foster and Kesselman, 1999). This paradigm is differentiated from traditional approaches to distributed computing by its emphasis on providing "a seamless, integrated computational and collaborative environment" (Baker et al., 2002) for the solution of complex problems by allowing coordinated resource sharing across dynamic virtual organisations (Foster et al., 2001).

The purpose of this paper is to introduce a grid-enabled framework for evolutionary multi-objective controller design. This framework will then be applied to the design of a robust controller for the flight dynamics of a real-world aircraft - a complex problem with many (often conflicting) objectives to consider. The paper is organised as follows: section 2 will provide a brief introduction to multi-objective evolutionary algorithms, section 3 will introduce the grid-enabled framework for multi-objective evolutionary optimisation, section 4 will outline the controller design problem considered in this paper and its solution using our grid-enabled framework, and section 5 will present our conclusions and outline some ideas for further work.

⋆ The authors gratefully acknowledge the financial support of the Engineering and Physical Research Council in the UK under Grant Number GR/R67668/01.

2. MULTI-OBJECTIVE EVOLUTIONARY ALGORITHMS

2.1 Evolutionary Algorithms

Evolutionary Algorithms (EAs) utilise some of the concepts behind natural selection to iteratively evolve a population of candidate solutions to a problem (Goldberg, 1989). They both explore the solution space of a problem (by using variation operators such as mutation and recombination) and exploit valuable information present in the previous generation of candidate solutions (by using a selection operator). The trade-off between exploration of undiscovered regions of the solution space and exploitation of promising areas already discovered by the algorithm is extremely important: too much exploration and the algorithm will take too long to converge on a useful solution, too much exploitation and the algorithm may converge prematurely to local optima.

A key feature of evolutionary algorithms that makes them applicable across many different problem domains (including those where conventional optimisation techniques may struggle) is the use of evaluation function information directly, rather than derivative information or other

auxiliary knowledge. For many non-trivial real-world applications this evaluation function information is obtained by computer simulation of the system. For example, in the optimisation of maintenance schedules for gas turbine aero-engines (Shenfield et al., 2010), the cost information for each schedule is obtained by computer simulation of the candidate solution over 25 years.

The use of computer simulations to obtain this evaluation function information leads to some new issues. To ensure that the results gained from the evolutionary algorithm are meaningful, the simulation must be complex enough to capture all the relevant dynamics of the true system. However, assuming that this level of complexity is obtainable, this may lead to the simulation becoming very computationally expensive. Since EAs are iterative and population based the simulation may have to be run several thousand times.

2.2 Multi-objective Evolutionary Algorithms

Many real-world engineering problems involve the satisfaction of multiple, often conflicting, objectives. In this case it is unlikely that a single optimal solution will exist. Instead, the solution of this kind of multi-objective problem leads to a set of Pareto optimal points, where any improvement in one objective will lead to a deterioration in one or more of the other objectives (see Fig. 1). Conventional multi-objective optimisation methods struggle in these situations as they are often unable to obtain a distribution of potential solutions from this Pareto optimal set, instead just finding a single point. However, since evolutionary algorithms search a population of candidate solutions in parallel, they are able to find multiple Pareto optimal solutions. This provides the engineer with a set of potential solutions to choose from, rather than a single solution that may not meet the required performance criteria.

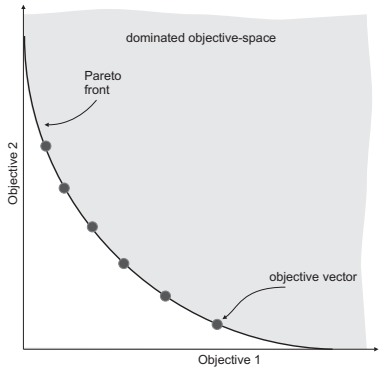

Fig. 1. The optimal solution set for a bi-objective problem

2.3 Parallel Evolutionary Algorithms

The computationally expensive nature of the evaluation process of evolutionary algorithms has motivated the development of parallel EAs. Early approaches to the implementation of parallel evolutionary algorithms can be classified into two categories which still apply today: single-population global EA implementations and EA implementations with multiple communicating populations (Cantú-Paz and Goldberg, 1999).

Single-population parallel evolutionary algorithms consist of a single *panmictic* population maintained globally. This form of parallelism may be effectively exploited using the well established Master-Slave communication paradigm (see Fig. 2). Typically the evaluation of candidate solutions in the algorithm is distributed amongst the worker nodes whilst the master node applies the evolutionary operators, such as selection and variation, centrally to the whole population (Fogarty and Huang, 1991). Chipperfield and Fleming (1995) also describe a similar scheme where both the evaluation of candidate solutions and the variation operators are performed by the worker nodes.

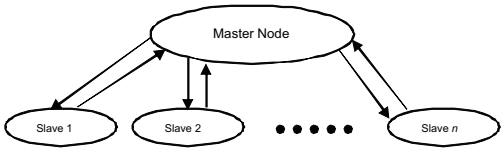

Fig. 2. The master-slave communication paradigm

These single-population, globally parallel EAs represent an important case of parallelism because they are functionally equivalent to sequential EAs. This means that existing EA theory and design guidelines can easily be applied to their use (Cantú-Paz and Goldberg, 1999). Although this type of strategy does not exploit all the parallelism inherent in the evolutionary algorithm, substantial improvements in performance can be achieved - especially in cases where the evaluation of candidate solutions is significantly more computationally expensive that the evolutionary operators themselves (Chipperfield and Fleming, 1995).

In contrast, the use of parallel EAs based around multiple communicating populations can potentially exploit more of the inherent parallelism in the algorithm by allowing subpopulations to evolve independently. However, using multiple populations substantially increases the complexity of the design process since consideration needs to be given to not only the number and size of subpopulations but also to the migration scheme and population topology used by the algorithm. The choice of each of these parameters affects both the efficiency of the algorithm and the quality of the overall solution.

3. A GRID-ENABLED FRAMEWORK FOR EVOLUTIONARY MULTI-OBJECTIVE OPTIMISATION

3.1 Grid Computing Technologies

The concept of grid computing is not new. As far back as 1969 Len Kleinrock suggested:

> "We will probably see the spread of 'computer utilities', which, like present electric and telephone utilities, will serve individual homes and offices across the country." (Kleinrock, 1969)

However, it is only recently that technologies such as the Globus Toolkit (Foster and Kesselman, 1999) and web services have emerged to enable this kind of aggregation of compute resources. One of the key features of the

open-source Globus Toolkit is its ability to provide a common means of interacting with the diverse range of local resource management systems that are often found on compute resources in the real-world. Globus builds on current web service technologies to support the creation and management of ensembles of services maintained by virtual organisations (Foster et al., 2002).

3.2 Parallelisation of the Evolutionary Algorithm

In section 2.3 we considered two models of parallel evolutionary computation: multiple communicating populations, and single-population master-slave implementations. The decision as to which of these forms of parallelism to implement must consider several factors, including ease of implementation and the potential performance gains from parallelism. Single-population parallel EAs are often easiest to implement and use, since experience gained with sequential algorithms is directly applicable. In contrast, the implementation of parallel EAs with multiple communicating populations requires the consideration of extra design choices increasing the complexity of the design process.

For the grid-enabled framework presented in this paper it was decided to use a single-population master slave model since it is less complex than the multiple communicating populations model and is well suited to implementation in a heterogeneous grid computing environment because it keeps inter-process communication to a minimum. Although this globally parallel EA does not exploit all the potential parallelism in the algorithm, it significantly accelerates the evaluation of candidate solutions (the most computationally expensive part of the algorithm).

3.3 Service-Oriented Architecture

We have chosen to implement our grid-enabled framework for evolutionary multi-objective optimisation in a service-oriented architecture using grid services based on the Globus Toolkit to provide access to the resources in the grid. This framework is written using the Java programming language so as to provide portable code that allows components of the framework to be easily run across various heterogeneous platforms.

A service-oriented architecture is essentially a collection of services that communicate with each other in order to perform a complex task. This service-oriented architecture approach to grid computing is well suited to the kind of master-slave parallelism used in our optimisation framework since, in this view of grid computing, the client acts as the master node and the service acts as the slave. The implementation of our optimisation framework (see Fig. 3) involves two different types of service. One service type exposes the operations of the evolutionary algorithm to the client and the other provides the ability to run evaluations of the objective function on the resources of the computational grid.

The Evaluation Factory Service shown in Fig. 3 implements a grid-based meta-scheduling architecture to provide access to the available grid resources via the Globus Toolkit. Our grid-enabled framework for evolutionary optimisation then assigns these evaluation function instances

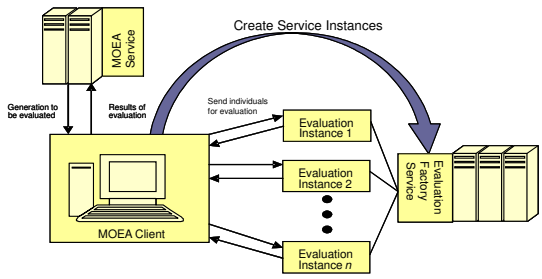

Fig. 3. Overview of the multi-objective evolutionary optimisation framework

to the available processors for execution (using an optimised workload allocation algorithm (Tang and Chanson, 2000)).

This approach to the grid-enabling of our multi-objective evolutionary optimisation framework also provides flexibility both in how the framework is used and in its maintenance. The use of services to provide functionality to our optimisation framework means that new features can be added, and existing features improved, simply by providing new services. In the context of our optimisation framework this functionality could be anything from the implementation of additional evolutionary operators to changes in the way candidate solutions are distributed and managed. The use of web services also means that the functionality can be accessed via the HTTP protocol. This allows elements of the optimisation framework to be easily integrated into an Internet portal and accessed by any device with a capable web browser.

3.4 The White Rose Grid

Our evolutionary multi-objective optimisation framework was implemented using the resources of the White Rose Grid (The White Rose University Consortium, 2010), a multi-institutional computational grid launched in 2002. The White Rose Grid consists of 4 core nodes - 2 at the University of Leeds, 1 at the University of Sheffield, and 1 at the University of York - providing over 500 processor cores for local users and distributed e-Science research. Each cluster is managed locally using a combination of Sun Grid Engine and the Globus Toolkit and is connected to the rest of the White Rose Grid via the high speed Yorkshire and Humberside Metropolitan Area Network (YHMAN).

4. DESIGN OF A ROBUST CONTROLLER FOR AIRCRAFT FLIGHT DYNAMICS

4.1 Flight Dynamics

The dynamics of an aircraft in flight can be described by the rotational moments around its centre of gravity (CG) in Cartesian space. These are shown in Fig. 4 where:

- **L** is the *rolling* moment of the aircraft.
- **M** is the *pitching* moment of the aircraft.
- **N** is the *yawing* moment of the aircraft.

These flight dynamics can be separated into *longitudinal* motions, in which the wings remain level (e.g. pitch), and *lateral* motions such as roll and yaw.

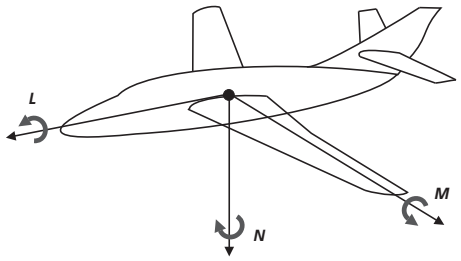

Fig. 4. The rotational moments of an aircraft

The equations of motion describing these flight dynamics are non-linear; however, by applying the *small disturbance theory* (Nelson, 1998), a linearised model can be found. This linearisation process will only give a good result in cases where the motion of the aircraft can be fully described by small deviations about a steady flight condition (such as in the flight of large commercial aircraft), and therefore should not be used in cases where large amplitude motions are likely to occur.

4.2 Controller Design

The control system for the flight dynamics of an aircraft must provide robust and responsive multi-variable control of the ailerons and the rudder, as well as guaranteeing stability in the presence of modelling uncertainty. H_∞ control theory (Zames, 1981) offers a proven method of designing controllers that are robust to such uncertainty. However, a drawback of robust stabilisation using H_∞ control is the inability of the designer to specify performance requirements (Skogestad and Postlethwaite, 1996), which can result in compensated systems that, whilst robust, perform unsatisfactorily. To overcome this limitation, McFarlane and Glover (1990) proposed using pre- and post-compensators to 'shape' the open-loop response of the plant (see Fig. 5), and then applying robust stabilisation. Selecting the weighting matrices for the pre- and post-compensators is typically challenging since these choices govern the performance of the resulting system.

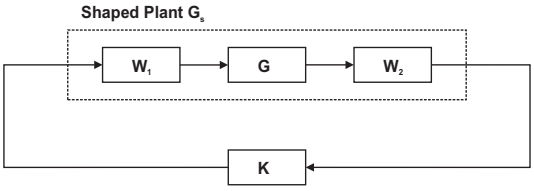

Fig. 5. The shaped and compensated plant

This paper aims to find an optimal H_∞ loop shaping controller for the lateral stability control of a Boeing 747 aircraft using our grid-enabled framework for evolutionary multi-objective optimisation. The Boeing 747 model used in this controller design is a linearised multivariable system with two inputs (the control signals for the aileron and rudder) and two outputs (the roll and sideslip angles), and can be represented by the following transfer function matrix:

$$\mathbf{G} = \begin{pmatrix} \mathbf{g_{11}} & \mathbf{g_{12}} \\ \mathbf{g_{21}} & \mathbf{g_{22}} \end{pmatrix}$$

where:

$$\mathbf{g_{11}} = \frac{0.1845s^2 + 0.04795s + 0.1995}{s^4 + 0.6807s^3 + 1.049s^2 + 0.3373s - 0.001979}$$

$$\mathbf{g_{12}} = \frac{0.06591s^2 - 0.12s - 0.5158}{s^4 + 0.6807s^3 + 1.049s^2 + 0.3373s - 0.001979}$$

$$\mathbf{g_{21}} = \frac{-0.01448s^2 - 0.01962s + 0.001359}{s^4 + 0.6807s^3 + 1.049s^2 + 0.3373s - 0.001979}$$

$$\mathbf{g_{22}} = \frac{0.005334s^3 + 0.4377s^2 + 0.1884s - 0.00432}{s^4 + 0.6807s^3 + 1.049s^2 + 0.3373s - 0.001979}$$

We will use our grid-enabled framework for evolutionary multi-objective optimisation to determine the optimal weights for the pre- and post-compensators used to improve the performance of the overall compensated system. These compensators have the following structures:

$$\mathbf{W_1} = \begin{pmatrix} \frac{s+a}{s} & 0 \\ 0 & \frac{s+b}{s} \end{pmatrix}$$

$$\mathbf{W_2} = \begin{pmatrix} c & 0 \\ 0 & d \end{pmatrix}$$

with the order of these compensators being determined by the order of the plant.

4.3 Optimisation of Controller Performance

The H_∞ loop shaping controller design process can be formulated as a multi-objective optimisation problem, where each performance requirement is treated as a separate objective, and thus solved using a multi-objective evolutionary algorithm (see section 2). The decision variables in this optimisation procedure are the weights, a, b, c, d, in the compensators. However, this controller design problem is computationally expensive since, for every candidate solution, an H_∞ controller has to be synthesized and the response of the compensated system obtained by computer simulation. The evaluation of a single candidate solution for this problem took in the order of 5.5 seconds on a Intel Core 2 Duo based PC with a clock speed of 2.60GHz.

To overcome the computationally expensive nature of this controller design problem we have used the grid-enabled framework for evolutionary multi-objective optimisation described in this paper. We chose a real-valued representation for the compensator weights since Fogel and Ghoziel (1997) have shown that there is no intrinsic advantage in choosing one bijective representation over another, although particular representations may be more computationally tractable or efficient for certain problems. As a consequence of this, modern MOEA practice emphasises choosing a representation that is appropriate for the problem under consideration (Michalewicz and Fogel, 2000). Selection in our algorithm was performed using Stochastic Universal Sampling (Baker, 1987) which guarantees sampling with zero bias and minimum spread, and is generally considered superior to other selection schemes for

many problems (Hancock, 1994). The extended intermediate recombination operator and BGA mutation operator (Mühlenbein and Schlierkamp-Voosen, 1993) were used to introduce variation into the population and prevent the search process from stagnating. However, it is important to note that the implementation of our grid-enabled framework in a service-oriented architecture (see section 3.3) provides a high degree of flexibility in the choice of algorithm architecture, representation, and evolutionary operators used. This flexibility means it is simple to adapt our framework to other optimisation problems.

A set of performance requirements arising from domain specific knowledge about the problem have been specified for the response of the compensated system (see Table 1). Some of these requirements are hard constraints and others are simply desired goals. Several of the performance requirements for this controller are in competition which makes achieving all the goals difficult. The grid-enabled MOEA implementation described in this paper uses the preference articulation operator proposed by Fonseca and Fleming (1998) to handle both the stated goals and hard contraints from Table 1.

Table 1. Performance requirements for the H_∞ controller design problem

Requirements	Type
Minimise the Overshoot in response to a step input	Goal (Overshoot $<$ 5%)
Minimise the Rise Time	Goal (T_r $<$ 3 seconds)
Minimise the Settling Time	Goal (T_s $<$ 4 seconds)
Prevent Aileron actuator saturation	Constraint (Aileron deflection $<$ 0.349 radians)
Prevent Rudder actuator saturation	Constraint (Rudder deflection $<$ 0.52 radians)
Controller must be robust to 30% multiplicitive uncertainty	Constraint

4.4 Results

It can be seen from Fig. 6 that there is a strong trade-off between the control signal for the aileron actuator and the other requirements in Table 1 (such as the rise time and settling time). This shows that the response of the final compensated system is limited by the maximum range of the aileron actuator.

Table 2 shows the achieved performance of the compensated system and Fig. 7 shows the response of the system to a 0.1 radians step change in the aileron control signal. It can be seen from Table 2 that all the performance requirements specified in Table 1 are satisfied, and significant improvements over all the goal values have been achieved.

Table 3 shows a representative set of execution times from the evolutionary multi-objective optimisation of the controller design problem presented in this paper. These times are averaged over 5 runs for both the results obtained from a single workstation and the results obtained using the resources of the White Rose Grid. As Table 3 shows, the grid-enabled optimisation framework presented in this paper has significantly reduced the time taken to optimise the performance of the compensated system.

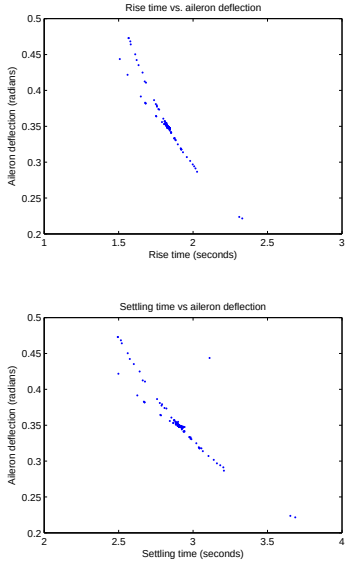

Fig. 6. Trade-offs between aileron control signal and other performance requirements

Table 2. Achieved performance of the compensated system

Goals	Achieved Value
Minimise the Overshoot in response to a step input (Goal: Overshoot $<$ 5%)	0.87%
Minimise the Rise Time (Goal: T_r $<$ 3 seconds)	1.83 seconds
Minimise the Settling Time (Goal: T_s $<$ 4 seconds)	2.9 seconds
Prevent Aileron actuator saturation (Constraint: Aileron deflection $<$ 0.349 radians)	0.3488 radians
Prevent Rudder actuator saturation (Constraint: Rudder deflection $<$ 0.52 radians)	0.0188 radians
Controller must be robust to 30% multiplicitive uncertainty (Constraint)	Robust to 35.9% multiplicitive uncertainty

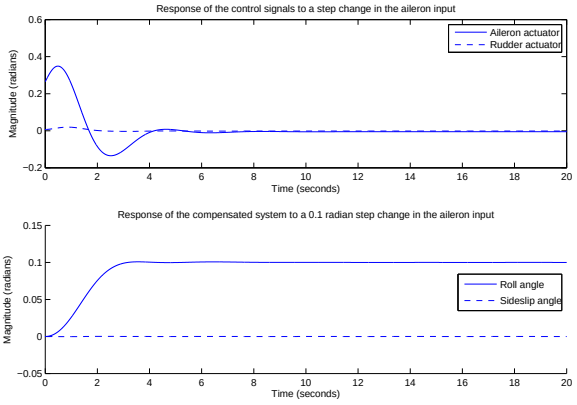

Fig. 7. Step response of the compensated system to aileron deflection

Table 3. Execution times for the optimisation of an H_∞ controller for aircraft flight dynamics

Single Workstation		Computational Grid	
50 gen.	100 gen.	50 gen.	100 gen.
28637 s	56698 s	8251 s	12582 s

5. CONCLUSIONS AND FURTHER WORK

Table 3 has shown that significant reductions in the execution times of our evolutionary multi-objective controller design process can be achieved by using our grid-enabled optimisation framework. Whilst the example presented in this paper uses a multi-objective evolutionary algorithm in the optimisation process, the service-oriented nature of our framework means it is easily extensible to other iterative optimisation algorithms such as ant-colony or particle swarm optimisation.

The grid-enabled framework for multi-objective optimisation described in this paper is best suited to computationally expensive evaluation functions such as the robust controller design problem presented in section 4. This is due to the communication overheads involved in the distribution and management of the evaluation function jobs across multiple diverse resources. For some computationally trivial evaluation functions this distribution and management may result in a degradation in performance compared with a sequential MOEA on a single machine. Whilst further work is planned to determine the scale of problems for which this framework is most effective, it is expected that research and development in the field of grid-middleware, job submission services and job management services will result in a reduction in these communication overheads. This will allow our framework to provide increased performance for less computationally expensive problems. However, our framework is not intended to replace sequential MOEAs in cases where the performance of the sequential MOEA is satisfactory.

REFERENCES

Baker, J.E. (1987). Reducing bias and inefficiency in the selection algorithm. In J.J. Grefenstette (ed.), *Proceedings of the Second International Conference on Genetic Algorithms*, 14–21. Lawrence Erlbaum.

Baker, M., Buyya, R., and Laforenza, D. (2002). Grid and grid technologies for wide area distributed computing. *Software: Practice and Experience*, 32(15), 1437–1466.

Cantú-Paz, E. and Goldberg, D.E. (1999). On the scalability of parallel genetic algorithms. *Evolutionary Computation*, 7(4), 429–449.

Chipperfield, A.J. and Fleming, P.J. (1995). Parallel genetic algorithms. In A.Y. Zomaya (ed.), *Parallel And Distributed Computing Handbook*, chapter 39, 1118–1144. McGraw-Hill.

Fleming, P.J. and Purshouse, R.C. (2002). Evolutionary algorithms in control systems engineering: a survey. *Control Engineering Practice*, 10, 1223 – 1241.

Fogarty, T.C. and Huang, R. (1991). Implementing the genetic algorithm on transputer based parallel processing systems. In H.P. Schwefel and R. Männer (eds.), *Parallel Problem Solving from Nature 1*, volume 496 of *Lecture Notes in Computer Science*, 145–149. Springer-Verlag, Berlin.

Fogel, D.B. and Ghoziel, A. (1997). A note on representations and variation operators. *IEEE Transactions on Evolutionary Computation*, 1(2), 159–161.

Fonseca, C.M. and Fleming, P.J. (1998). Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part I: A unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 28(1), 26–37.

Foster, I. and Kesselman, C. (1999). The Globus Toolkit. In I. Foster and C. Kesselman (eds.), *The GRID: Blueprint for a New Computing Infrastructure*, chapter 11, 259–278. Morgan Kaufmann.

Foster, I., Kesselman, C., Nick, J.M., and Tuecke, S. (2002). Grid services for distributed system integration. *IEEE Computer*, 35(6), 37 – 46.

Foster, I., Kesselman, C., and Tuecke, S. (2001). The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications*, 15(3), 200–222.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.

Hancock, P.J.B. (1994). An empirical comparison of selection methods in evolutionary algorithms. In T.C. Fogarty (ed.), *Evolutionary Computing - AISB Workshop*, volume 865 of *Lecture Notes in Computer Science*, 80–94. Springer-Verlag.

Kleinrock, L. (1969). UCLA press release. URL `http://www.lk.cs.ucla.edu/LK/Bib/REPORT/press.html`.

McFarlane, D. and Glover, K. (1990). *Robust Controller Design Using Normalized Coprime Factor Plant Descriptions*, volume 138 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag.

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

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

Nelson, R.C. (1998). *Flight Stability and Automatic Control*. McGraw-Hill, second edition.

Shenfield, A., Fleming, P.J., Allan, J., and Kadirkamanathan, V. (2010). Optimisation of maintenance scheduling strategies on the grid. *Annals of Operations Research*, 180(1), 213 – 231.

Skogestad, S. and Postlethwaite, I. (1996). *Multivariable feedback control - Analysis and Design*. John Wiley & Sons.

Tang, X. and Chanson, S.T. (2000). Optimizing static job scheduling in a network of heterogeneous computers. In *Proceedings of the International Conference on Parallel Processing*, 373–382.

The White Rose University Consortium (2010). The white rose grid website. URL `http://www.wrgrid.org.uk/`. Viewed 20 September 2010.

Zames, G. (1981). Feedback and optimal sensitivity: model reference transformations, multiplicitive seminorms, and approximate inverse. *IEEE Transactions on Automatic Control*, 26(2), 301–320.