APPLICATION OF A GENETIC ALGORITHM IN CRANE MOVEMENT SCHEDULING

Soren J. Henriksen [*],
Liuping Wang [*],**Graham C. Goodwin** [*],**Andrew Shook** [**],

[*] *Centre for Integrated Dynamics and Control*
University of Newcastle, Callaghan NSW 2308, Australia
Ph +61 2 4921 7072, FAX: +61 2 4960 1712
[**] *Growth and Technology, BHP Copper*
7400 North Oracle Road, Tucson Az. 85704, USA

Abstract: This paper develops an operator guidance system for an industrial process which involves scheduling ideas from manufacturing systems. The industrial process under study is the converter aisle in a copper smelter. Our objective is to increase throughput by improving utilization of the smelter's resources. This paper partly addresses this objective by applying scheduling techniques to optimize the usage of the plant's bridge cranes which supply material movement. As a benchmark process, a mathematical model for the cranes is developed with constraints allowing realistic simulation studies to be performed. Genetic algorithms are used to schedule the cranes' optimal movement.

Keywords: Discrete Event Systems, Genetic Algorithms, Scheduling algorithms

1. INTRODUCTION

This paper discusses the problem of scheduling crane operation in an industrial application. The specific example used is part of the process in the San Manuel copper smelter. There are four bridge cranes available for the transportation of materials in the converter aisle of the smelter. The scheduling of these cranes is currently performed manually where the converter operators communicate directly with crane operators to request services. Our objective is to design an operator guidance system to assist the operators with scheduling decisions made in the normal operation of the plant.

The scheduling of crane operations may be viewed as a discrete event optimal control problem. A background to theory on discrete event systems may be found in (Ramadge and Wonham, 1989). As a scheduling problem, it has similarity to other optimisation problems, such as the operation of elevators (Pepyne *et al.*, 1996). The dynamics of operation can be regarded as state transitions in response to external and internal events, and the performance measure may be introduced as the total time taken for fixed set of jobs to be performed. For this class of problems, we look to traditional control for inspiration and note the basic concept behind the control of continuous time plants is inversion (Goodwin *et al.*, 1998). It follows that one strategy that could be used for crane operation is to attempt some form of inversion of a discrete event model of the plant. This means finding a family of inputs that will cause the desired output. Unlike many continuous systems, however, the models for discrete event systems are not easily invertible. Hence it is necessary to add a search strategy to the basic idea of inversion. The basis for this approach is, in principle, straightforward. Namely, one simulates all of the control possibilities and chooses the best one based on the performance measure.

There are many algorithms available for performing the search task in the design of discrete event system control. Among the well-known methods are the random search, branch and bound (Hillier and Lieber-

[1] This project was supported by BHP Copper.

man, 1967), hillclimbing (Bolc and Cytowski, 1992) and genetic algorithms (Bolc and Cytowski, 1992). In a random search, control actions are chosen randomly for a large number of simulations, and the best solution is taken. This is conceptually easy, but inefficient in obtaining a solution. The branch and bound search technique differs from a random search since it is exhaustive over all feasible solutions. It begins as an exhaustive search, but possible subsets of solutions are evaluated during computation and discarded as soon as they are found to be sub-optimal. Hillclimbing is a search heuristic based on "greedy" techniques. This method attempts to find a maximum in the performance index by tracing out a path over time with an increasing index. It works well for convex spaces, but for other systems, the solver is likely to become trapped in a local optimum, and never find the global optimum. Genetic algorithms are a variant on the basic hillclimbing strategy. By their design, they are less susceptible to being trapped in a local maxima. The basic concept used is essentially motivated by an analogy with biological systems. In particular the process of natural selection is simulated with the solutions, so that over time, the quality of the solution increases.

In this paper, a genetic algorithm is used in the design of a scheduler for the crane operations. Our decision to use this method is based on an analysis of existing methods, which leads to the conclusion that genetic algorithms have reasonable computational time and can converge to a globally acceptable solution.

The outline of the paper is as follows. Section 2 gives the process description and poses the problem of crane scheduling as a discrete event optimisation. Section 3 discusses the genetic algorithm. Section 4 gives the design of the scheduler of crane operation using the genetic algorithm. Section 5 shows simulation results.

2. CRANE OPERATION IN THE COPPER SMELTER

The process under study is a section of the San Manuel copper smelter. This smelter uses a flash furnace and peirce-smith copper converters. After minimg and concentration, copper concentrate is supplied at 30% copper and the smelting process purifies this to 99.8% copper. Figure 1 shows the layout of the converter aisle in the smelter. The flash furnace smelts the copper concentrate at about 1200°C, where it separates into two liquid phases. The slag which contains silica, alumina and iron oxide is discarded, while the matte which contains copper, sulphur and unoxidised iron is treated by the converter. The matte is tapped out of the flash furnace at about 68% copper.

There are four bridge cranes available for transportation in the converter aisle. One of these four is kept in maintenance or as a reserve in a bay at the end of the aisle. The three remaining cranes are the main transporters in the aisle. Each crane may be used to lift one ladle at a time. As the cranes operate on one set of tracks, they cannot pass under normal operating conditions. This imposes the major constraint on the process operation

The crane operation may be regarded as a discrete event system. The state of each crane, i, may be expressed by its position, P_{Ci} and its configuration C_{Ci}. The restrictions on the positions of the three cranes $0 \leq i \leq 2$ in operation are:

$$0 \leq P_{C0}, \quad P_{Ci} \leq P_{Ci+1} - d_{min}, \quad P_{C2} \leq P_{Cmax}$$

where d_{min} is the minimum distance between two cranes and P_{Cmax} is the length of crane track.

A typical crane movement involves three stages:

(1) Fill a ladle with material.
(2) Move the ladle to a new location.
(3) Empty the ladle.

In situations where one crane movement conflicts with another, some of these stages need to be delayed. A major part of the guidance system is to choose which crane is to be used in each job and which movements are to be delayed in the case of conflict.

3. THE GENETIC ALGORITHM

Genetic algorithms are used to perform a search which is similar in nature to hillclimbing. The benefit of this approach is that it is more suited to non-convex spaces. (Bolc and Cytowski, 1992, Bloc 92) includes an introduction to genetic algorithms, while (Michalewicz, 1994, Michalewicz 94) and (Fogel, 1995, Fogel 95) also contain more detailed material.

The only interaction between the genetic algorithm and the problem to be solved is through the objective function. The objective function provides a measure of the quality of a potential solution.

During the search, many potential solutions are evaluated. Each of these is encoded in a vector, which is termed a chromosome. The chromosome may take on any form, but classically it consists of a string of binary digits. With some modifications to the basic genetic scheme, it is possible to have a vector of integer or real numbers to represent the solution. A genetic population containing a number of solutions is maintained during the search. On each iteration, some or all of these solutions are replaced by newly generated ones.

After each chromosome is generated, it is evaluated for fitness. This involves decoding the chromosome into a solution vector and then calculating the objective function for it. The fitness of a chromosome to reproduce is based on the value of the objective function. Each of the chromosomes x_i, is assigned a probability p_i of reproduction. The traditional method is to use a technique called "roulette wheel selection". In this method, each chromosome in the population is allocated a sector in a circle. The area of the sector

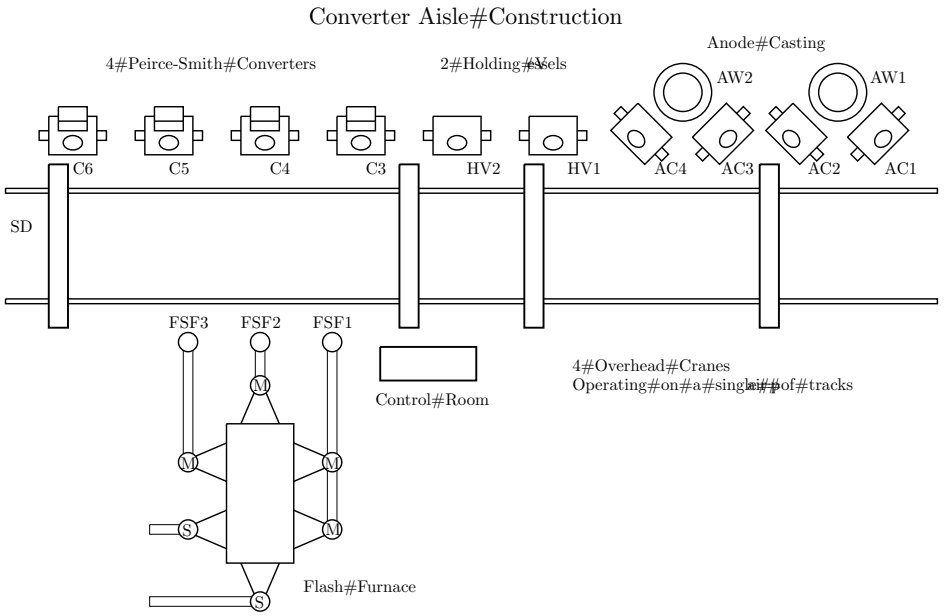

Fig. 1. Diagram of the converter aisle

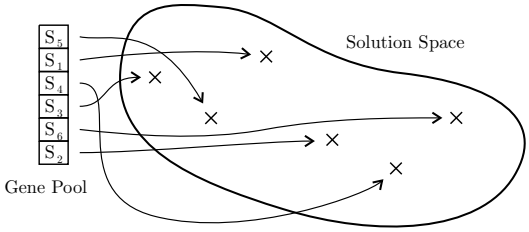

Fig. 2. The genetic population is represented by a table of chromosomes

is proportional to the probability p_i, and weighted so that $\sum p_i = 1$. Figure 3 shows the geometry of the roulette wheel. In order to choose a parent, a point is

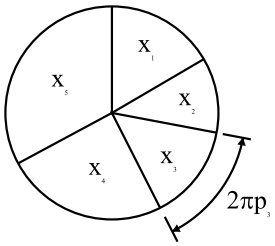

Fig. 3. Roulette wheel method of selecting parent chromosomes.

chosen at random on the circumference of the circle. The chromosome corresponding to that point is used as the parent.

New chromosomes are generated by selecting parents and applying "genetic operators" on them to obtain a new chromosome. The two most popular operators are crossover and bit mutation. The Crossover operator involves cutting each of the parent chromosomes into two sections and swapping a section of genes between the two chromosomes. This results in two new child chromosomes, as shown in Figure 4.

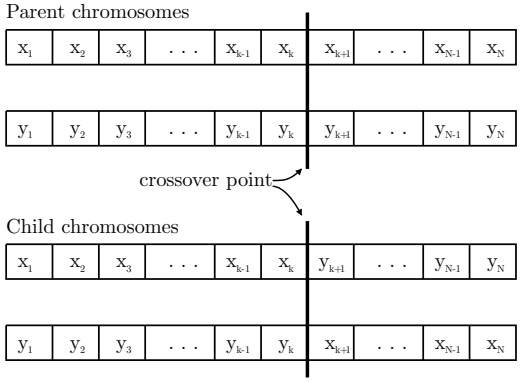

Fig. 4. Crossover operation on a pair of parent chromosomes

The mutation operation is simply a random modification of genes in the chromosome. This is needed to generate new genetic "material" that was not present at the start of the search. In the case of binary genes, the mutation is simply a random inversion of a bit in the chromosome.

In a traditional genetic algorithm, each iteration involves replacing all of the existing generation of chromosomes with a new generation of children. For a population of P chromosomes, the following steps are executed $\frac{P}{2}$ times.

(1) Select two chromosomes from the population using the roulette wheel method.
(2) with probability P_c apply a crossover operation at a random location in the vector.
(3) with probability P_m apply a mutation to the new chromosomes.

P_c is usually high (between 0.5 and 1), while P_m is generally small (less than 1%). At the end of this procedure, the old genetic population is replaced by the new population of P chromosomes. Many variants of this scheme are possible. In particular, some methods

specifically retain the good chromosomes from one generation to the next to ensure that the quality of the best solution never decreases.

The above process is repeated for as many iterations as are necessary. Depending on the search requirements, it will be stopped either once the solution becomes good enough, or the amount of available computation time has elapsed.

4. SCHEDULER DESIGN

4.1 *Process Representation*

The genetic algorithm scheduler is based on a simplified model of plant operations. The required crane movements are represented by a directed graph showing the order of the tasks to be accomplished. The graph used is an extension of the network flow graph used in operations research. The nodes on this graph represent each "operation" that occurs. An operation is a simple activity that occurs at one position in the aisle, such as skimming the contents of a converter into a ladle. An example graph is shown in Figure 5. In the example, the nodes **A** and **B** are used to indicate

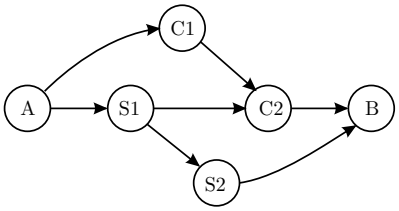

Fig. 5. An example operation flow graph (OFG) for a skim followed by a charge

the start and end respectively and do not map to a physical process. **S1** represents the skimming of the converter into a ladle, while **S2** is the emptying of the ladle at the slag dump. The link between these two nodes indicates that the skim must occur first. After the converter is skimmed, it is ready to receive copper matte from a different ladle. This is represented by the link from **S1** to **C2**, where **C2** is the charging of the converter with matte. Prior to the charging of the converter, the ladle must be filled with matte. This operation is represented by **C1**. An important feature of the graph is that it allows the filling of the ladle (**C1**) to occur concurrently with the skim operation(**S1**).

4.2 *Crane Movement Representation*

A graphical format has been developed to represent the crane movements within the plant. The movements are plotted as the crane position against time. Possible crane movements for the operations specified in Figure 5 is shown in Figure 6. In this example, two crane movements are represented, with crane 1 performing a skim and crane 2 the charge operation. Each of the crane movements contains two horizontal

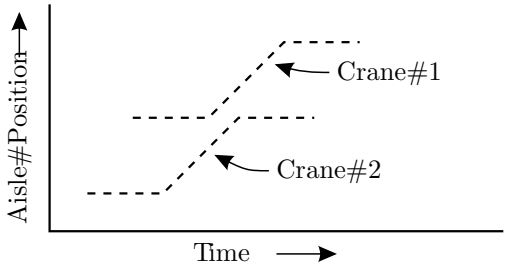

Fig. 6. Example plot of crane movements.

sections, which indicate the periods where the crane is stopped performing an operation. Between these periods, there is a line of constant slope representing a crane moving at constant speed. For this example, the horizontal sections map to the nodes of Figure 5 and the interconnecting sections to the edges. To reduce the complexity of these graphs, the positions of the cranes are not plotted when they are not servicing a job.

The above example is for the case where there is no conflict in the desired crane movements. The main object of the crane scheduler is to resolve conflicting crane service requests. Figure 7 illustrates the basic case where two movements conflict. The conflict

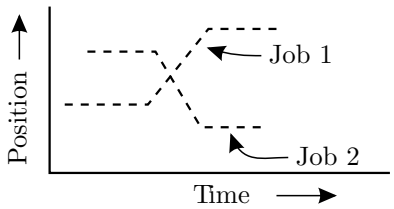

Fig. 7. Example of a conflict.

occurs when the lines cross, indicating that the cranes will collide if they follow the specified path. In this case, some of the operations need to be delayed in order to generate a realisable schedule. Some of the ways in which the conflicts may be resolved are shown in Figure 8. In cases **a** and **b**, one of the jobs is delayed in its entirety until the other one has finished using that section of track. An alternate approach is shown in cases **c** and **d**, where one of the cranes is moved to a location where it does not interfere with the movement of the other crane. In addition to these cases, different crane movements result when an alternative crane allocation is chosen.

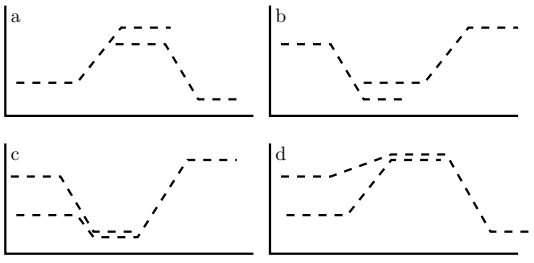

Fig. 8. Possible ways of resolving the conflict.

4.3 *The Solution Vector*

The solution vector must contain all of the information required to build a deterministic schedule. There are two sets of information required:

(1) The allocation of cranes to jobs.
(2) The order of job attendance when there are crane conflicts.

The first of these is a simple mapping, but the second is more difficult. In the simple case above, this involves choosing between the four cases in Figure 8. The vector adopted uses a concept of "relative priority". For each node in the network flow graph (e.g. Figure 5), the crane and a relative priority are specified. Any conflicts are resolved by supplying the resources to the task with the higher priority, while the other waits for availability.

One feature of this solution vector is that the different operations involved in a single crane movement may have different priorities. As a consequence, one crane movement may be interrupted by separate operation of an intermediate priority. This is necessary to describe cases such as **c** and **d** in Figure 8.

4.4 *Simulation Method*

The role of the simulator is to generate a schedule of crane movements(e.g. Figure 6), based on an operation flow graph (e.g. Figure 5) and a solution vector). Note that in all of these representations, both position and time remain as continuous variables. This simulation differs from conventional discrete event simulators as it does not grid the continuous variables into a series of discrete states. Instead, a set of continuous variables are generated to represent the solution. Specifically, the solution consists of the vertices of the crane movements as shown in Figure 8. The advantage of this approach is that the solution is simpler and more meaningful.

The simulation is achieved by the application of the basic algorithm below.

Iterate through each operation in order of priority
(a) find the earliest time it can be scheduled
(b) allocate the required crane resources to this operation

At each step in the algorithm, an additional segment of the crane movement graph is constructed. In order to calculate the earliest time in (a), it is necessary to consider both the completion time of the previous operations, and the availability of crane resources.

4.5 *Genetic Algorithm Implementation*

The crane scheduler combines the genetic algorithm described in Section 3 with the simulator described above. The performance measure chosen was "the expected length of time required to perform a fixed set of operations". Evaluation of this measure is achieved by simulating the given set of operations and finding the completing time of the last operation

The chromosome format is derived directly from the solution vector. Instead of using a binary string to represent the genetic structure, the parameters are left in their natural format. The crane allocation is retained as an integer between 1 and 3, and the relative priority as a floating point number. The only effect of this on the genetic algorithm is that mutation operators need to be developed. For the crane allocation genes, a mutation results in new crane being chosen at random for the given operation. When a mutation occurs on a priority gene, a random displacement is added to to the existing priority.

5. SIMULATION RESULTS

The genetic algorithm scheduler was simulated over a typical set of smelter operations. The length of process time to be simulated was chosen to be 3 hours. During 3 hours process time, approximately 60 operations occur. In the simulation the 60 operations are scheduled to give a total of 120 genes in the algorithm. Apart from increasing number of genes, the algorithm adopted is the same as the one given in the simple example above. The search with 120 genes, and a genetic population of 64 chromosomes, requires approximately 100ms per iteration on a Pentium PC. Each iteration involves the evaluation of a new genetic population which, in this case, requires the simulation of 64 solution vectors.

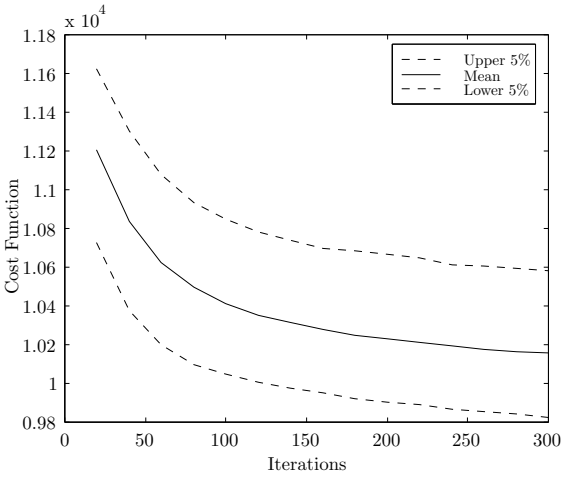

Fig. 9. Performance of the Genetic Algorithm as a function of search length.

The dependence of the solution quality on the number of iterations is shown in Figure 9. This graph was obtained by executing the optimisation routine 2500 times and averaging the performance measure across these trials. The two bounds shown in the figure enclose 90% of the results in the trials. A further 5% performed better than the trace marked "Lower 5%". If the transport movement constraints are ignored, the theoretical minimum for this cost function over the

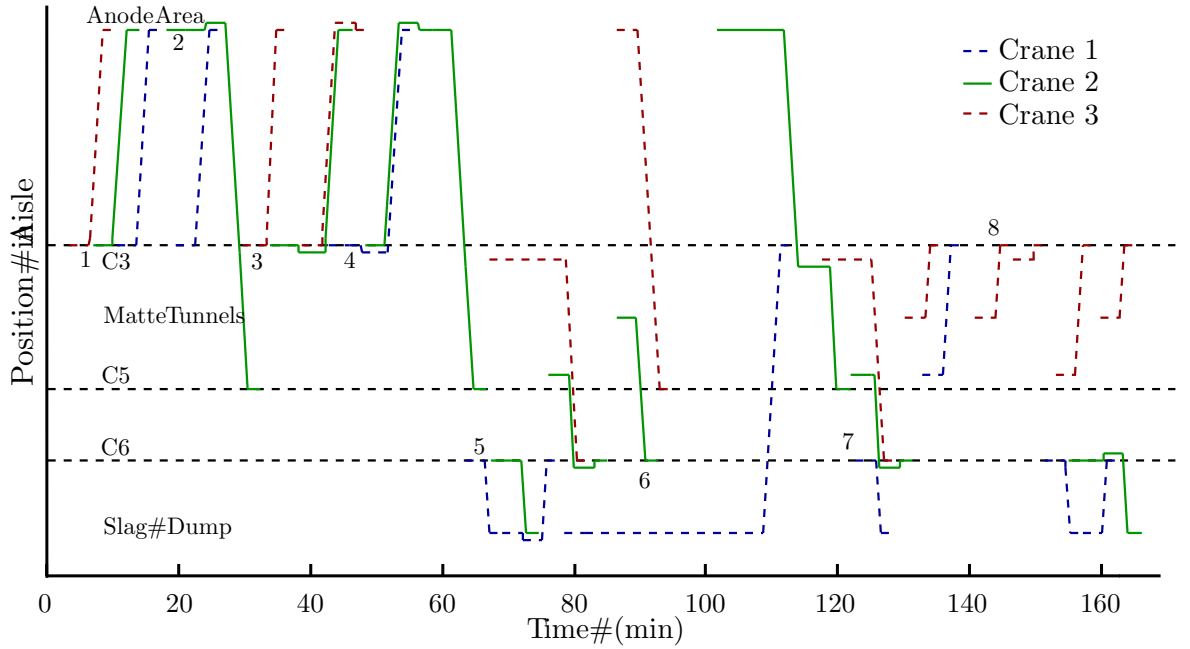

Fig. 10. Crane movements over time scheduled by the genetic algorithm

given set operations is 8380. After extensive simulation trials, the best measure achieved while including the constraints is 9482.

Figure 10 shows a resulting schedule after 300 iterations of the algorithm. This plot shows the position of the cranes with respect to time. Note that the trajectories of the cranes are not continuous. Between the line segments the crane is not in use. The numbers on the graph denote points of interest in crane movements which are explained as follows.

(1) Initially all three cranes are used in the transfer operation from converter number three to the anode area. During this time, the other converters do not require crane attendance.
(2) Crane 2 is used to pick up anode scrap for converter 5. This involves interrupting the transfer process.
(3) While crane two is attending the scrap job, crane three is used to initiate another transfer. By necessity, crane one is idle at this time.
(4) The final five movements of the transfer process for converter 3 appear to be close to optimal as the converter is always involved in a skim operation.
(5) Only two cranes are used here, but the third is being used to minimise later delays.
(6) There is a delay in obtaining the third ladle of matte. This appears unavoidable, however, due other conflicting requirements.
(7) The second skim of converter six exhibits good coordination. All three cranes are used for skim and charge operations. Crane three has been unloaded first to allow it to go on to service converter three.
(8) The charging of converter three is inefficient. This is because these operations are not on the critical path, so any improvement in scheduling is not reflected in the performance index. This problem may be overcome through the choice of performance index.

6. CONCLUSION

This paper described the application of scheduling techniques to an industrial process. Graph techniques were used to model and simulate the physical process, while a genetic algorithm was used to perform the optimisation. The final result was able to generate useful schedules within acceptable computation time.

7. REFERENCES

Bolc, L. and J. Cytowski (1992). *Search Methods for Artificial Intelligence*. Academic Press. London.

Fogel, D. (1995). *Evolutionary Computation*. IEEE Press. New York.

Goodwin, G.C., S. Graebe and M. Salgado (1998). *Principles of Control System Design*. Book in Preparation.

Hillier and Lieberman (1967). *Introduction to Operations Research*. 3rd ed.. Holden-Day Inc.. Oakland, California.

Michalewicz, Z. (1994). *Genetic Algorithms + Data Strctures = Evolution Programs*. 2nd ed.. Springer-Verlag.

Pepyne, D.L., D.P. Looze, C.G. Cassandras and T.E. Djaferis (1996). Application of q-learning to elevator dispatching. In: *13th IFAC World Congress*. Vol. J. San Francisco, CA. pp. 317–322.

Ramadge, P.J.G. and W.M. Wonham (1989). The control of discrete event systems. In: *Proceedings of the IEEE*. Vol. 77. pp. 81–98.