Higgins, A., Kozan, E. and Ferreira, L. (1997). "Heuristic techniques for train scheduling", Journal of Heuristics, Vol 3, 43 - 62

Heuristic Techniques for Single Line Train Scheduling

A. HIGGINS[1]
Ph: (07) 864 1291
Fax: (07) 864 2310
E-Mail: A.Higgins@qut.edu.au

E. KOZAN[1]
Ph: (07) 864 1029
Fax: (07) 864 2310
E-Mail: E.Kozan@qut.edu.au

L.FERREIRA[2]
Ph: (07) 864 1542
Fax: (07) 864 1515
E-Mail: L.Ferreira@qut.edu.au

[1] School of Mathematics
[2] School of Civil Engineering

Queensland University of Technology, PO Box 2434, QLD 4001, Australia

ABSTRACT

Optimising a train schedule on a single line track is known to be NP-Hard with respect to the number of conflicts in the schedule. This makes it difficult to determine optimum solutions to real life problems in reasonable time and raises the need for good heuristic techniques. The heuristics applied and compared in this paper are a local search heuristic with an improved neighbourhood structure, genetic algorithms, tabu search and two hybrid algorithms. When no time constraints are enforced on solution time, the genetic and hybrid algorithms were within five percent of the optimal solution for at least ninety percent of the test problems.

Key words: Train Scheduling, Local Search, Tabu Search, Genetic Algorithm, Hybrid Algorithm

1 Introduction

In Australia, the average length of haul for interstate freight movements is around 1500 kilometres with the bulk of this network consisting of single line track. The train dispatching function outside urban areas is currently performed using mainly manual methods. Trains which operate over a single track and can only overtake and cross each other at specific locations referred to here as sidings. This type of operation is common throughout the world. The train controller resolves train conflicts on a time-distance graph using experience and knowledge of prevailing conditions.

The main objective for developing a train scheduling model is to provide the train controller with a tool which helps him/her to perform the train dispatching function. The model is not designed to replace the train controller. The operator' s experience and knowledge of local conditions, will continue to be used. However, with such models and solution techniques available, the train controller is able to quickly update a schedule as unplanned events occur.

The purpose of a single line train scheduling model in this paper is to resolve the train conflicts (overtaking as well as crossing) at the sidings in such a way so as to achieve some pre-defined objective. Typical examples of this type of model can be found in Kraay et al. (1991), Mills et al. (1991) and Higgins et al. (1996). When optimising a train schedule in terms of determining which siding each conflict should be resolved at, the computation complexity increases exponentially as the number of conflicts in the schedule increase (Cai and Goh 1994).

In practical applications, an optimal solution to the train scheduling problem is not always considered a very high priority, as it is usually too time consuming to find. In addition, the solution may not be able to be implemented due to unquantifiable constraints. This leads to a desire to develop heuristic techniques which will find near optimal solutions in a short space of time. In this paper, the local search heuristic (**LSH**), genetic algorithms (**GA**), tabu search (**TS**), and two hybrid algorithms (**HA1** and **HA2**) are applied to the train scheduling problem.

Cai and Goh (1994) used the greedy heuristic when the velocities of all trains travelling in same direction were assumed to be equal. Rules were included in the heuristic to cater for heavy traffic. In situations where several trains travel close together in one or both directions, a rule will determine the most efficient means to resolve the group of conflicts (at the local level). Either all inbound trains go first or all outbound trains go first.

A rounding heuristic put forward by Kraay et al. (1988), was used to separate out potentially good train schedules. For each feasible schedule, the heuristic obtains a total for the distance of each conflict to the corresponding unresolved conflict. The feasible schedules are ranked via the total distances. The arrival and departure times for the best few schedules are determined.

An efficient learning heuristic was proposed by Kraay (1993) which uses characteristics of previous train schedules to generate a new schedule. This approach is based on the cyclic nature of train schedules. The learning heuristic resolves conflicts by seeing how they are resolved in previous schedules. If the previous schedules are poor, the new schedule will also be poor. The LSH has been tried on the train scheduling problem by the same author as a comparison to a learning heuristic. Kraay (1993) claimed the LSH performs quite well, except for the occasional poor solution, although it completely out performed the learning heuristic. While the above heuristics are fast, the quality of the solution can be extremely far from optimal.

For the LSH and TS in this paper, an improved neighbourhood is proposed which allows the simultaneous shift of more than one conflict in each move. The neighbourhood assumes that when the position of one conflict is shifted, the rest of the conflicts for one of the trains involved is also shifted. The GA is applied using an efficient chromosome representation to save storage and improve convergence time. The improved neighbourhood is used when the mutation operator is performed. Two different hybrid algorithms are proposed, namely: one which uses the LSH as a new operator and; the second imbeds the TS into the crossover operator so as to conduct a search for suitable parents and crossover points.. Solution time of the heuristics, as well as achievement of optimality and diversification of solution from optimality, are compared.

2 The Model

In this section, the model as used in Higgins et al. (1996) is put forward. The model allows a combination of single and double line track segments.

2.1 Definitions

Some of the main terms used in this paper are defined below.

Conflict Delay:- Is the delay to a train if it must wait at a siding for another train to cross or overtake.

Minimum Headway:- The minimum length of time separating two trains on a single line track. This is usually determined by signals in the case when the trains are travelling in the same direction. When travelling in opposite directions, the minimum headway is determined by the time required for one train to clear the track segment sufficiently before the opposing train can enter it.

Planned Schedule:- Is generated at a tactical level (medium term planning) by rail planners to be used on a daily or weekly basis. The actual schedule will be the same as the planned schedule if no unforeseen events occur.

Resolving a Conflict:- Is when one of the trains involved in the conflict must be forced to the siding for the other train to cross or pass. This ensures safe operation (feasible schedule).

Siding:- A section of track which can be used for the crossing and passing of trains under single track operations. The terms 'crossing loop' or 'passing loop' are also used in some countries to describe such track sections. A train station on a single line track will usually contain a siding.

Train Conflict:- There are two situations, namely: when two trains approach each other on a single line track and the continuation of either or both trains' journey would not be possible; and when a faster train catches up a slower train travelling in the same direction.

2.2 Definition of Variables

The set of trains is given by $I=\{1,2,.....,m,m+1,.....,N\}$ for which inbound trains are numbered from 1 to m and outbound are from $m+1$ to N. The ordering of the trains in this set is considered in terms of earliest departure time from the origin station. The set of sidings is represented by $Q=\{1,2,....,NS\}$. Let $P = \{P_1, P_2\}$ where P_1 = set of single line track segments and P_2 = set of double line track segments. The variables used in the model are listed and described below.

The integer decision variables for determining which train traverses a track segment first (and the determination of where conflicts are resolved) are given by:

$$A_{i,j,p} = \begin{cases} 1 & \text{if inbound train } i \le m \text{ traverses track segment } p \in P \text{ before inbound train } j \le m \\ 0 & \text{otherwise} \end{cases}$$

$$B_{i,j,p} = \begin{cases} 1 & \text{if inbound train } i \le m \text{ traverses track segment } p \in P_1 \text{ before outbound train } j > m \\ 0 & \text{otherwise} \end{cases}$$

$$C_{i,j,p} = \begin{cases} 1 & \text{if outbound train } i > m \text{ traverses track segment } p \in P \text{ before outbound train } j > m \\ 0 & \text{otherwise} \end{cases}$$

The correct setting of these integer decision variables will ensure the safe operation of conflicting trains. The arrival and departure time continuous decision variables are as follows:

XA_q^i = arrival time of train $i \in I$ at siding $q \in Q$
XD_q^i = departure time of train $i \in I$ from siding $q \in Q$
$XD_{O_i}^i$ = departure time of train $i \in I$ from its origin station
$XA_{D_i}^i$ = arrival time of train $i \in I$ at its destination station

The required input parameters for the model are defined as follows:

O_i = origin station of train $i \in I$
D_i = destination station of train $i \in I$
$h_p^{i,j}$ = minimum headway between trains $i, j \in I$ on segment $p \in P$
d_p = length of segment $p \in P$
$YD_{O_i}^i$ = earliest departure time of train $i \in I$ from its origin station
$\underline{v}_p^i$ = minimum allowable average velocity of train $i \in I$ on segment $p \in P$

$\bar{v}_p^i$ = maximum achievable average velocity of train $i \in I$ on segment $p \in P$

W^i = priority of train $i \in I$ (highest for passenger trains)

S_q^i = scheduled stop time for train $i \in I$ at siding $q \in Q$.

An illustration of a single line track used for the single line train scheduling model is provided in Figure 1. Here, track segment $p - 2 \in P_2$ is double line.

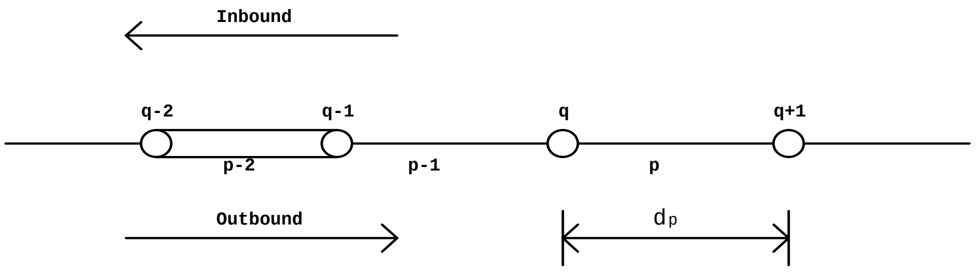

Figure 1: Rail corridor: single and double track schematic representation

2.3 Assumptions and Inputs

The following assumptions are made with regard to the model used in this paper:

- Crossing and overtaking can occur at any siding or double line track segment;
- Trains can follow each other on a track segment provided a minimum headway (safety zone) is maintained;
- For double line track segments, it is assumed one lane is allocated for inbound trains and one lane is allocated for outbound trains. Usually, signal points will be set up this way;
- Scheduled stops are permitted at any intermediate siding for any train; and
- Only one train can occupy a siding at any one time. Some times two short trains can fit on a siding but will not be allowed in this paper.

The model will require the following data as input to the model:

- The earliest departure times of the trains;
- The priorities of each train. These are determined by several factors such as the type of train, customer contract agreements and train load;
- The upper and lower average velocity limits for each train (which are dependent on the physical characteristics of the track segment and the train); and
- The length of time of any scheduled train stops. These stops may include loading/unloading, refuelling and crew changes.

2.4 The Objective Function and Constraints

The objective function will depend upon whether the train plan is being constructed or if the trains are being rescheduled. For the purposes of this paper, it is assumed the train plan is being developed and that the expected arrival times will not be known in advance. The objective function involving the minimisation of total conflict delay takes the following form:

$$\text{Min} \quad Z = \sum_{i \in I} W^i * (XA^i_{D_i} - YD^i_{O_i}) \tag{1}$$

The model is subject to various constraints to ensure safe operation, enforce speed restrictions and permit stops. The following and overtake constraints for outbound trains $i, j \in I$ are as follows:

$$\left.\begin{aligned} M*C_{i,j,p} + XA^i_{q+1} \geq XA^j_{q+1} + h^{i,j}_p \\ M*C_{i,j,p} + XD^i_q \geq XD^j_q + h^{i,j}_p \end{aligned}\right\} \forall\ p \in P \ \text{ and } i, j > m \tag{2}$$

$$\left.\begin{aligned} M*(1\text{-}\, C_{i,j,p}) + XA^j_{q+1} \geq XA^i_{q+1} + h^{i,j}_p \\ M*(1\text{-}\, C_{i,j,p}) + XD^j_q \geq XD^i_q + h^{i,j}_p \end{aligned}\right\} \forall\ p \in P \ \text{ and } i, j > m \tag{3}$$

and for inbound trains $i, j \in I$:

$$\left.\begin{aligned} M* A_{i,j,p} + XA^i_q \geq XA^j_q + h^{i,j}_p \\ M* A_{i,j,p} + XD^i_{q+1} \geq XD^j_{q+1} + h^{i,j}_p \end{aligned}\right\} \forall\ p \in P \ \text{ and } i, j \leq m \tag{4}$$

$$\left.\begin{aligned} M*(1\text{-}\, A_{i,j,p}) + XA^j_q \geq XA^i_q + h^{i,j}_p \\ M*(1\text{-}\, A_{,i,j,p}) + XD^j_{q+1} \geq XD^i_{q+1} + h^{i,j}_p \end{aligned}\right\} \forall\ p \in P_1 \ \text{ and } i, j \leq m \tag{5}$$

where constant M is chosen large enough so that both equations in each crossing and overtake constraint are satisfied. For the solution techniques in this paper, these constraints are only enforced whenever a conflict occurs.

The first inequality of constraint 2 implies that if outbound train $j \in I$ is to travel along track segment $p \in P$ first, then outbound train $i \in I$ must depart siding $q \in Q$ after train $j \in I$ plus the minimum headway between the two trains. The second inequality implies that if outbound train $j \in I$ is to travel along track segment $p \in P$ first, train $i \in I$ must arrive at siding $q+1 \in Q$ after train $j \in I$ plus the minimum headway. Constraint 3 is similar except train $i \in I$ goes first. Constraints 4 and 5 are the same as constraints 2 and 3, except for inbound trains. The constraints for when two trains approach each other are:

$$\left.\begin{aligned} h^{i,j}_p + XA^j_{q+1} \leq XD^i_{q+1} + M*B_{i,j,p} \\ h^{i,j}_p + XA^i_q \leq XD^j_q + M*(1\text{-}\, B_{i,j,p}) \end{aligned}\right\} \forall p \in P_1,\ (i \leq m, j > m) \tag{6}$$

The first inequality of constraint 6 implies that if outbound train $j \in I$ travels along track segment $p \in P_1$ first, inbound train $i \in I$ must depart siding $q+1 \in Q$ after train $j \in I$ arrives plus a minimum headway. This headway includes the time lost due to accelerating after the delay at the siding. The second part of the equation applies when inbound train $i \in I$ goes first.

Given the upper and lower average achievable velocities for each train on each segment, the upper and lower limits for travel time of train $i \in I$ on segment $p \in P$ are given by:

$$\begin{aligned} &\frac{d_p}{\bar{v}^i{}_p} \le XA^i_{q+1} - XD^i_q \le \frac{d_p}{\underline{v}^i{}_p} \qquad i > m,\ p \in P \\ &\frac{d_p}{\bar{v}^i{}_p} \le XA^i_q - XD^i_{q+1} \le \frac{d_p}{\underline{v}^i{}_p} \qquad i \le m,\ p \in P \end{aligned} \tag{7}$$

The maximum average achievable velocity takes into account the velocity of the train before it enters track segment $p \in P$. This velocity may be less if the track segment prior to the current one has a slow upper average achievable velocity. To stop any train from departing before its earliest departure time and any train departing an intermediate siding before it arrives, the following constraints are included:

$$\left.\begin{aligned} &XD^i_{O_i} \ge YD^i_{O_i} \\ &XA^i_q + S^i_q \le XD^i_q \end{aligned}\right\} \forall\ i \in I,\ q \in Q \tag{8}$$

The objective is to minimise the objective function (equation 1) subject to constraints 2 - 8. Although not included in the formulation, the allowance of multiple sidings in parallel is enforced during the solution procedure, without increasing computation time.

While the arrival and departure times are decision variables, their values are directly dependent on the integer decision variables when the objective is the minimisation of total conflict delay (equation 1). When the integer decision variables are fixed (i.e. conflicts are at fixed positions), the arrival and departure times are determined by constructing the schedule (Higgins et al. 1996).

3 Local Search Heuristic (LSH)

In this section a brief description of the LSH (similar to that used by Kraay (1993)) is given. Before the heuristic is applied, a good feasible initial solution (resolved train schedule) is obtained by using the first upper bound obtained from the branch and bound technique (Higgins et al. 1996).

A train schedule is represented by $CR = \{CR_1, CR_2, \ldots\ldots, CR_{CO}\}$, where $CR_c \in CR$ is the c^{th} conflict as it appears in time. At the c^{th} conflict CR_c = {the delayed train; the train with right of way; and the track segment which these two trains intersect}. The value *CO* represents the total number of train conflicts in the schedule.

The LSH is based upon repeatedly replacing a current solution with an improved neighbouring solution. A move transforms the current solution to the neighbouring one. The neighbourhood for shifting the position of a resolved conflict CR_c (as used by Kraay 1993) is described by following possible moves (where 1 and 2 refer to the case of a crossing conflict).

1. If the inbound train in CR_c is delayed at siding $q \in Q$, test for improvement by delaying it at siding $q+1 \in Q$, or the outbound train at siding $q-1 \in Q$;

2. If the outbound train in CR_c is delayed at siding $q \in Q$, test for improvement by delaying it at siding $q-1 \in Q$, or the inbound train at siding $q+1 \in Q$;

3. If CR_c is an overtake conflict at siding $q \in Q$, test for improvement by shifting the conflict to sidings $q+1 \in Q$ and $q-1 \in Q$.

The main aim for applying the above possible moves is to try to improve the train schedule in terms of reducing total conflict delay. While moving a conflict to a different siding may increase the conflict delay for that particular conflict, the total conflict delay for the whole schedule may be reduced. The pseudocode for the LSH when it is applied to the single line train scheduling problem is described by Algorithm 1 below.

Algorithm 1

S: Indicator for when a move improves the solution.
C: The conflict in *CR* currently being tested for improvement.

```
0. Generate an initial resolved schedule and let $CR = \{CR_1, CR_2, ......., CR_{CO}\}$ be the set
   of resolved conflicts. Set $C$=0, $S$=0. Obtain a value for objective $Z$ from $CR$.

1. Set $C$=$C$+1.
   IF $C > CO$ and $S$ = 0 THEN
        Stop
   ELSEIF $C > CO$ and $S$ = 1
        Set $C$ =0, $S$ =0
        Go to Step 1.
   END {IF}

2. FOR each of the moves which can be applied to conflict $CR_C$:
        Obtain $CR_C^*$ and $Z^*$ by applying the move to $CR_C$. Resolve any new conflicts and
        update $CR^*$.
   END {FOR}
   IF $Z^* < Z$ THEN
        Let $Z = Z^*$, $CR_C = CR_C^*$, $S$ =1.
   END {IF}

3. Go to Step 1.
```

The LSH (Algorithm 1) works be taking a conflict from the resolved train schedule (keeping all other conflicts fixed) and try shifting it to one of the two neighbouring sidings (which is the neighbourhood). If the solution is improved after moving the conflict, it is kept, otherwise the old solution remains. The LSH will try moving the position of each conflict in the train schedule (to one of the two neighbouring sidings) starting from the first conflict (and re-starting at the first conflict after the last) until there is no further improvement.

3.1 Defining an Improved Neighbourhood

The main disadvantage of the LSH with the current neighbourhood, is that a move only shifts the position of one conflict at a time. Situations do occur where shifting the position of two or more conflicts at the same time is the only way for the solution to be improved. This situation is not just confined to the train scheduling problem but occurs with most integer programming problems. The computation complexity increases when considering a move which shifts the position of more than one conflict simultaneously. The size of the neighbourhood for which a move shifts the position of one conflict is of order CO, since each conflict can be shifted to one of the neighbouring sidings. When testing a move involving a simultaneous shift of n conflicts, the number of possible moves in the neighbourhood is up to $2^n * \frac{CO!}{(CO-n)!*n!}$, provided each conflict can be shifted to one of the two feasible neighbouring sidings.

To reduce the size of the neighbourhood, only moves which are more likely to improve the solution are tested. For a move which shifts the position of more than one conflict, only conflicts which are adjacent to one another and have one train common to all conflicts are considered in a move. Referring to Figure 2, a move which shifts the position of both conflicts A and E will be tested as well as a move which shifts the position of conflicts C, D and E. In the case of conflicts A and E, the outcome of shifting the position of conflict A will directly depend on where conflict E is shifted. This is further illustrated in Figure 3. The solution will only improve if both conflicts (between trains 1 and 3, and trains 2 and 3) are shifted to sidings $q \in Q$ and $q+2 \in Q$ respectively (as represented by the path of the dotted line).

When the position of several conflicts are shifted in a move, the outcome of shifting each conflict will depend on where each of the other conflicts are shifted. Generally, all conflicts would have to be shifted forward or back for the solution to improve.

q+2

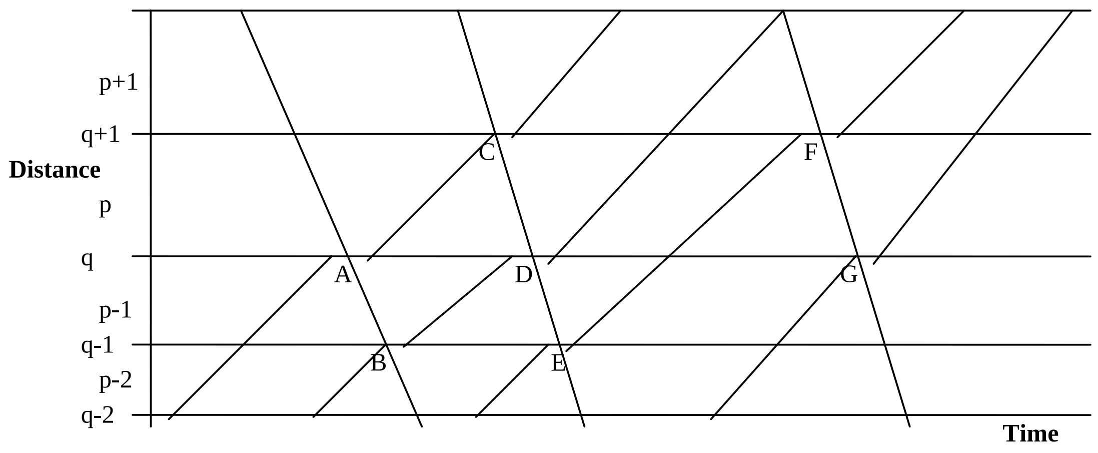

Figure 2: Train graph to illustrate the dependence when shifting conflicts

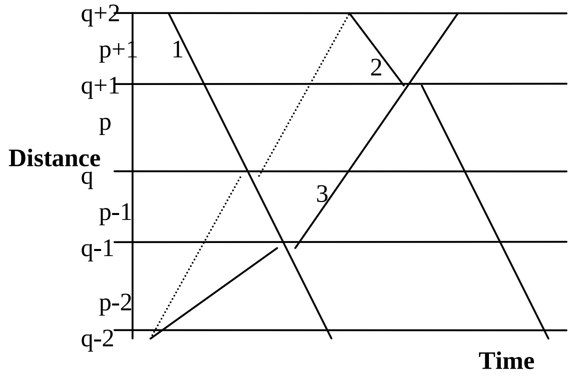

Figure 3: Illustration of a move which shifts the position of two conflicts

The new neighbourhood is defined as follows: When considering a particular conflict in the train schedule (say conflict *c* which is between inbound train $i \in I$ and outbound train $j \in I$), there are four possible moves. Let $DR \subset CR$ be the set of remaining conflicts for train $i \in I$ from conflict *c*. The four possible moves are:

1. For each conflict *d*, $DR_d \in DR$, obtain the track segment $p \in P$ from DR_d. Change this track segment to $p-1 \in P$.

2. For each conflict *d*, $DR_d \in DR$, obtain the track segment $p \in P$ from DR_d. Change this track segment to $p+1 \in P$.

Moves **3.** and **4.** are the same as **1.** and **2.** except they are applied to train $j \in I$. Shifting the position of an overtake conflict is the reverse of a crossing conflict when the train (for which the remaining conflicts are to be shifted) is the faster train.

4 Genetic Algorithm Approach

GA are search techniques inspired by genetic theories, which have been used to find good solutions to optimisation problems without early convergence to local optima (Goldberg 1989; and Davis 1987 and 1991). In GA, a population is a collection of train schedule solutions for which each is represented by a chromosome made up of a set of genes. The GA is based upon the assumption that fit solutions are more likely to reproduce and the characteristics of the fit solutions (called parents at the time of reproduction) will be transferred to the offspring. After successive generations, the population will tend to grow fitter and weak solutions will die off. The train schedule solutions in the population will tend to their optimal fitness after a large number of generations.

Representation as a Chromosome

When representing a train schedule solution, it is important to keep the chromosome representation as short as possible so as to allow faster convergence and not sacrifice solution quality. Initially, the most obvious representation is to have each of the binary conflict decision variables as a gene in the chromosome. This will result in a chromosome of considerable length and most of the genes (variables) do not provide any description to the train schedule solution. A better representation is to use each conflict in the train schedule solution as a gene. Each gene will contain three attributes, namely: the train delayed; the train with right of way; and the track segment which the conflict occurs. An example is demonstrated in Figure 4a which is the chromosome representation of the train schedule solution in Figure 4b. The first gene indicates that train 1 is delayed by train 4 on track segment 2.

	Gene 1	Gene 2	Gene 3	Gene 4	Gene 5	Gene 6
Parent	1,4,2	2,4,3	5,1,2	5,2,3	3,5,3	6,3,2

Figure 4a: Chromosome representation of train schedule (shown in Figure 4b)

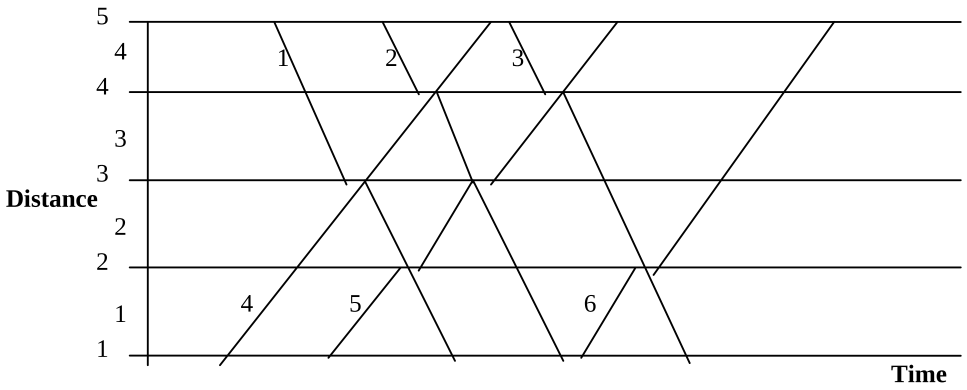

Figure 4b: Six conflict train schedule solution

This will thoroughly represent a train schedule solution with the minimum size chromosome and does not contain genes which have no practical determination of the fitness. However, this representation can cause some problems. The chromosome representations for the train schedule solutions may not all be of the same length. Some will have more resolved conflicts than others, depending on how the earlier conflicts are resolved. When a child train schedule solution is produced from two parent solutions, it may not represent a fully resolved schedule. The unresolved conflicts must be resolved by shifting to the nearest siding. These new resolved conflicts are added to the chromosome representation in the appropriate place, where the first conflict (gene) in the chromosome is the earliest conflict and the last conflict is the latest. Infeasible child solutions are replaced with one of the parents. Duplicated conflicts in a child solution can occur. These are due to conflicts in the parent solutions being at different positions in the chromosome representation. When a crossover is performed, the same conflict is copied from both parent solutions to the child solution. The duplicate conflicts in the child solution must be removed from the chromosome. Unlike chromosome representations for many other integer programming problems, each gene in the chromosome is completely dependent upon previous genes.

Crossover Operator

When two train schedule solutions (parents) in the population are selected to mate (probability of selection based on the fitness), genes from both solutions are used to make two offspring. A single point crossover is used for the train scheduling problem so as to keep the number of infeasible child train schedule solutions at a minimum. The crossover point is randomly selected, with the genes prior to that point being copied to one child solution, whilst those after that point are copied to the other. The same applies to the second parent, and what is produced are two child solutions in the next generation. Sometimes a parent selected for mating will not produce offspring but itself is transformed to the next generation.

Forms of Mutation

When two parents mate to produce offspring, mutation may also occur in the production of the offspring. Mutation is a random change of a gene. For example, if a gene can take the value of 0 or 1 and is equal to 0 when the child is produced, it will be changed to 1 when mutated. Although, the probability of a gene being mutated is usually low, it adds variation into the population. Since a good solution needs to be found in as few generations as possible, it is desirable to have a mutation which has a good chance of improving the solution. Just changing one conflict (gene) is not a good form of mutation in the train scheduling problem. As seen with the new neighbourhood, a solution is more subject to improvement if the rest of the conflicts for one of the trains in conflict are shifted as well. The form of mutation considered involves randomly choosing a move using the new neighbourhood.

Generation of Initial Population

A good initial population will increase the chances of finding a good solution. It is desirable to keep the initial population as small as possible so as to keep the number of generations small. When generating a train schedule solution of the initial population, there is a probability associated with the conflict being resolved at a siding. The probability that an inbound train $i \in I$ is delayed at siding $q+1 \in Q$ in the event of a conflict with outbound train $j \in I$ on track segment $p \in P$ is $PR_{i,j}^{p}$=B/(A+B), where distances A and B are illustrated in Figure 5. Here, the probability of delay is assumed proportional to the length of delay. An overtake situation is also illustrated in the same figure for which an outbound train $i \in I$ will be delayed by an outbound train $j \in I$ with probability $PR_{i,j}^{p}$=B/(A+B).

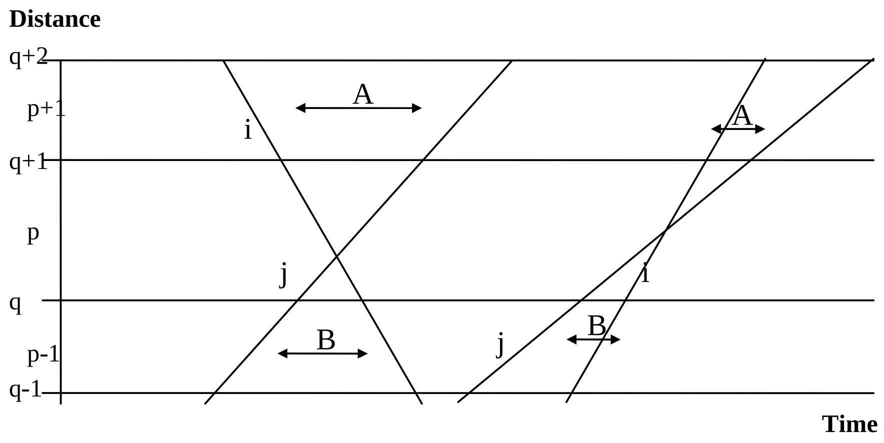

Figure 5: Illustration of conflict delay to train $i \in I$ as opposed to train $j \in I$

The algorithm for generating each train schedule solution in the initial population (of size *PO*) is as follows:

1. Obtain the unresolved schedule. This will make available a list of conflicts in the order which they occur.

2. Select the first unresolved conflict.

3. Determine the two trains $i, j \in I$ involved and the track segment $p \in P$ which it occurs.
Using a random number generator, delay train $i \in I$ with probability $PR_{i,j}^{p}$ and train $j \in I$ with probability $PR_{j,i}^{p}$.
After delaying one of these two trains, update the schedule and select the next unresolved conflict.

4. IF there are no more unresolved conflicts THEN Stop, otherwise Go to Step **3**.

Fitness Evaluation

Determination of the fitness evaluation function is important since it is not desirable to have strong solutions dominating the population too early, nor to have the population taking a long time to converge. After investigating several fitness evaluation functions, it was decided that the fitness was best calculated by the following:

- Let $C_{\max} = \max(Z_k)$ where Z_k is the objective function value for solution *k*.
- The fitness of solution *k* is $f_k = (C_{\max} - Z_k)^{\lambda}$.
- The probability of selection for solution *k* is $f_k \big/ \sum_k f_k$.

Having the fitness value as a quadratic function of the objective value ($\lambda = 2$) was found to give better results than a linear function ($\lambda = 1$), since it gives a higher probability of selection for the fitter solutions and a lower probability for the poorer ones. Having higher values of λ caused strong solutions to dominate the population too easily and thus premature convergence resulted.

Solution Procedure for GA (Algorithm 2)

The GA used to solve the single line train scheduling problem is as follows:

0. Generate initial population of size *PO*.

1. Determine fitness and probability of selection for each solution in the population

Repeat:
2.
Select *PO*/2 pairs of solutions based on their probability of selection.
Perform mutation on the pairs of selected solutions.
Reproduce and create offspring of the new generation from parents using a randomised single point crossover position on each selected pair of parents.
Remove any duplicated conflicts from each child solution.
If any child solution is infeasible, transfer one of the parents across as the child.
Resolve any unresolved conflicts in each child solution.
Obtain the fitness value and probability of selection for each child solution. The child solutions will be parents in the next generation.

Until: variance of fitness of solutions in the population is small enough so that no significant improvement in average fitness of population will occur

5 Tabu Search Technique

The TS heuristic (Glover 1990 and 1993) uses the local search principles whilst escaping local optima solutions. This is done by allowing up-hill (non improving) moves when no down-hill (improving) moves are available. When up-hill moves are allowed, a tabu list of length *L* must be kept to prevent the move from being reversed in the next *L* iterations. This is important to prevent cycling through sequences of solutions. When an up-hill move is accepted, the best move in the next iteration will be down-hill to the old solution. An aspiration criterion will override the tabu status if the move results in a better solution than any so far.

The next problem is to define the neighbourhood. A good neighbourhood structure will allow more meaningful moves to be performed and avoid generating infeasible solutions. The new neighbourhood structure defined in Section 3 is used. Its size is at most four times the number of conflicts. At each iteration, the neighbourhood (or a sample of it) is searched and the best move (if not tabu) is implemented, even if it results in a worse solution than the current one. Methods for reducing the amount of the neighbourhood searched involve taking random samples (Gendreau et al. 1991; and Semet and Tiallard 1993). A complete neighbourhood search will always produce better results than a partial neighbourhood. If good sampling techniques are used, the effects of a partial neighbourhood exploration can be reduced.

Diversification is used in TS to force the search into a different path, thus preventing a possibly poor path being searched for a long period of time. Diversification is performed after every *T* iterations, in which one of the best solutions so far is taken as the current solution and the tabu list length is changed randomly about its mean.

Moves which create infeasible solutions are discarded. If new unresolved conflicts are found after a move, they are resolved at the nearest feasible siding. The TS heuristic is terminated when the maximum number of iterations is exceeded.

Several different variations of the TS heuristic have been tried and tested (Higgins et al. 1995). One of the best variations (which is used in this paper) is to penalise moves (Srivastava and Chen 1993) so as to prevent long term repetition. The penalised objective function takes the following form:

$$Z^* = Z + fq_v^t * Z * CONST$$

where fq_v^t is the number of times move *v* has been performed in the TS up to the current iteration *t*. As a result of experimenting, a value of 0.01 was determined for *CONST*.

6 Hybrid Algorithms

The main objective for developing hybrid algorithms is to combine the advantages of two or more different heuristics. Although the GA in Section 7.1 tend to converge slower to a good solution, it was found to arrive at solutions which are almost unreachable using techniques such as TS and LSH which are based upon local search. However, the TS and LSH can quickly find a local improved solution. One of the most popular hybrid algorithms is to have a LSH as a new operator in the GA (Miller et al. 1993; and Nordstrom and Tufekci 1994). In these algorithms a LSH is applied to some or all of the population members after the mutation and crossover operators are performed. Since the GA is poor at improving a solution locally, the LSH complements the GA. However, an application of the LSH to all members of the population restricts diversity as well as making the algorithm very time consuming. Because of this, the LSH is applied to a small portion of the population (usually fitter solutions). The first hybrid algorithm proposed in this paper (HA1) is to apply the LSH (with improved neighbourhood) to the best five percent of the population, after the crossover operator is performed in Algorithm 2.

The second hybrid algorithm (HA2) incorporates the advantages of TS into the crossover operator. A similar strategy was carried out by Tan and Lim (1996) using simulated annealing. Instead of randomly matching up parents using a probability of selection, a search for suitable parent matchings is performed. The first parent is selected using the probability of selection (as before). Using the population and possible crossover positions as the neighbourhood, a search (of sample size *NSS*) is performed to find a suitable parent and crossover position which does not produce offspring of the next generation which are tabu. A child solution is tabu if it appears in the previous *PL* populations. The tabu status is over ridden if the fitness of the child solution is better than any so far in the current population; or it is has equal fitness to the best member of the current population and less than T_t of the current population (generation *t*) contains this member. Only a part of the new population (of size PD_t at generation *t*) is created using the TS method. The rest ($PO - PD_t$) is created using the normal crossover.

The crossover operator of Algorithm 2 is replaced as follows:

```
Perform normal crossover on ($PO - PD_t$)/2 pairs of solutions.
DO $PD_t$/2 times
    Pick first parent using probability of selection.
    DO NSS times
        Pick second parent using probability of selection.
        Randomly pick single point crossover position.
        Generate offspring and determine fitness of each (after removing
        duplicate conflicts and resolving unresolved conflicts).
    END {DO}
    Pick two fittest offspring which are not tabu (unless satisfying aspiration
    criteria) and add to next generation.
END {DO}
```

7 Comparison of Heuristics

The purpose of this analysis is to investigate which algorithms give the best solutions and with the least calculations. Problems sizes range between 15 trains and 50 trains or between 13 conflicts and 113 conflicts. The results of Table 1 show the average proportion increase (averaged over four train problem instances for each problem size) in total conflict delay over the optimal solution for each problem size. The optimum solution was calculated using the branch and bound technique (Higgins et al. 1996). Results for the GA, TS, HA1 and HA2 were averaged over 20 runs for each train problem instance. The standard deviations of the average proportion increase are shown in the parentheses. As a result of experimenting for GA, HA1 and HA2, the population size was set at 40, probability of crossover was 0.7 and probability of mutation was 0.01. The GA is considered to have converged when the standard deviation of the fitness of the solutions in the population falls below 0.01 hours. After experimenting with HA2, the parameters *NSS* and PD_t were set at 26 for all generations *t*, and the tabu list length, *PL*, was set to 3. For TS, the mean tabu list length, *L*, was set to 7, and can be changed (during diversification) to a value between 4 and 10 with uniform probability.

Referring to Table 1, the both hybrid algorithms completely outperformed the other heuristics with HA2 producing better results than HA1 on twenty seven out of the thirty two test problems. GA and TS heuristics produced better results than all versions of the LSH, with the GA results out-performing the TS on twenty six test problems. The LSH heuristics were inferior to GA and TS due to a lesser ability to escape from local optimal solutions. The LSH with the new neighbourhood gave better results than the original neighbourhood (Kraay 1993), especially for larger problem sizes.

Table 2 shows how often the arrival/ departure time sub-problem (integer variables fixed) was calculated for each heuristic. The number of calculations for the TS is the iteration limit multiplied by the neighbourhood sample size. As a result of experimenting, the iteration limit was set at 150, and the sample size of neighbourhood searched was set at 20 for problems with less than 30 trains and 30 for problems with more than 30 trains.

It is difficult to compare the number of calculations of TS to GA since convergence is defined differently. The number of calculations for GA is higher than both of the LSH but did not increase significantly with problem size. Using the LSH as a new genetic operator (HA1) increased the number of calculations by up to seven times. While HA2 required more calculations than HA1 for smaller size problems, the number of calculations did not increase that much for larger size problems. This is due to the imbedded TS searching a sample of the neighbourhood.

Overall, the HA2 was found to be a superior hybrid to HA1 as it found better solutions in a shorter space of time for larger size problems. The GA was superior to the TS when the TS was terminated at the same time as the completion of the GA. When the comparison is made with the LSH, the trade-off depends on whether the priority is with solution quality or solution time.

Table 1: Conflict delay of heuristic solutions (average proportion increase over optimal solution)

Heuristics Tested

No. Trains (No. Conflicts)	LSH (A1)	LSH (A2)	GA	TS	HA1	HA2
15 (13-23)	0.115	0.075	0.010 (0.002)	0.035 (0.007)	0.003 (0.003)	0.003 (0.003)
20 (20-28)	0.050	0.047	0.005 (0.002)	0.035 (0.007)	0.001 (0.001)	0.000 (0.000)
25 (25-35)	0.190	0.180	0.002 (0.000)	0.112 (0.015)	0.001 (0.000)	0.000 (0.000)
30 (49-52)	0.130	0.205	0.120 (0.034)	0.080 (0.009)	0.092 (0.012)	0.025 (0.008)
35 (55-67)	0.150	0.135	0.010 (0.012)	0.080 (0.015)	0.002 (0.001)	0.001 (0.000)
40 (79-81)	0.135	0.120	0.025 (0.014)	0.110 (0.006)	0.003 (0.001)	0.003 (0.001)
45 (91-95)	0.280	0.145	0.040 (0.015)	0.115 (0.012)	0.001 (0.000)	0.000 (0.000)
50 (103-113)	0.060	0.055	0.020 (0.010)	0.010 (0.005)	0.003 (0.001)	0.002 (0.001)

A1:	LSH with original neighbourhood.	HA1:	GA with LSH as new genetic operator.
A2:	LSH with improved neighbourhood.	HA2:	GA with TS imbedded in crossover operator.
GA:	Genetic algorithm.	TS:	Tabu search.

Table 2: Number of calculations of the arrival/departure time sub-problem

Heuristics Tested

Problem	LSH (A1)	LSH (A2)	GA	TS	HA1	HA2
15 (13-23)	50	81	540	3000	1055	1440
20 (20-28)	55	96	860	3000	1520	3350
25 (25-35)	101	148	990	3000	2365	3270
30 (49-52)	130	251	1300	3000	3870	4960
35 (55-67)	294	455	1805	3000	5490	6320
40 (79-81)	361	603	1950	4500	11610	8290
45 (91-95)	421	700	1710	4500	13740	7600
50 (103-113)	474	742	2130	4500	14050	9870

A1:	LSH with original neighbourhood.	HA1:	GA with TS as new genetic operator.
A2:	LSH with improved neighbourhood.	HA2:	GA with TS imbedded in crossover operator.
GA:	Genetic algorithm.	TS:	Tabu search.

7.1 Comparisons with Time Constraints

In the previous sub-section, the heuristics were compared in terms of objective function value and length of time required to determine the best solution. This does not demonstrate the rate of convergence for each of the algorithms. In practice, there may be a time limit for which a solution must then be made available. The best solution for each algorithm is considered after a fixed period of CPU time. The problems are terminated at intervals ranging from 5 seconds to 50 seconds of CPU time and the best solution at each interval is recorded. Results are summarised in Figure 6 for 20 train problems and Figure 7 for 50 train problems for all heuristics and the branch and bound (B and B) technique. Each value in these figures represents the increase over the optimal solution as a proportion of the optimal solution. A value of 0.05 would mean that the solution is five percent above the optimal.

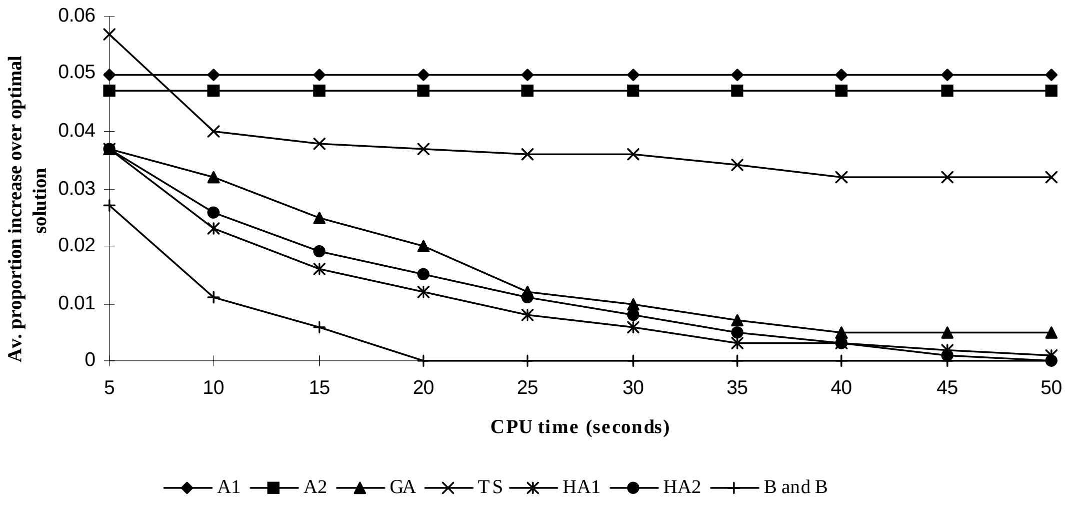

Figure 6: Performance of algorithms with respect to solution time (20 train problems)

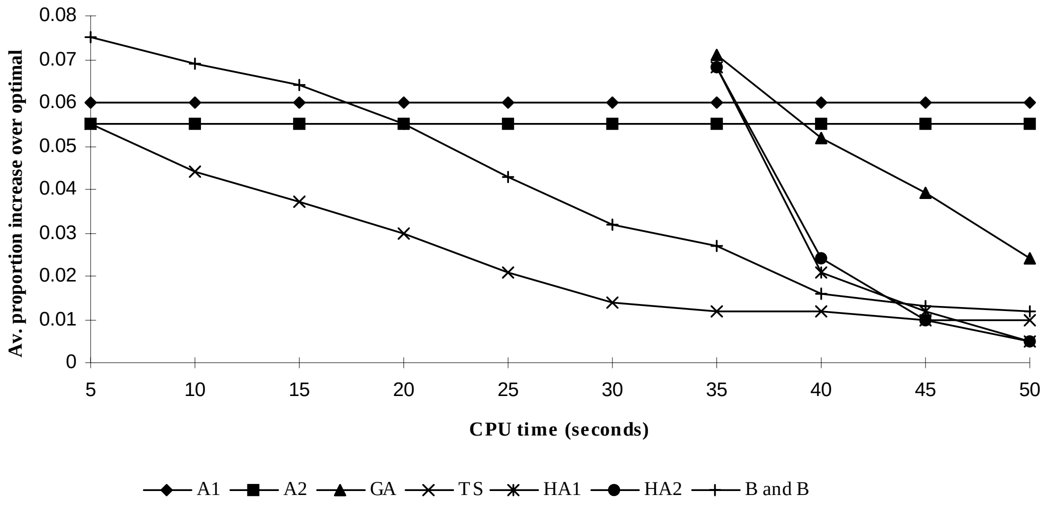

Figure 7: Performance of algorithms with respect to solution time (50 train problems)

Referring to Figures 6 and 7, the LSH (with both neighbourhoods) were at their best values (or close to it) after 5 seconds, but did not improve significantly in the longer time constraints.

For the 50 train problems, GA, HA1 and HA2 were not able to find an initial population within the shorter time constraints (less than 35 seconds in both figures). Once the initial population was found, the GA still gave poor results due to the slower convergence than the other heuristics. Since HA1 used the LSH as a genetic operator, it was able to find some good solutions as soon as the initial population was constructed. The TS and LSH converged quickly at first (since many good improving moves were found), after which the rate of improvement slowed down quickly. The convergence for GA, HA1 and HA2 were more constant throughout the solution process.

The convergence rate for the branch and bound technique is similar to the TS and LSH heuristics. When the optimal solution is approached using the branch and bound technique, an improved solution is more difficult to find, thus convergence is slowed.

8 Summary

Presented in this paper was an application of several heuristics to the single line train scheduling problem. The heuristics included the LSH (with improved neighbourhood), GA, TS and two hybrid algorithms. To the authors knowledge, all heuristics (except LSH) were applied for the first time.

The new neighbourhood allows the position of more than one conflict to be shifted in a single move. The size of the neighbourhood (which increases exponentially with problem size), was reduced to one which grows linearly with the number of conflicts in the schedule. When GA were applied, an efficient chromosome representation was used to allow fast convergence. Mutation was performed by randomly choosing a move from the new neighbourhood. One of the hybrid algorithms imbedded the TS into the GA crossover operator so as to conduct a search for suitable parents and crossover points. The other hybrid algorithm used the LSH as a new genetic operator.

Comparisons were made between each of the heuristics with and without constraints on computation time. Time constraints are usually enforced in rail operations, since a solution must be found repeatedly in an on-line manner. This means the ratio of solution quality to computation time must be high. When enforcing time constraints, the best solutions so far were recorded after termination.

The LSH with the improved neighbourhood produced better solutions than the LSH with the original neighbourhood (Kraay 1993) for twelve out of the thirty two problems tested. Solutions to the remaining problems were the same for both neighbourhoods. Although the solutions were still not as good as the GA or TS, computation times were fast and the heuristic gave solutions close to their best when time constraints of various lengths were enforced.

Genetic algorithms gave solutions within five percent of the optimal for ninety percent of the problems tested, and found the optimal solution for nearly fifty percent of these. The algorithm performed poorly compared to the LSH when time constraints were enforced as an initial population was not found for most problems.

On average, the solutions for TS were not as good as the GA, as the optimal solution was only found for ten percent of the problems tested. It was however the best performing heuristic for the larger size problems when time constraints were enforced.

Both hybrid algorithms gave better results than the other heuristics with HA2 producing solutions within the one percent of the optimal for more than ninety percent of the test problems. The computation time for both hybrid algorithms were up to seven times that of the GA. Computation time did not increase as much for HA2 for larger problems sizes.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments on an earlier version of this paper.

References

Cai, X. and Goh, C. J. (1994). "A Fast Heuristic for the Train Scheduling Problem", *Computers and Operations Research*, Vol 21, 499 - 510.

Davis, L. (1987). Genetic Algorithms and Simulated Annealing, Pitman.

Davis, L. (1991). Handbook on Genetic Algorithms, New York: Van Nos-trand Reinhold.

Gendreau, M., Hertz, A. and Laporte, G. (1991). "A Tabu Heuristic for the Vehicle Routing Problem", *Centre de Recherche sur les Transports,* Publication 777.

Glover, F. (1990). "Tabu Search: A Tutorial", *Interfaces,* Vol 20, 74 - 79.

Glover, F. (1993). "A Users Guide to Tabu Search", *Annals of Operations Research,* Vol 41, 3 - 28.

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

Higgins, A., Kozan, E and Ferreira, L. (1995). "Rescheduling of Trains on a Single Line Track: Improved Heuristic Solution Techniques" *Physical Infrastructure Research Report , 1-95,* Queensland University of Technology, Brisbane.

Higgins, A., Kozan, E. and Ferreira, L. (1996). "Optimal Scheduling of Trains on a Single Line Track", *Transportation Research B,* Vol 30, In Press.

Jovanovic, D. (1989). "Improving Railroad On-time Performance: Models, Algorithms and Applications", PhD Thesis at Department of Decision Sciences. The Wharton School. University of Pennsylvania.

Kraay, D. (1993). "Learning Methods in Optimisation: With Applications to Railroad Control", PhD Thesis at Decision Sciences Department, The Wharton School, University of Pennsylvania.

Kraay, D., Harker, P. and Chen, B. (1988). "Optimal Pacing of Trains in Freight Railroads: Model Formulation and Solution" Working Paper 88-03-03, Decision Sciences Department, The Wharton School, University of Pennsylvania.

Kraay, D., Harker, P. and Chen, B. (1991). "Optimal Pacing of Trains in Freight Railroads", *Operations Research,* Vol 39, 82 - 99.

Miller, J. A., Potter, W. D., Gandham, R. V. and Lapena, C. N. (1993). “An Evaluation of Local Improvement Operators for Genetic Algorithms”, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol 23, 1341-1351

Mills, R. G., Perkins, S. E. and Pudney, P. J. (1991). "Dynamic Rescheduling of Long Haul Trains for Improved Timekeeping and Energy", *Asia-Pacific Journal Operational Research*, Vol 8, 146 - 165.

Nordstrom, A. and Tufekci, S. (1994). “A Genetic Algorithm for the Talent Scheduling Problem”, *Computers and Operations Research*, Vol 21, 927 - 940

Semet, F. and Taillard, E. (1993). "Solving Real Life Vehicle Routing Problems Efficiently Using Tabu Search", *Annals of Operations Research*, Vol 41, 469 - 488.

Srivastava, B. and Chen, W. (1993). “Part Type Selection Problem in Flexible Manufacturing Systems: Tabu Search Algorithms”, *Annals of Operations Research*, Vol 41, 279 - 297.

Tan, B. T. G. and Lim, S. M. (1996). “Automated Parameter Optimisation for Double Frequency Modulation Synthesis Using the Genetic Annealing Algorithm”, *Journal of the Audio Engineering Society*, Vol 44, 3 - 15