Techniques for inserting additional trains into existing timetables

R. L. Burdett and E. Kozan

School of Mathematical Sciences, Queensland University of Technology, PO Box 2434, QLD 4001, Australia

Abstract

In this paper techniques for scheduling additional train services (SATS) are considered as is train scheduling involving general time window constraints, fixed operations, maintenance activities and periods of section unavailability. The SATS problem is important because additional services must often be given access to the railway and subsequently integrated into current timetables. The SATS problem therefore considers the competition for railway infrastructure between new services and existing services belonging to the same or different operators. The SATS problem is characterised as a hybrid job shop scheduling problem with time window constraints. To solve this problem constructive algorithm and meta-heuristic scheduling techniques that operate upon a disjunctive graph model of train operations are utilised. From numerical investigations the proposed framework and associated techniques are tested and shown to be effective.

Keywords: Train scheduling, job shops, time windows, meta-heuristics.

1. Introduction

This paper considers how to alter an existing train timetable to include additional train services. This problem is of considerable difficulty and must be performed in practice. The abbreviation SATS has been coined to describe this problem and is used throughout this paper. SATS is both a scheduling and rescheduling problem. For example the additional train services are not present in the original timetable and are scheduled for the first time while the existing services are either left untouched or are rescheduled. It should be noted that the original schedule in which the additional trains are to be inserted, is expected to be feasible, but this is not a necessity. The SATS problem is also viewed as a rescheduling problem because the determination of a satisfactory timetable is a process of iterative refinement. The construction of a timetable "in one go" is unlikely due to the negotiation process between the different customers who are requesting rail access. SATS however is not a "reactive" or "dynamic" scheduling problem in the traditional sense since the insertion of additional trains is planned well in advance of when the schedule must be commenced.

The primary motivation for this research occurs as a result of the additional demands that are being placed upon railway operators in Australia and elsewhere to provide third party usage of infrastructure. An important aspect of this problem is therefore the competition amongst new services and or with existing services for railway infrastructure. The railway operator / planner must decide where competition is permitted and where it is restricted. Put in another way the scheduling of the additional train services must be performed so that disruptions to existing services are minimised or similarly kept within acceptable levels. If this distinction is not made then a new timetable that greatly differs from the original could be obtained. This is a valid though different problem again and has already been significantly addressed for example in our previous work.

What an acceptable level of disruption is however is fairly subjective. It should also be mentioned that in some circumstances some train services must be strictly fixed and can suffer no disruption. The level of acceptable disruption also widely differs according to the train service type. Passenger trains for example are usually subject to very strict timing. This timing is necessary because of the type of path these services take (i.e. through more heavily utilised city networks) and the regular frequency of the services (i.e. same service every day, or several times a day). Similarly passenger's tolerance to delays and alterations is quite limited. Freight train services on the other hand do not usually have strict timings because they are infrequent services. Similarly the services travel longer distances on less utilised corridors at off peak times. These services are also inherently slower because the locomotives are older and less well maintained and the weight of the goods and the length of the trains are large. The journey time is therefore inherently more unpredictable.

Another important consideration is the utilisation level of the original timetable and the time frame. The utilisation level of the existing timetable is fairly well known. For example by visualisation alone one can tell whether the system is lightly or heavily utilised. Analytically one can tell by counting the number of train paths and comparing this number with some measure of capacity (see Burdett and Kozan 2006). Similarly periods of idle time may be used to determine the extent of the utilisation. Adding a large number of trains to a timetable that heavily utilises the system will ultimately be unsuccessful if minimal disruptions are desired. A relaxation of certain conditions (and tolerances) will be necessary in practice thus promoting some type of iterative process. Furthermore adding a small number of trains to a timetable that heavily under utilises the system will be trivial. Adding trains at incorrect periods of time or having train paths overlap different periods is similarly undesirable. With these issues in mind the following process is proposed for solving the SATS problem:

Phase 1 (FX strategy): Fix all previously scheduled services. Apply constructive algorithms to add new train services and then apply meta-heuristics to refine (improve) the solution further or to remove infeasibilities.

Phase 2 (SFX strategy): Selectively fix and unfix some previously scheduled services and or operations. Reapply meta-heuristics to refine the solution

Phase 3 (UFX strategy): Unfix all previously scheduled services. Reapply meta-heuristics to refine the solution.

The SATS process does not necessarily have to be applied in the order shown above nor is every phase required. For example when track infrastructure (capacity) is already heavily utilised, new trains can only be inserted by disrupting existing services. Phase 1 will be inappropriate and possibly even phase 2. When capacity utilisation is light then phase 1 should be applied. Phase 2 may be applied multiple times with different operations being fixed and unfixed between applications. In the event that phase 1 and 2 are unsuccessful phase 3 is the final resort where any existing service may be altered. In each phase additional train services may be added using a constructive algorithm however it is suggested that the constructive algorithm be applied in the first stage only. In later phases meta-heuristics should only be applied.

The purpose of the remainder of is paper is to generalise several different train scheduling problems and to demonstrate (after suitable extensions) the effectiveness of the hybrid job shop scheduling framework, constructive algorithm and simulated annealing meta-heuristic approach of Burdett and Kozan (2008) for solving SATS related problems. The approaches developed in Burdett and Kozan (2008) have been shown to be very effective for creating a new train schedule from scratch so it is evident that they should be applied to solve the related SATS problem. It should be noted that the constructive algorithm is already able to insert additional train services. What it does not do (or has not been tested to do) until now is obtain a new schedule that minimises disruption to existing services and fixes current services.

This paper is needed because no other paper has explicitly addressed this problem even though, as our review of the literature will show, some past approaches could be modified / adapted to solve it. In addition this paper utilises a job shop scheduling framework and disjunctive graph model of train operations for solving SATS problems and this is still a relatively new avenue for railway problems. The only other competing disjunctive graph model of train operations is that of D'Ariano et al (2007) and Mazzarello and Ottaviani (2007) which are both based upon the alternative graph representation of Mascis and Pacciarelli (2002).

The format of the paper in the following sections is as follows. In section 2 the related research is reviewed. The main details of our scheduling approach are then described in section 3. This includes a disjunctive graph model of train operations, a constructive algorithm and a Simulated Annealing approach. Extensions involving time windows are then presented. Following this is an approach to fix operations and enforce periods of machine unavailability. In section 4 a numerical investigation is described and the results are summarised. The conclusion and future research directions are lastly presented in section 5.

2. Related Research

The SATS problem is related to a variety of topics in the literature. The first and foremost is railway transportation. Train scheduling, rescheduling and routing problems have had a great deal of attention in recent years. Burdett and Kozan (2008) may for example be consulted for a list of foremost papers published between 1990 and 2005. Since then we have observed the following papers: Dessouky et al (2006), Carey and Crawford (2007), D'Ariano et al (2007), Goverde (2007), Mazzarello and Ottaviani (2007), Rodriguez (2007) and D'Ariano et al (2008). Dessouky (2006) applied a branch and bound procedure to determine the optimal dispatching times for trains in complex rail networks. Though effective on a case study it is reported that effective heuristics are necessary for more realistic sized problems. Carey and Crawford (2007) considered railway networks consisting of complex stations linked by multiple one way lines in each direction and developed heuristic algorithms to assist in the task of finding and resolving conflicts in draft train schedules. D'Ariano et al (2007) considered the creation of a conflict free timetable after train operations are perturbed. The scheduling problem is viewed as a job shop with no store constraints and modelled by an alternative graph formulation. A truncated branch and bound approach is used to find a conflict free schedule. Goverde (2007) introduced a stability theory to analyse timetable sensitivity and robustness to delays. A linear system description of a railway timetable in max-plus algebra was used. Mazzarello and Ottaviani (2007) developed an advanced traffic management system for optimising traffic movement in large railway networks. The system consists of a train scheduling and routing module (CDR) and a speed regulation module (SPG). The CDR system is based upon an alternative graph representation of train movements and is used to generate a conflict free schedule. Rodriquez (2007) considered the routing and scheduling of trains through a junction and proposed a constraint programming model. D'Ariano et al (2008) considered the concept of flexible timetables as a policy to improve punctuality. A detailed model for conflict resolution based upon an alternative graph formulation was proposed and solved by three greedy heuristics and a branch and bound algorithm.

To our knowledge the SATS problem has had very little “direct” attention in the above mentioned literature. The only paper to our knowledge is Cacchiani et al (2008). In that paper the insertion of freight trains into an existing timetable of passenger services was considered. A heuristic algorithm based on a lagrangian relaxation of an ILP model was proposed.

There are however indications that some of the previous models and techniques could be modified and adapted to solve the SATS problem.

For example Carey (1994) formulated a general purpose MILP model for train scheduling. Costs or penalties associated with trains departing from or arriving at stations at times other than those which are desired can be added to the objective function provided that they are linear. This feature allows several different train scheduling problems to be characterised and solved; this includes the SATS problem. Since this model is too large and complex to be solved a strategy was proposed that paths the trains one at a time until all trains are pathed once, and if necessary to iteratively re-path trains until an acceptable solution is found. Each sub problem is solved using branch and bound and is constructed from the original model by holding fixed the sequence order (but not the timing) of all already pathed trains. The main drawback of this approach is that it is myopic and it does not explicitly differentiate between existing services and new services. In the SATS problem not all trains have to be added to the schedule, some are already there. It may however be necessary for existing services to be rescheduled in later passes of that approach as changes are forced upon them from the insertion of new services; this needs to be further investigated. In addition it may not be allowable for the timings of certain trains to be altered; this requires a modification to the sub problem model.

Zhou and Zhong (2005) have also taken a similar approach. They considered the generation of a timetable for double track railway applications with multiple objectives. By applying two practical priority rules the integer programming model for the second objective criterion can be decomposed into a number of single train sub problems which are sequentially solved using branch and bound. The approach could in theory be adapted for the SATS problem too. The model would however need to be modified to include suitable time window constraints as they are not currently present. The priority rules used in that paper are rather simplistic and some new priority rules would need to be developed for SATS problems. The approach is also somewhat unsuitable because the departure time of trains from their origin are determined by a preliminary model whose goals are different to that of SATS problems.

Carey and Carville (2003) focused upon the adjustment (revision) of a draft timetable that contains various types of conflict for a single multi platform station with multiple in and out lines typical of those found in Europe. This problem is equivalent to the dispatching problem that considers how to restore the feasibility of a schedule after a disturbance. Methods for finding and resolving conflicts, enforcing headway constraints and assigning platforms to trains were formalised. The advocated heuristic approach considers one train at a time. Provisional or final arrival, departure time and platform are assigned. The train is then fixed temporarily or permanently. The algorithm can cycle back to reconsider previously fixed trains and seek further improvements. Some trains may even be fixed before the algorithm starts. Minimising deviations from desired times may be taken into account by using appropriate cost functions like those defined in Carey (1994). This approach could in theory be adapted for the SATS problem however it is customised for a single complex railway station and is more elaborate than is necessary for the SATS scenarios considered in this paper. For example the conflict resolution and platform assignment aspects are unnecessary. The approach is somewhat unsatisfactory because the draft timetable includes all the trains; it does not consider the alternative times that the new train services can enter the railway. In other words the creation of a draft timetable is not considered and would need to be in order for it to be viable for the SATS problem.

The SATS problem is also related to job shop and other scheduling problems, particularly those with due dates and earliness-tardiness criteria. Earliness and tardiness is of particular importance in this paper because existing train services should be scheduled to arrive and depart on time and new services should satisfy preferred time windows if they have been defined. The SATS problem however is more difficult since every operation may have due date conditions imposed as opposed to jobs. Due date / tardiness problems have been addressed greatly in the literature and on a regular basis. Recent examples included Zesheng et al (1996), Jain and Elmaraghy (1997), Valls et al (1998), Asano and Ohta (2002), Sourd (2005a,b), Viswanath et al (2006), Hendel and Sourd (2006). Zesheng et al (1996) considered a job shop environment and applied a simulated annealing heuristic integrated with an enhanced exchange heuristic. Jain and Elmaraghy (1997) applied genetic algorithms with a reduced surrogate crossover and adaptive mutation mechanism to schedule jobs in flexible manufacturing systems. Valls et al (1998) applied tabu search to a generalised job shop scheduling problem with parallel machines, job batches, setup times, release dates and due dates. The proposed tabu search algorithm uses a sophisticated set of moves aimed at aggressively resolving violated constraints. Asano and Ohta (2002) considered the minimisation of total weighted tardiness in job shop scheduling problem with job specific due dates and delay penalties. The problem was solved with a modified shifting bottleneck approach. Viswanath et al (2006) studied the problem of hierarchical minimisation of makespan and completion time variance (CTV) in static job shops. This problem has applications in many industries where it is necessary to provide jobs the same treatment in minimum time or similarly to keep the system utilisation at high levels while achieving fairness to individual jobs. This paper is of particular relevance to solving the SATS problem in which the fair and impartial access of trains to railway infrastructure is a requirement. Two simulated annealing heuristics were presented and a lower bound for the CTV. Forward and backward scheduling strategies were compared on 82 benchmark problems. Sourd (2005a and b) considered a single machine scheduling problem with general completion time costs and idle period penalties and costs. A dynamic programming approach was developed to schedule sequenced jobs. Even though the sequence is given the problem is not trivial since the optimisation criterion is irregular. More specifically the earliness and tardiness costs are not linear on the whole time horizon. This occurs in practice for example when several “good” time periods exist but have “bad” time periods in between. Hendel and Sourd (2006) considered a one-machine scheduling problem where the objective is the minimisation of earliness-tardiness costs. A local search approach was developed that must deal with job interchanges in the sequence and the timing of sequenced jobs. The paper shows that there are more efficient methods to find the best solution in these interchange based neighbourhood.

The problem of explicitly scheduling additional jobs has had some attention in the scheduling literature. Hall and Potts (2004) is the most recent paper to our knowledge. This paper considered the rescheduling for new orders and was motivated by incidents occurring in manufacturing environments. The re-scheduling must take into account the effect of disruption on a previously obtained optimal schedule. The effect of disruption was modelled as either a constraint or as an additional component in the objective. This paper mainly provides an investigation of the theoretical aspects of the problem and is fairly limited from an application point of view. Furthermore only one machine problems are considered. More complex flow shop and job shop environments that are applicable to train scheduling are not addressed. Due dates are only given to job completion.

The SATS problem contains aspects of job shop scheduling with periods of machine non-availability. Three recent references associated with this topic are Schmidt (2000), Aggoune (2004) and Liao et al (2005). The first paper provided a review of results associated with deterministic scheduling problems where machines are not continuously available for processing. More specifically one machine, parallel machine and flow shop environments were discussed. The general conclusion of this paper was that problems with machine availability constraints are very important and will attract significant attention by researchers. Positive results occurred only for single machine and parallel machine problems. Flow shop, open shop and job shop systems are very challenging and will require enumerative and heuristic algorithms in order to be solved. The second paper specifically addressed flow shop scheduling with availability constraints (FSPAC). This paper stated that only a small number of methods exist in the literature for solving problems with at most two machines and few use a non pre-emption constraint. Therefore two variants of the non pre-emptive FSPAC were considered. In the first case maintenance tasks were strictly fixed and in the second time windows were imposed. An important condition that should be noted is that an operation is started only if its execution can be finished before the machine becomes unavailable. When scheduling trains however the non pre-emption constraint of that paper is invalid. A valid replacement condition is that an operation can only be started if its departure time occurs before the machine becomes unavailable. This requirement is due to the blocking conditions inherent in train scheduling problems that are not present in many manufacturing problems. A schedule generator that builds feasible solutions given a priority rule between jobs was developed. To optimise the input sequence of the schedule generator a heuristic algorithm based upon a genetic algorithm and tabu search was proposed to solve the makespan minimisation problem. The introduction of time windows (i.e. a relaxation of the fixed maintenance times) led to superior makespan solutions. The third paper considered a two parallel machine problem where one machine is not available during a time period. The period is fixed and known in advance. One machine however is always available. Both the pre-emption and non pre-emption cases were investigated. The problem was partitioned into four sub problems, each of which is solved by an algorithm based upon a previous approach labelled TMO. TMO uses lexicographic search to derive an optimal schedule. TMO uses exhaustive search but eliminates lots of unnecessary situations based upon some powerful rules.

3. The Scheduling Approach

At each stage of the SATS solution process a scheduling instance that is computationally intractable (i.e. NP-hard) must be solved. This scheduling problem can be modelled and solved in a variety of ways. In this paper the hybrid job shop framework of Burdett and Kozan (2008) is extended for this purpose.

3.1. The Disjunctive Graph Model

A critical aspect of the sequencing approach is the representation of trains as jobs, sections as machines, and train movements across sections and other section occupancies as operations. Index *i*, *j* and *k* are used to signify train, section and stage respectively. The *k*th operation (i.e. stage) of train i is denoted as $o_{i,k}$ and has a planned sectional running time (occupancy time) of $p_{i,k}$ on section $m_{i,k}$, a planned dwell of $\delta_{i,k}$ at the section boundary and entry and exit times $entry_{i,k}$ and $exit_{i,k}$. Each train *i* has K_i operations. The length of a train and section is denoted by l_i and l_j . The time it takes train *i* to exit the *k*th section after the front has exited is denoted by $lag_{i,k}$. The extent of this lag depends on the length of the train and the speed it is travelling at when it departs the section. The stage and position of the front when the rear is departing the *k*th section in the route is also denoted by $fstage_{i,k}$ and $fpos_{i,k}$ respectively and can be calculated from the railway topography. Passing loops that separate adjacent sections of rail and allow trains to pass each other are represented as capacitated buffers. The capacity of a buffer is typically two trains.

On every section there is a sequence of train movements that must be identified (i.e. this is the solution). This is because only one train may be present on any section at any moment in time. A disjunctive graph model of train operations allows trains to be correctly scheduled from the given sequences and is an essential element of the hybrid job shop scheduling framework. The disjunctive graph has the following components:

$$V = (so, 0.0) \cup \bigcup_{\forall i \in J} \bigcup_{k=1}^{K_i} \left(o_{i,k}, p_{i,k} + \delta_{i,k} \right) \cup (si, 0.0) \tag{1}$$

$$A = \bigcup_{\forall i \in J} \left(so, o_{i,1}, rlt_i \right) \cup \bigcup_{\forall i \in J} \bigcup_{k=2}^{K_i} \left(o_{i,k-1}, o_{i,k}, 0.0 \right) \cup \bigcup_{\forall i \in J} \left(o_{i,K_i}, si, lag_{i,K_i} \right) \tag{2}$$

$$E = \bigcup_{j \in M} \bigcup_{\forall o_{i,k}, o_{i',k'} \in E_j | i \neq i'} \left(e_1 \cup e_2 \right) \text{ where } E_j = \left\{ o_{i,k} \mid m_{i,k} = j \right\} \tag{3}$$

$$e_1 = \begin{cases} \left(o_{i,k^*}, o_{i',k'}, s_{o_{i,k}, o_{i',k'}} - \delta_{i,k^*} - p_{i,k^*} \left(1 - fpos_{i,k} / l_{m_{i,k^*}} \right) \right) \text{ for } k^* = fstage_{i,k} \le K_i \\ \left(o_{i,k^*}, o_{i',k'}, s_{o_{i,k}, o_{i',k'}} + p_{i,k^*} \left(fpos_{i,k} / l_{m_{i,k^*}} \right) \right) \text{ for } k^* = fstage_{i,k} > K_i \end{cases} \tag{4}$$

$$e_2 = \begin{cases} \left(o_{i',k^*}, o_{i,k}, s_{o_{i',k'}, o_{i,k},} - \delta_{i',k^*} - p_{i',k^*} \left(1 - fpos_{i',k'} / l_{m_{i',k^*}} \right) \right) \text{ for } k^* = fstage_{i',k'} \le K_{i'} \\ \left(o_{i',k^*}, o_{i,k}, s_{o_{i',k'}, o_{i,k}} + p_{i',k^*} \left(fpos_{i',k'} / l_{m_{i',k^*}} \right) \right) \text{ for } k^* = fstage_{i',k'} > K_{i'} \end{cases} \tag{5}$$

In (1) each train operation has an associated node and node weight; *V* is the set of nodes and also includes a dummy source and sink node *so* and *si* respectively. The conjunctive (fixed) arc set *A* is constructed in (2); each three tuple represents an arc between two nodes with the given arc weight. The release time of train *i* and train operation $o_{i,k}$ is denoted by rlt_i and $rlt_{i,k}$ respectively. Release times are incorporated for additional timing constraints that may be imposed by planners. The set of disjunctive arcs is created by expression (3)-(5). Equation (3) states that there are disjunctive arcs associated with every section and for each pair of train operations that traverse this section. For example for each pair of train operations $\left(o_{i,k}, o_{i',k'}\right)$ that use this section there is a pair of arcs as defined by e_1 and e_2, one for the precedence $o_{i,k} \prec o_{i',k'}$ and one for the precedence $o_{i',k'} \prec o_{i,k}$ respectively. As this problem is not a classical job shop the disjunctive arcs are not $\left(o_{i,k}, o_{i',k'}\right)$ and $\left(o_{i',k'}, o_{i,k}\right)$. The length of trains is included and the arcs must start from other operation nodes namely o_{i,k^*} and o_{i',k^*} which are later operations of train *i* and i' respectively. The arc weights in essence project backwards from these later nodes and take into account the processing requirements that occur between $\left(o_{i,k}, o_{i,k^*}\right)$ and $\left(o_{i',k'}, o_{i',k^*}\right)$ respectively (where k^* is different in each). It should be noted that $s_{o_{i,k},o_{i',k'}}$ and $s_{o_{i',k'},o_{i,k}}$ are the headway (setup) times between two train operations.

A valid sequence is manifested as a selection of disjunctive arcs from set E that do not cause a cycle in the disjunctive graph. To obtain a schedule of entry and exit times a non delay scheduling policy is assumed. That is, unforced idle time is not allowed and train operations must be scheduled as early as possible. In other words a train should immediately enter a section if that section is free. The longest path algorithm is applied to the disjunctive graph to accomplish this. The result is longest path values $lpv_{i,k}$ for each operation. The following equations must be evaluated afterwards or else applied during the longest path algorithm to explicitly obtain the schedule.

$$entry_{i,k} := lpv_{i,k} - p_{i,k} - \delta_{i,k} \qquad \forall i \in J, k = K_i,..,1 \tag{6}$$

$$exit_{i,k} := \begin{cases} lpv_{i,k} + lag_{i,k} \quad \forall i \in J; k = K_i \\ lpv_{i,k^*} - \delta_{i,k^*} - p_{i,k^*}\left(1 - fpos_{i,k} \big/ l_{m_{i,k^*}}\right) \quad \forall i \in J; k = K_i - 1,..1; k^* = fstage_{i,k} \mid k^* \le K_i \\ lpv_{i,k^*-1} + p_{i,k^*-1}\left(1 - fpos_{i,k} \big/ l_{m_{i,k^*-1}}\right) \quad \forall i \in J; k = K_i - 1..1; k^* = fstage_{i,k} \mid k^* = K_i + 1 \end{cases} \tag{7}$$

$$delay_{i,k} = exit_{i,k} - \left(entry_{i,k} + p_{i,k} + \delta_{i,k} + lag_{i,k}\right) \; \forall i \in J, k = K_i,..,1 \tag{8}$$

The alternative graph models of train operations mentioned in D’Ariano et al (2007, 2008), and Mazzarello and Ottaviani (2007) are similar to the above model in concept but have several important differences. The first is that our model is an activity on node approach while theirs is an activity on arc. Consequently one additional node is needed per train in their approach. In addition our conjunctive arcs have a weight of zero and our disjunctive arc weights are non zero. In their approach the opposite occurs. The disjunctive arcs between pairs of operations always originate from the immediate successor operation in their model. In our approach the immediate successor operation may not be sufficient if the train length exceeds the length of the associated section. In that case our disjunctive arcs originate from a later successor operation.

Our approach takes into account planned dwell time and its effect on other train operations in the form of additional time lag. Dwell time is stored as additional node weight and is also used to compute disjunctive arc weights. In their model planned dwell time is manifested on conjunctive arcs. In addition two nodes are needed for each location where dwell time occurs. This is a second source of additional nodes and results in roughly one extra node for each planned dwell.

Our model has been designed as a fixed speed model but not a constant speed model. In other words train speeds may differ on each section. The alternative graph model however may treat the processing time as a variable, i.e. the train speed may be selected as long as it does not exceed some maximum amount. This is achieved by adding a fixed reverse arc and ensuring that a positive length cycle does not occur.

Passing facilities are modelled differently in our approach. We treat passing facilities as capacitated buffers in order to remove routing decisions from the problem. In so doing we need to compute buffer occupancy levels and ensure that they do not exceed the capacity of the passing loop infrastructure and cause buffer occupancy violations (BOV) in final solutions. Buffer occupancy is determined after the disjunctive graph has been evaluated and decoded. In particular the operation entry and exit times may be used as they provide an interval of machine occupancy. The original overlapping intervals of machine occupancy are split into separate non-overlapping intervals, each with its own integer parameter signifying the number of occupants. If the number of occupants of any non-overlapping interval exceeds the capacity of the buffer, then a buffer occupancy violation (BOV) has been found. Buffer occupancy violations may be temporarily permitted and are penalised in the objective function.

3.2. Schedule Construction and Refinement

In order to obtain an optimal schedule for a given objective criteria, a constructive algorithm (CA) and a simulated annealing (SA) approach from Burdett and Kozan (2008) are utilised.

The purpose of the constructive algorithm is to provide a robust and substantial search capability as apposed to the alternative of obtaining any solution as quick as possible using dispatching rules. The constructive algorithm builds the sequences and hence the schedule train by train and train operation by train operation using insertion and backtracking mechanisms. The order that trains are inserted is typically from largest to smallest total transit time (processing time). At each step an operation is trialled in various positions of a partial sequence and evaluated for merit. The best position is chosen and the insertion is made permanent. If each position is infeasible and results in a cycle then backtracking of a previous move(s) is performed. Moves that cause cycles are recorded and not made again.

An existing schedule is improved using a simulated annealing approach. Our simulated annealing approach is quite conventional in structure; a more complex variant was unnecessary but could be investigated as a source of future research. For example there is an outer temperature loop which is terminated when the number of temperature steps reaches a predefined limit and an inner loop which is terminated after a given number of state changes at the given temperature. A state change is accepted using the standard metropolis function. Lastly the temperature is altered at the end of the inner loop. The basic geometric cooling schedule was used and the initial temperature was chosen using past experience and experimentation.

The solution which is a collection of operation sequences (i.e. one for each non buffer machine) may only be realistically and efficiently perturbed in a small number of ways. For example operations may be individually shifted, pairs of operations may be interchanged (exchanged), or a sub sequence of operations may be reversed. Entire trains may also be removed and re-inserted using the constructive algorithm as an alternative perturbation strategy. From numerical investigations it was found that shifting an operation one position to the left or right within a sequence is sufficient as a means of perturbing (refining) an existing train schedule and allows relatively good solutions to be obtained with reasonable computational expense. Due to a number of limitations of this unitary perturbation method several innovative compound interchange strategies were developed. In each strategy a number of train operations are shifted in one go. One compound move undoes a current overtaking or passing situation. Another compound move does the opposite (i.e. makes trains pass or overtake each other). The focus of the proposed compound moves is to perform interchanges without causing multiple overtaking conflicts and secondly to move whole trains past each other easily within an existing schedule. In our approach simple interchanges and exchanges are used as the basis of the compound moves. An interchange swaps the position of two adjacent operations while exchanges swap the relative position of two operations that are not necessarily adjacent. The compound moves require several of these “single” moves to be performed “simultaneously

In both the CA and SA approach solution feasibility may be strictly enforced or infeasible moves may be permitted and penalised in the objective function. Solution infeasibility refers to the situation where buffer occupancy violations are permitted to occur. The number of BOV is penalised. The penalty value may be fixed or else increased proportionally as the temperature decreases. Infeasibilities caused from cycles in the disjunctive graph are strictly prohibited.

3.2. Time Windows

Time windows are the primary mechanism by which the SATS scheduling problem is most efficiently characterised. Time windows are given to both existing services and to new services. Typically the time windows given to new services would be soft constraints, (i.e. preferences) whereas time windows for existing services would be strict inviolable conditions. Exceptions to these assumptions are permissible and may be necessary in some applications.

Time windows $tw_{i,k}^{\text{entry}}$ and $tw_{i,k}^{\text{exit}}$ are introduced for each operation $o_{i,k}$ for the entry and exit time on each section that is traversed during a train's journey and provide a complete profile of each train's allowable movements. These parameters represent the pair $\left(earliest_{i,k}^{\text{entry}}, latest_{i,k}^{\text{entry}}\right)$ and $\left(earliest_{i,k}^{\text{exit}}, latest_{i,k}^{\text{exit}}\right)$. They are introduced to impose the following:

$$earliest_{i,k}^{\text{entry}} \leq entry_{i,k} \leq latest_{i,k}^{\text{entry}} \quad \forall i,k \tag{9}$$

$$earliest_{i,k}^{\text{exit}} \leq exit_{i,k} \leq latest_{i,k}^{\text{exit}} \quad \forall i,k \tag{10}$$

The valid interval of occupancy for operation $o_{i,k}$ is therefore given by $\left(earliest_{i,k}^{\text{entry}}, latest_{i,k}^{\text{exit}}\right)$ and denoted as $tw_{i,k}$. There are three types of time window: unrestricted, loose and tight. For example a time window of $[0,\infty)$ signifies that there is no limitation on entry or exit. A time window $[\alpha,\beta]$ signifies a restricted but loose time window if $\alpha < \beta$. If $\alpha = \beta$ then the time window is tight and fixed entry and or exit is "desired". The difference between the time window range and the planned occupancy time signifies the level of flexibility. In other words, a larger time window provides a greater level of flexibility.

Time window values are defined so that the following relationships hold between adjacent operations of the same train:

$$earliest_{i,k}^{\text{entry}} = earliest_{i,k-1}^{\text{exit}} - lag_{i,k-1} \quad \forall i, k = 2,..,K_i \tag{11}$$

$$latest_{i,k}^{\text{entry}} = latest_{i,k-1}^{\text{exit}} - lag_{i,k-1} \quad \forall i, k = 2,..,K_i \tag{12}$$

The lag parameter is required because trains may be present on multiple sections at once because of their length. Two of the time window parameters, namely latest entry and earliest exit are dependent values and should be computed in the following way:

$$latest_{i,k}^{\text{entry}} = latest_{i,k}^{\text{exit}} - p_{i,k} - lag_{i,k} \quad \forall i,k \tag{13}$$

$$earliest_{i,k}^{\text{exit}} = earliest_{i,k}^{\text{entry}} + p_{i,k} + lag_{i,k} \quad \forall i,k \tag{14}$$

It should be noted that the time windows for the first and last operation (i.e. k=1 and k= K_i) signify a general time window for a train. Parameters signifying the time window violations for an operation are denoted by $twv_{i,k}^{\text{entry}}, twv_{i,k}^{\text{exit}}$ and $twv_{i,k}$. They are computed in the following way:

$$twv_{i,k}^{\text{entry}} = earliness_{i,k}^{\text{entry}} + tardiness_{i,k}^{\text{entry}} \quad \forall i,k \tag{15}$$

$$twv_{i,k}^{\text{exit}} = earliness_{i,k}^{\text{exit}} + tardiness_{i,k}^{\text{exit}} \quad \forall i,k \tag{16}$$

$$twv_{i,k} = twv_{i,k}^{\text{entry}} + twv_{i,k}^{\text{exit}} \quad \forall i,k \tag{17}$$

The parameters $earliness_{i,k}^{\text{entry(exit)}}$ and $tardiness_{i,k}^{\text{entry(exit)}}$ represent the earliness and tardiness of operation entry and exit with respect to the given time windows. An operation however may not be early and tardy at the same time. Removal of the entry (exit) superscript denotes the total earliness and tardiness of the operation. Earliness and tardiness are computed in the standard way:

$$earliness_{i,k}^{\text{entry}} = \max\left(earliest_{i,k}^{\text{entry}} - entry_{i,k}, 0\right) \quad \forall i,k \tag{18}$$

$$tardiness_{i,k}^{\text{entry}} = \max\left(entry_{i,k} - latest_{i,k}^{\text{entry}}, 0\right) \quad \forall i,k \tag{19}$$

$$earliness_{i,k}^{\text{exit}} = \max\left(earliest_{i,k}^{\text{exit}} - exit_{i,k}, 0\right) \quad \forall i,k \tag{20}$$

$$tardiness_{i,k}^{\text{exit}} = \max\left(exit_{i,k} - latest_{i,k}^{\text{exit}}, 0\right) \quad \forall i,k \tag{21}$$

The expression for the time window violation includes both earliness and tardiness. If one and not both is important then earliness and tardiness criteria must be defined for each operation. The time windows however can be altered so that this is unnecessary. For example if early entry or exit is not important then a $[0,\beta]$ window can be defined. The equation for early entry and early exit will always result in a zero value. Similarly if late entry or late exit is not important then a $[\alpha,\infty)$ window can be defined.

Due to the possibility of infeasible schedules being created when all time windows are strictly enforced, the penalisation of time window violations via the objective function is necessary. In other words time window constraints are not directly enforced. It should be noted that numerous schedules may have the same level of disruption but completely different makespans. The minimisation of disruptions is not necessarily sufficient as a single objective criterion for this problem. The SATS problem therefore has two objectives. The primary objective is to schedule trains so that all time windows are satisfied. This is equivalent to minimising the total weighted time window violations or similarly the total cost of all violations. The secondary objective is to optimise some generic measure of schedule quality such as makespan or total train delays. The objective function has two weighted terms and the relative importance of each objective dictates the values of the associated weights. The objective function for this approach is as follows:

$$\text{Minimise } z = \lambda_1 \left[\max_i \left(exit_{i,K_i} \right) \right] + \lambda_2 \left[\sum_{\forall (i,k)} \left(\omega_{i,k} twv_{i,k} \right) \right] \tag{22}$$

The time window violation penalty value or weight for operation $o_{i,k}$ is given by $\omega_{i,k}$ and has a default value of one. In reality this value will be train type dependent and may even be zero.

3.3. Fixing Existing Trains

The penalisation of time window violations provides a simple and straightforward method to persuade operations to be scheduled at desired times in our sequencing approach. However satisfying time windows becomes much more difficult when the intervals are narrow and competition for resources at a particular time is great. Operations with "tight" time windows such as those belonging to existing train services are even harder to explicitly fix using the usual time windows mechanism. Furthermore even if they can be fixed, the time window mechanism is not always particularly efficient at doing so. Lastly the scheduling of sequenced tasks is not necessarily trivial as reported in Sourd (2005).

Because of these factors an alternative approach to keep operation entry and or exit times explicitly fixed is introduced. After all the primary purpose of the SATS problem is very often the identification of feasible insertion positions for new jobs without the requirement to shift or otherwise disrupt existing jobs. Our main approach to keep an operation's entry (and exit) time fixed is quite simple. We introduce the condition that it is not permissible to violate the time window of a fixed operation. In other words any move that does not cause fixed operations to have time window violations is deemed feasible. Algebraically $\sum_{\forall i,k \mid fixed_{i,k}=true} \left(twv_{i,k} \right) = 0$ implies feasibility of the move. The objective is modified so that only unfixed operations contribute to the time window violations term as follows:

$$z = \lambda_1 \left[\max_i \left(exit_{i,K_i} \right) \right] + \lambda_2 \left[\sum_{\forall (i,k) \mid fixed_{i,k} = false} \left(\omega_{i,k} twv_{i,k} \right) \right] \tag{23}$$

The elegance of the fixed operation strategy is that entire trains do not need to be fixed. For example isolated operations of a job can be selected as fixed. Similarly operation entry may be fixed or unfixed independent of fixing or unfixing operation exit. It should be noted that the position of fixed operations in the associated machine sequence may change however the entry and exit times do not.

In order to solve the SATS problem the jobs and associated operations that should be fixed must be identified. One way is through time window attributes. For example an operation may be classified as fixed in some sense if one or both of its time windows (i.e. for entry and exit) are tight. If both time windows are tight then the operation is completely fixed otherwise it is partially fixed. A more robust approach is to introduce parameters $fixed_{i,k}^{\text{entry}}$, $fixed_{i,k}^{\text{exit}}$ and $fixed_{i,k}$ to denote what level of fixing if any is to occur. Consequently an operation with a tight entry and or exit time window does not necessarily have to be set as fixed. Similarly it is also important to note that a fixed operation does not necessarily have to have a tight time window. That is, an operation may be fixed within a loose time window. In our approach however an operation with a tight time window is automatically fixed. Similarly an operation that does not have a tight time window is not automatically fixed. If it must be fixed then the operation must be manually selected and designated as fixed through some type of interface.

If these flags are not true then the train is free to move and the time windows may be violated. In essence these flags signify whether the time window is a hard or soft constraint. It should also be mentioned that fixing the entry time of an operation forces the exit time of the predecessor operation to be also fixed. Similarly fixing the exit time of an operation also fixes the entry time of the successor operation:

$$fixed_{i,k}^{\text{entry}} = fixed_{i,k-1}^{\text{exit}} \quad \forall i,k \mid k > 1 \tag{24}$$

$$fixed_{i,k}^{\text{exit}} = fixed_{i,k+1}^{\text{entry}} \quad \forall i,k \mid k < K_i \tag{25}$$

The introduction of fixed entry and exit conditions for operations means that the latest entry and earliest exit values are now computed in the following way:

$$latest_{i,k}^{\text{entry}} = \begin{cases} earliest_{i,k}^{\text{entry}} & fixed_{i,k}^{\text{entry}} = true \\ latest_{i,k}^{\text{exit}} - p_{i,k} - lag_{i,k} & fixed_{i,k}^{\text{entry}} = false \end{cases} \tag{26}$$

$$earliest_{i,k}^{\text{exit}} = \begin{cases} latest_{i,k}^{\text{exit}} & fixed_{i,k}^{\text{exit}} = true \\ earliest_{i,k}^{\text{entry}} + p_{i,k} + lag_{i,k} & fixed_{i,k}^{\text{exit}} = false \end{cases} \tag{27}$$

Once a schedule has been generated, non fixed operations may be selected as fixed at any time and vice versa. It should be noted that unfixing (releasing) a fixed operation leaves the original time window unaltered. If time window violations associated with this operation are no longer allowed then an unrestricted time window must be set manually (i.e. in some interface). It should also be noted that unscheduled (un-inserted) operations can not be set as fixed. Fixing an unfixed (scheduled) operation requires a redefinition of the operations time window to its current entry and exit time. Fixed operations must be also assigned release times in the following way in order to be scheduled correctly.

$$rlt_{i,k} = earliest_{i,k}^{\text{entry}} \quad \forall i,k \mid fixed_{i,k}^{\text{entry}} = true \tag{28}$$

$$rlt_{i,k+1} = earliest_{i,k}^{\text{entry}} \quad \forall i,k \mid k < K_i,\ fixed_{i,k}^{\text{exit}} = true \tag{29}$$

The first equation is for when an operations entry time is to be fixed. Similarly the second is for when the exit time is to be fixed. If release times are not defined then unfixing a fixed operation and then moving it may cause another fixed operation to be shifted resulting in a time window violation. Consequently the operation can not be moved even though it is unfixed. This is because the longest path algorithm schedules operations are early as possible.

The act of fixing an operation “in place” makes a machine “off limits” to other jobs. Consequently a period of machine unavailability / non availability (PNA) is created. In a more general sense a period of machine non availability may be viewed (classified) as an idle operation, maintenance operation or previously scheduled operation. In our approach a single “isolated” PNA operation should be defined as a separate job while “PNA” operations belonging to the same previously scheduled job should be grouped as one job.

In our sequencing approach all PNA are represented by fixed operations. More specifically no PNA can exist independently and without an accompanying operation. This is because an operation provides a sequencing object whereas an interval does not. Maintaining fixed operations in the schedule has several advantages. Non fixed operations may be scheduled at correct “distances” (i.e. in time) from fixed operations without the need to add artificial release times. Feasible (and or infeasible) intervals of machine occupancy need not be calculated for every operation of every non fixed job. This approach does not need complex and irregular cost functions like those used in Sourd (2005a), Sourd (2005b) and Hendel and Sourd (2006).

4. Numerical Investigations

This section provides details of comprehensive numerical investigations. The primary aim of the numerical investigations is to identify whether good solutions can be obtained using the methodology and techniques proposed in this paper and our previous paper. For example we would like to know what effect the introduction of time window constraints has and whether CA and SA approaches still provide good quality solutions. We would also like to know how to use CA and SA most appropriately; for example what settings should be used and so forth.

Of the different settings that may be used the CA was applied with the greedy insertion option. That is, train operations are inserted in the position that gives the best objective function value. Both the BOV permit and BOV penalise options were tested as each option has previously demonstrated merit.

The application of Simulated Annealing with different perturbation strategies and different control parameters was also investigated. The strategy for perturbing a solution was the shifting of individual operations. The more advanced compound interchange move was also tested but too many time window violations were created. This strategy has therefore been abandoned at this stage.

Four different ways of selecting an operation are reasonable for this problem. The first three have been used in previous numerical investigations. They are labelled: random, most delayed, critical. The fourth is a new alternative specifically for this problem. It involves the random selection of an operation from only those that have time window violations. As the search progresses the number of operations with time window violations should be reduced.

4.1. Test Problems

In order to properly test the solution techniques a variety of test problems must be generated. Choosing a set of unbiased test problems however is not simple. For example the number of current trains must be decided as should the number of new trains. The utilisation level of the existing schedule may be constant or variable in which case there may be periods of heavy and light traffic. For new trains, time windows may be set or not set. For current trains, operations may be fixed or time windows may be set, or no time windows may be set. Furthermore the size of the time windows can be set in many different ways. It should also be mentioned that the SATS process can not be followed exactly as the preferences of different parties is not known.

With these issues in mind we have decided to apply the techniques to some of the more important parts of the SATS process. Firstly the construction of an existing timetable from "scratch" using time window constraints only is tested (Part 1a). For example in practice it is quite conceivable that a large number of services will have time window constraints. This scenario is labelled "REBUILD" for obvious reasons. Secondly (Part 1b) an existing schedule is taken and additional train services with no time window conditions are inserted so that the disruption to existing services is minimised. Existing services are given tight time windows but may be shifted if improvements to the makepsan can be realised. This scenario is labelled "EXTEND" and is a shortcut to the "REBUILD" option. Thirdly (Part 2) the insertion of additional services subject to the fixing of the existing services in the "EXTEND" scenario is tested. This scenario is labelled "FIXED" and is a shortcut to the EXTEND option.

The test problems that were selected have been taken from Burdett and Kozan (2008) and are indicative of real life applications. They are primarily serial and non serial lines with alternating sections and sidings. The specifications of these problems are shown in Table 1. It should be noted that *N* is the number of trains, *M* is the number of sequenced sections, *B* is the number of un-sequenced capacitated buffer machines, and *O* is the total number of operations.

These problems increase in size and complexity and contain a mixture of 60, 80, 100, 120 and 160 km/h trains in order to make the problems more challenging. The variation in train speeds ensures that fast trains will run into slower trains quite easily and careful planning of passing and overtaking is necessary in order to maximise throughput. The track topographies are mostly single line bi–directional typical of railways in Australia and North America. Railways containing many parallel lines and more complex stations typical of European countries are not considered and are a source of future research. The problem sizes are also quite typical of job shop scheduling instances found in recent papers and some small to medium sized railway applications. Solving larger railway problems is a source of future research. It should be noted that the added complexities of this problem (such as blocking, capacitated buffers, etc) make it more difficult to solve than classical job shop scheduling problems. This has been observed for example by Mascis and Pacciarelli (2002).

Multiple occurrences of some trains are incorporated. The new occurrences are the simulated third parties. In particular the addition of one to five additional trains is required for the each of the nine problems. The existing schedules are the best found from previous numerical investigations. They are relatively close to optimal for the makespan objective as has been shown from a comparison with lower bounds. Their level of optimality however is of no consequence in this paper because they are simulating an existing schedule which is to be altered.

Table 1. Test problem specifications (45 instances)

Case Study	Track Topography	Input-Output Points	Dimensions *N / M / B / O*	Makespan (minutes)	# of added trains	# of test problems
1	Serial (24 km)	2	6/3/2/30	48.44	1-5	5
2	Serial (164 km)	2	6/5/4/54	279.7	1-5	5
3	Serial (58.2 km)	2	10/10/9/190	88.82	1-5	5
4	Serial (56.2 km)	4	24/10/7/408	127.99	1-5	5
5	Circular (39 km)	1	15/5/3/120	196.5	1-5	5
6	Network (188.6 km)	5	54/30/11/2214	494.95	1-5	5
7	Serial (121.67 km)	2	20/20/19/780	202.92	1-5	5
8	Serial (88.96 km)	2	20/12/12/480	220.22	1-5	5
9	Serial (260.25km)	2	20/24/23/940	403.41	1-5	5
						45

Two objectives were investigated. The first (OBJF1) attempts to minimise the makespan plus total time window violations, while the second (OBJF2) solely attempts to minimise total time window violations.

4.2 Results

Due to the extent of the numerical investigations, the results have been condensed greatly and a summary of the most important results only are shown.

4.2.1. Part 1

In this part of the numerical investigations existing trains are unfixed.

4.2.1.1. Constructive Algorithm

The constructive algorithm was applied with both the BOV permitted (labelled P) and BOV forbidden (labelled F) options and for both objective functions. Furthermore the solution was constructed in two different ways using the “REBUILD” and “EXTEND” options. The constructive algorithm was therefore applied 8 times for each of the 45 test problem and 360 instances were solved in total. The main results for comparison are: the objective function value, the makespan, the time total window violations and the CPU times. Because there are so many raw results and it is difficult to make any firm conclusions from these values, three column charts were constructed for each of the nine examples. A number of conclusions can be reached from these 27 charts.

Significantly better solutions are obtained in all cases when the EXTEND option is taken. This was partially expected because less work must be done in order to obtain a new schedule. Rebuilding the current schedule when it is not required is unnecessary and takes more CPU time. However the REBUILD option is still valuable and in fact critical on the related general time windows problem. The REBUILD option though inferior to the EXTEND option still gives reasonable starting solutions.

With respect to buffer occupancy handling, the BOV permit option resulted in superior solutions than the BOV forbid option. The occurrence of too many BOV in some of the final solutions however is a drawback.

The difference in solution quality between the two objective criteria is not particularly great when BOV are permitted, although the first objective is generally superior in terms of minimal makespan and time window violations. When BOV are not permitted the first objective is significantly better than the second objective function.

As the number of inserted trains was increased the makespan of the schedule also increases. The relative difference between the makespan and a lower bound approximation to the makespan remained fairly constant or increased slightly. In other words the solution quality did not deteriorate as more trains were added to the problem. The chart in Figure 1 demonstrates the average effect. These values were obtained by averaging the results over the nine examples.

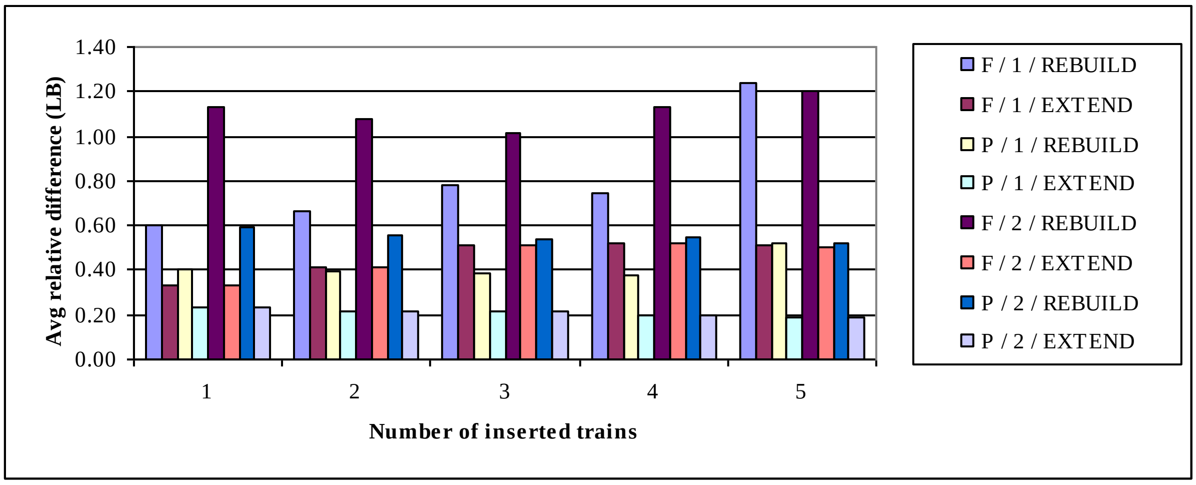

Figure 1. A comparison of the makespan results

The solution quality in terms of makespan was fairly good overall. However the solution quality in terms of time window violations was not particularly good considering that a total time window violation of zero minutes is sought. The level of violations that were generally achieved is a value of several thousand or tens of thousands of minutes. The results for example 9 are shown in Figure 2 and are indicative of the results for the other examples. Only in example 1 were solutions obtained with a total time window violation of zero minutes. The over-penalisation of time window violations is correct but tends to “muddy” the results. In particular the results may seem worse than they actually are because the time window violations appear to be very large. This leads to the conclusion that a better measure is perhaps required. For example a relative or average measure is maybe more indicative of solution quality.

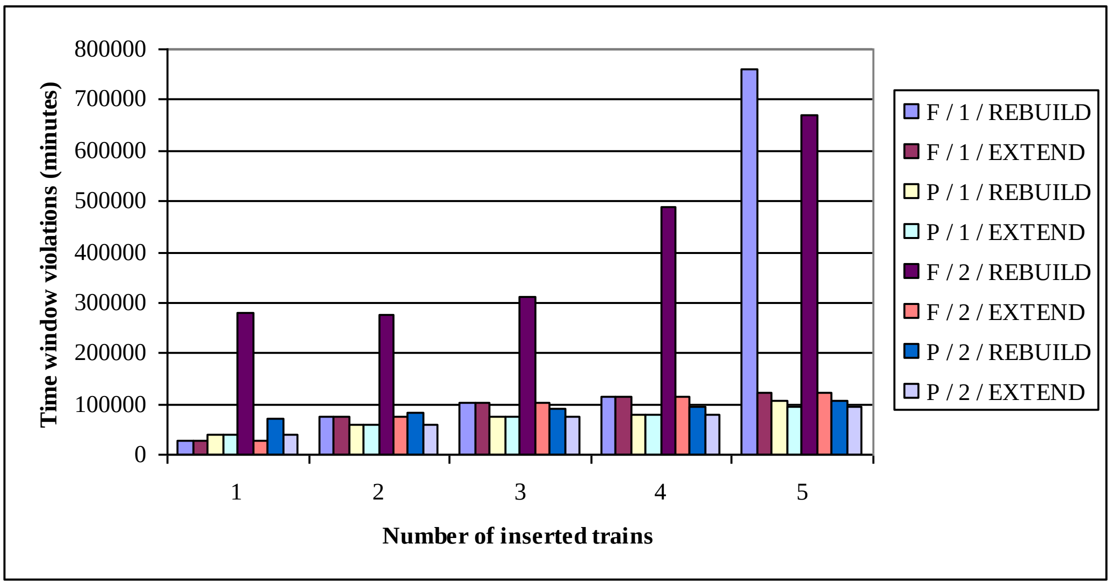

Figure 2. A comparison of the total time window violation results

The CPU times required by the CA typically increased with the number of trains to be inserted or remained the same. On examples 1-5 the CPU times were negligible, i.e. no more than several seconds. On examples 6-9 the CPU times were greater; these are shown in Table 2.

Table 2. CPU times (in seconds) for example 6-9

Case Study	# added	F/1/ REBUILD	P/1/ REBUILD	F/2 / REBUILD	P/2/ REBUILD	F/1/ EXTEND	P/1/ EXTEND	F/2/ EXTEND	P/2/ EXTEND
	1	38	25	39	25	11	1	11	1
6	2	37	27	37	27	12	2	12	2
	3	39	29	39	29	13	3	13	3
	4	47	31	47	31	17	5	17	5
	5	42	33	42	32	20	7	21	8
	1	5	3	54	19	5	3	5	3
7	2	12	7	61	21	12	7	13	7
	3	24	10	44	25	24	10	25	11
	4	105	13	59	28	108	14	113	14
	5	66	33	67	33	140	17	147	18
	1	2	0	67	4	3	0	3	0
8	2	21	1	241	4	22	1	23	1
	3	22	2	356	5	23	2	24	2
	4	22	3	833	6	24	3	26	3
	5	820	7	851	7	26	4	27	5
	1	15	5	62	26	15	5	16	5
9	2	28	10	48	30	29	11	30	11
	3	50	15	136	35	51	15	53	16
	4	54	19	80	41	56	19	58	20
	5	424	46	394	47	80	26	79	27

The longest time was just short of 15 minutes but the usual time was less than a minute. The BOV forbid option took the longest and there was a dramatic increase in CPU time when five trains were to be added to an existing timetable. This construction option finds it more difficult to thread trains through the existing timetable and obtain a feasible schedule. The CA is also forced to perform more backtracking which takes more CPU time.

4.2.1.2. Simulated Annealing

Simulated Annealing was applied in order to improve the best constructive algorithm solution. Based upon the results obtained by the constructive algorithm in 3.2.1.1 SA was applied with objective 1 and not objective 2. Since the "EXTEND" solutions are superior to the "REBUILD" solutions they are used as the starting point for the application of SA.

Three cooling strategies were used and they are as follows: (1/0.01/0.99/100), (10/0.01/0.99/100) and (100/0.01/0.99/100). The format of the parameters is starting temperature, final temperature, temperature reduction factor and evaluations at each temperature. These parameters were chosen based upon past experience and testing. Only the starting temperature was altered in this numerical investigation. The reason for this is that the starting temperature primarily dictates the amount of work that must be done by the algorithm.

The main results are summarised in Figures 3, 4 and 5 below. It should be noted that in the charts, label "F" stands for forbid BOV and "P" for permit BOV. The numerical value after the "/" stands for the operation selection strategy. To obtain Figure 3 the minimum makespan values obtained for the different test problems was used to define relative difference (error) values. In Figure 5 similar calculations were used to compare the time window violations. The average time window value was used to define relative difference values because of data range issues over the different problem sizes and divisions by zero. There are 27 values on the x axis due to the 9 examples and three SA parameter sets. In Figure 5 the total relative differences associated with the makespan are compared so that the effect of different SA parameters can be seen.

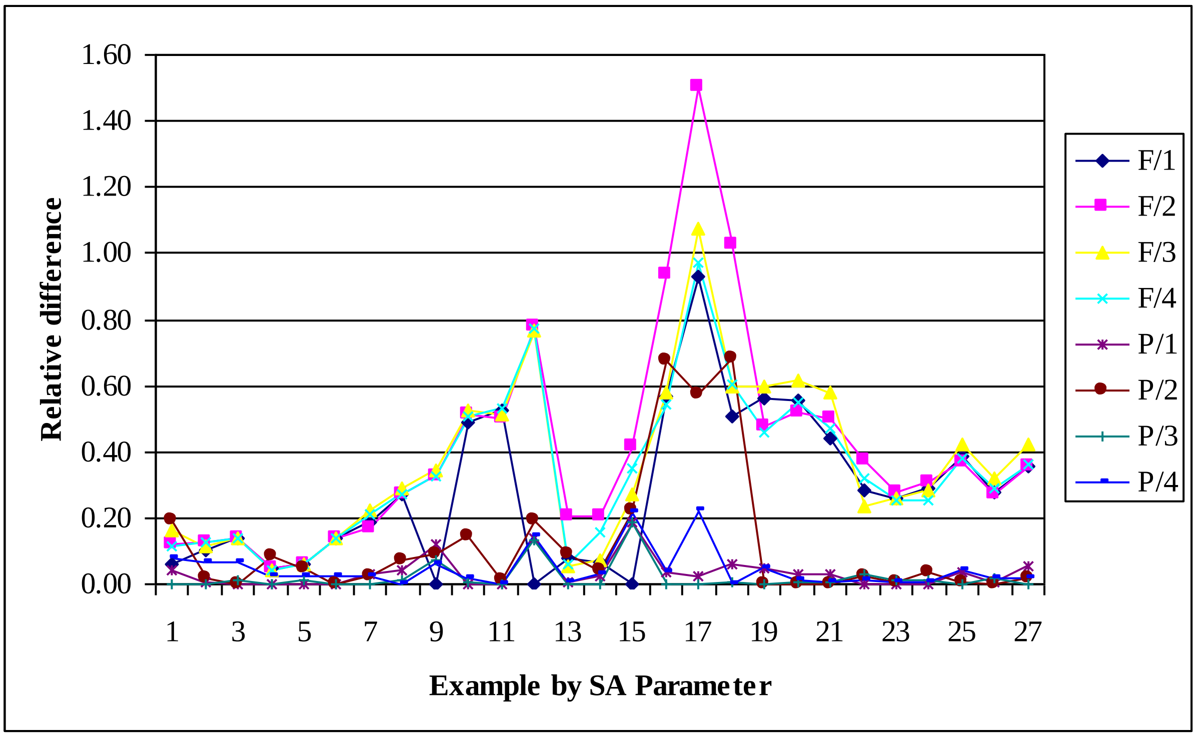

Figure 3. Comparison of relative makespan values

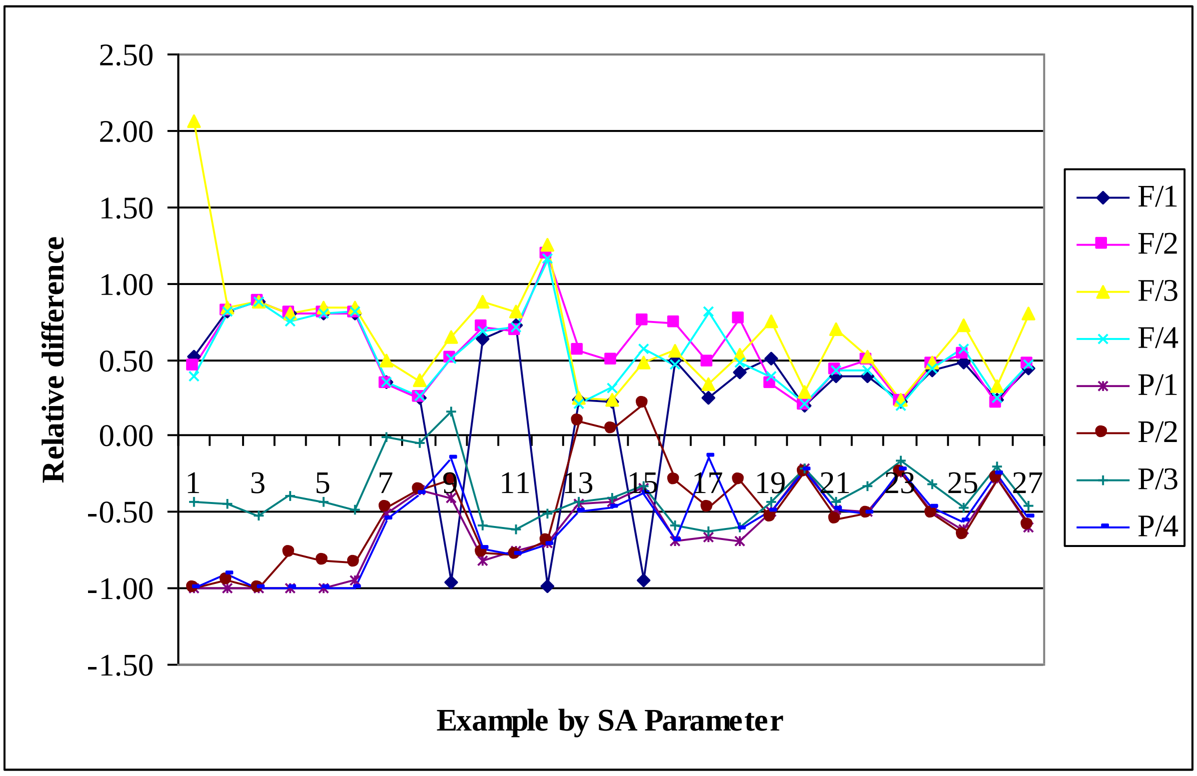

Figure 4. Comparison of average relative time window violations

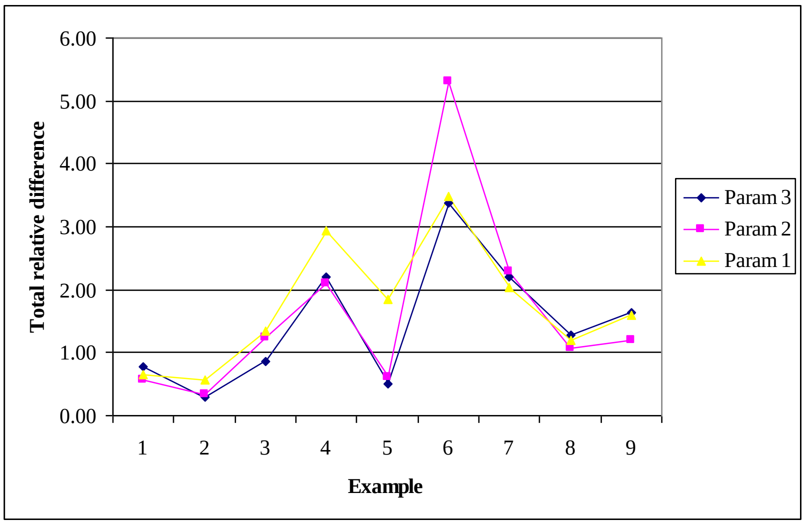

Figure 5. SA parameter comparisons

There is quite a difference in solution quality between the BOV forbid and BOV permit options. Furthermore when BOV are forbidden the random selection of operations for shifting was found to be best. The difference in solution quality for different SA control parameters was not that significant. In other words a faster cooling schedule should be used because it takes less CPU time. When BOV are permitted the results are a little different. In particular the selection of operations from the critical path becomes the best strategy. The difference between these results and those obtained from the next best strategy, i.e. the random selection strategy, however is not great. The new strategy for selecting operations is adequate but does not outperform the previously proposed and tested operation selection strategies.

On some of the problems a higher starting temperature was more advantageous. Since the search space is so restrictive for the SATS problem a higher starting temperature can allow more of the search space to be evaluated and more solutions of lesser quality to be tested in the early stages of the search. This is not unexpected. For example, when the majority of operations have time window constraints then a random operation selection mechanism is quite adequate. Similarly a selection mechanism that chooses operations according to their level of deviation from a preferred time window is a more "pro-active" approach to reducing total time window violations. SA reapplication also seems to be desirable on this type of problem. Superior solutions are not always obtained on the first run of SA.

An indication of the CPU times required by SA is shown in Table 3 below. These values are for SA with the first perturbation operator, i.e. the random operation selection. The CPU times for the other option are comparable and are omitted as there are too many to display.

Table 3. A snapshot of SA CPU times (seconds)

SA Param	Added trains	Example								
		1	2	3	4	5	6	7	8	9
	1	7	13	112	145	45	440	570	499	2109
	2	9	20	133	167	49	994	1415	589	2339
1	3	11	24	150	183	50	1212	1802	643	2490
	4	11	26	176	189	51	1320	2009	704	2731
	5	14	31	200	196	62	1269	2010	688	2870
	1	12	28	172	239	73	661	864	760	3035
	2	15	34	222	258	83	1524	2320	937	3492
2	3	16	40	257	275	86	1720	2778	997	3847
	4	19	49	304	298	90	1946	3116	1124	4181
	5	24	58	344	301	98	714	2809	1071	4600
	1	17	39	247	326	109	907	3030	948	4295
	2	20	46	317	368	122	2448	3644	1206	4789
3	3	23	58	353	368	115	2465	1356	1364	5166
	4	27	66	421	394	113	2471	3684	1517	4830
	5	33	79	480	408	120	2363	4231	1481	5238

4.2.2. Part 2

In this part of the numerical investigations existing trains are fixed.

4.2.2.1. Constructive Algorithm

The constructive algorithm was applied with the BOV forbid and BOV permit options to the previous examples. The existing timetable was however fixed and no alterations were allowed. Therefore no time window violations could occur and the objective criterion was simply makespan minimisation. The raw results are summarised in Table 4 by adding the results of each sub problem.

Table 4 shows that the BOV permit option gives superior makespan values. The improvement is roughly 5-10% on closer inspection of the data but as much as 33% on one example. The number of BOV's that were incurred on each sub problem was not too great.

These numbers increase proportionally with the number of added trains. In summary, the BOV permit solutions were not greatly superior and the application of this option may not be warranted because of the additional effort required to remove the BOV. The BOV forbid solutions are feasible and "ready to go" at no extra effort.

Table 4. Summarised constructive algorithm results for the FIX strategy

Case Study	BOV Forbid	BOV Permit			
	cmax	cmax	#BOV	impr	RE(impr)
1	488.16	452.16	48	36.00	0.07
2	3357.04	2639.54	68	717.50	0.21
3	829.08	744.74	53	84.34	0.10
4	1012.02	970.21	58	41.82	0.04
5	1892.16	1892.16	0	0.00	0.00
6	2627.72	2500.90	60	126.82	0.05
7	1415.31	1377.91	39	37.39	0.03
8	1551.29	1476.82	186	74.47	0.05
9	3068.59	2937.14	26	131.45	0.04

We now compare the solutions with those obtained in Part 1. The raw results are again omitted. The values though are plotted graphically in Figure 6 below to show the underlying differences more clearly.

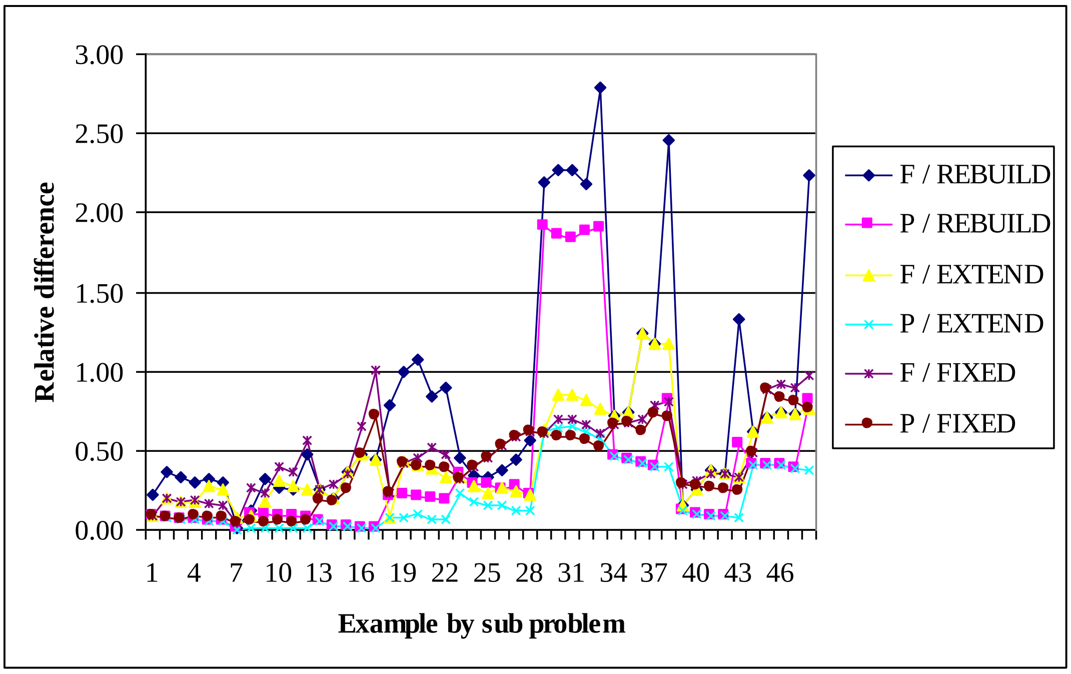

Figure 6. Comparison of relative difference values for the makespan

The raw results are further condensed in Table 5 and Figure 7 by adding the results of each sub problem. These values show that the makespan values obtained (i.e. by the FIXED option) are only slightly inferior to the previous REBUILD and EXTEND solutions. This is a very good result. In contrast the solutions in the first part of the numerical investigation are vastly different to the existing timetables and this is shown by the large time window violations. Time windows can easily be violated if an improvement to the makespan is possible using the two strategies in Part 1. If the existing timetable is fixed then fewer opportunities exist to reduce the makespan.

Table 5. Condensed relative error values for the makespan

Case Study	REBUILD		EXTEND		FIXED	
	BOV forbid	BOV Permit	BOV forbid	BOV Permit	BOV forbid	BOV Permit
1	1.84	0.41	1.17	0.41	0.97	0.47
2	1.46	0.48	1.18	0.07	1.87	0.29
3	1.75	0.13	1.75	0.13	2.56	1.81
4	4.59	1.03	1.63	0.39	2.10	1.82
5	2.52	1.70	1.58	0.95	2.92	2.92
6	11.69	9.40	3.92	3.11	1.65	1.35
7	6.33	2.54	5.05	2.12	3.63	3.41
8	2.46	0.92	1.47	0.47	1.63	1.32
9	5.03	2.44	3.56	2.00	4.16	3.78

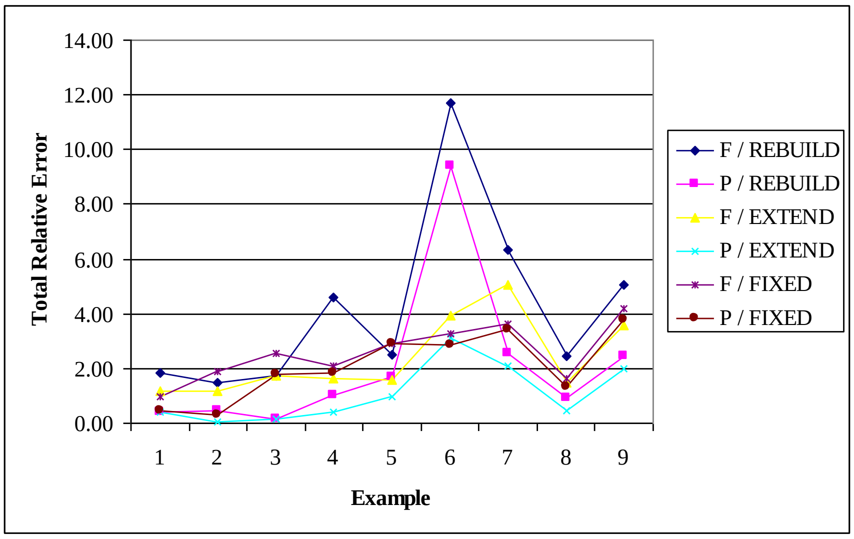

Figure 7. Condensed relative error values for the nine examples

The CPU times are comparable with those previously shown in Table 2.

4.2.2.2. Simulated Annealing

The SA results are summarised in this section. In order to obtain the summarised results the relative error values were firstly computed for each example and for each sub problem. For each example the five resulting values were added together to give an overall impression of what is going on. These values are shown in Table 6. In summary very little separates the results. The BOV permitted solutions are slightly better and the random selection strategy was generally best for SA. The SA algorithm was able to improve the CA solutions in most situations and quite well as indicated by the negative signs.

It should be noted that the "shift one position only" perturbation strategy may be overly restrictive on some problems involving fixed operations. For example an operation may need to be shifted past two or more "fixed" operations. The initial schedules that have been used in this paper highly utilise the railway and hence the insertion of additional trains can only occur toward the end of the original schedule. Consequently the shifting of operations by more than one position to the left or right in the machine sequences is not necessary here.

Table 6. Summarised SA results (relative errors) for the FIX strategy

Case Study	BOV not permitted				BOV permitted			
	SA1		SA3		SA1		SA3	
	RE(LB)	RE(CA)	RE(LB)	RE(CA)	RE(LB)	RE(CA)	RE(LB)	RE(CA)
1	0.49	-0.41	0.51	-0.40	0.44	-0.03	0.48	0.01
2	0.44	-1.04	0.44	-1.04	0.16	-0.13	0.16	-0.12
3	1.11	-0.83	1.23	-0.77	0.56	-1.18	0.56	-0.81
4	0.73	-0.93	1.14	-0.65	0.72	-0.98	0.68	-0.81
5	1.70	-0.77	1.75	-0.74	1.74	-1.00	1.71	-0.76
6	3.27	0.00	3.39	0.08	2.87	0.00	2.87	0.00
7	3.01	-0.35	3.16	-0.27	2.90	-0.34	2.92	-0.29
8	1.27	-0.27	1.22	-0.31	1.25	-0.06	1.11	-0.17
9	2.79	-0.72	2.83	-0.70	2.53	-0.83	2.53	-0.69

The CPU times are comparable with those previously shown in Table 3.

5. Conclusions

The problem of scheduling additional train services was considered in this paper. In order to solve this problem a process of iterative refinement was first proposed. More specifically it is a process of fine-tuning in which the demands of various customers / operators is negotiated through the alteration of time windows and fixing statuses. At each stage of this process a scheduling instance that is computationally intractable must be solved. This scheduling problem can be solved in a variety of ways and constructive heuristics and meta-heuristic algorithms were used in this paper.

The SATS scheduling problem is different from the usual timetable construction problem due to additional timing constraints. These timing constraints may be optionally viewed as “soft” or “hard”. In this paper both settings were provided for in our techniques and investigated in our numerical investigations. In the former case, the timing constraints of trains are primarily enforced by penalising the total time window violations in the objective. This approach does not guarantee that timing conditions are met however and allows any operation to be disrupted. In the second case, timing conditions may be strictly enforced by defining an operation as fixed and then disallowing any time window violation associated with fixed operations to occur. This mechanism of fixing an operation is very powerful yet is very simple. This mechanism addresses the SATS problem more directly than the general penalisation of time windows mechanism; it also encompasses periods of machine unavailability such as maintenance activities and enforced idle periods.

The numerical investigation consisted of two parts. In part 1 only time window constraints were used to enforce adherence to the existing schedule. In part 2 the existing schedule was strictly enforced by fixing particular operations. From numerical investigations it was observed that building the existing schedule from scratch using time windows is not effective. There are too many violations particularly after any perturbation is made. There is very little concrete information as to how to improve the solution.

In summary the approaches were shown to be quite effective and reasonably robust though it is clear from the numerical investigations that further improvements and fine tuning are possible. Simulated Annealing was shown to improve upon good quality starting solutions but not so well otherwise. Judging the quality of solutions when time window violations exist is difficult. Good solutions can be reached on larger problems only with a lot of extra effort, i.e. effort above and beyond what is required in most other scheduling problems. The literature in recent years also shows this. For example extra effort is required to run the meta-heuristic algorithms again and again until a good quality solution results. Very high level computing requirements are needed for problems such as this. We observed that the complexity of the SATS problem is reduced as more of the existing operations are fixed and fewer operations have time window constraints.

In our approach operations were scheduled using a non delay scheduling process. This approach however schedules operations as early or as late as possible depending on whether forwards or backwards scheduling is performed. An operation with a time window requirement that falls between the earliest entry and the latest entry or similarly the earliest exit and latest exit however can not be satisfied and will result in a time window violation. A secondary procedure to shift an operation for example by adding "artificial" release times is necessary in order to reduce time window violations. What improvement is possible is a source of future research as is the development of suitable procedures.

Multi-criteria scheduling was touched upon in this paper but more work could be done in the future. The complexity of the SATS problem ensured that a more conventional single objective criterion was used. A purely cost based objective could be more beneficial for the SATS problem. This topic also provides a topic for future research. Another avenue for future research is the incorporation of a cancellation mechanism in the SATS problem. For example if a service can not be scheduled at desired times then it may need to be cancelled and or replaced with another service.

REFERENCES

Aggoune R., 2004. Minimising the makespan for the flow shop scheduling problem with availability constraints. *European Journal of Operational Research* 153(3), 534-543

Asano M., Ohta H., 2002. A heuristic for job shop scheduling to minimize total weighted tardiness. *Computers and Industrial Engineering* 42(2-4), 137-147

Burdett R., Kozan E., 2006. Techniques for absolute capacity determination in railways. *Transportation Research Part B* 40(8), 616-632.

Burdett R., Kozan E., 2008. A sequencing approach for train timetabling. *OR Spectrum* doi:10.1007/s00291-008-0143-6.

Cacchiani V., Caprara A., Toth P., 2008. Freight transportation in railway networks. Published online at: http://arrival.cti.gr/uploads/Documents.0126/ARRIVAL-TR-0126.pdf.

Carey M., 1994. A model and strategy for train pathing with choice of lines, platforms, and routes. *Transportation Research Part B* 28(5), 333-353.

Carey M., Carville S., 2003. Scheduling and platforming trains at busy complex stations. *Transportation Research Part A* 37(3), 195-224.

Carey M., Crawford I., 2007. Scheduling trains on a network of busy complex stations. *Transportation Research Part B* 41(2), 159-178.

Ching-Jong L., Der-Lin S., Chien-Hung L., 2005. Makespan minimisation for two parallel machines with an availability constraint. *European Journal of Operational Research* 160(3), 445-456

D'Ariano A., Pacciarelli D., Pranzo M., 2007. A branch and bound algorithm for scheduling trains in a railway network. *European Journal of Operational Research* 183(2), 643-657.

D'Ariano A., Pacciarelli D., Pranzo M., 2008. Assessment of flexible timetables in real-time traffic management of a railway bottleneck. *Transportation Research Part C* 16(2), 232-245

Dessouky M., Lu Q., Zhao J., Leachman R., 2006. An exact solution procedure for determining the optimal dispatching times for complex rail networks. *IIE Transactions* 38(2), 141-152.

Goverde R., 2007. Railway timetable stability analysis using max-plus system theory. *Transportation Research Part B* 41(2), 179-201

Hall N. G. and Potts C. N., 2004. Rescheduling for new orders. *Operations Research* 52(3), 440-453.

Hendel Y., Sourd F., 2006. Efficient neighbourhood search for the one machine earliness-tardiness scheduling problem. *European Journal of Operational Research* 173(1), 108-119

Jain A.K., Elmaraghy H.A., 1997. Production scheduling/rescheduling in flexible manufacturing. *International Journal of Production Research* 35(1), 281-309.

Mascis A., Pacciarelli D., 2002. Job-shop scheduling with blocking and no-wait constraints. *European Journal of Operational Research* 143(3), 498-517

Mazzarello M., Ottaviani E., 2007. A traffic management system for real-time traffic optimisation in railways. *Transportation Research Part B* 41(2), 246-274

Rodriguez J., 2007. A constraint programming model for real time train scheduling at junctions. *Transportation Research Part B* 41(2), 231-245.

Schmidt G., 2000. Scheduling with limited machine availability. *European Journal of Operational Research* 121(1), 1-15

Sourd F., 2005a. Punctuality and idleness in just in time scheduling. *European Journal of Operational Research*, 167(3), 739-751

Sourd F., 2005b. Optimal timing of a sequence of tasks with general completion costs. *European Journal of Operational Research* 165(1), 82-96

Valls V., Perez M. A., Quintanilla M.S., 1998. A tabu search approach to machine scheduling. *European Journal of Operational Research* 106(2-3), 277-300.

Viswanath K.G., Appa I. S., Srinivasan G., 2006. Hierarchical minimisation of completion time variance and makespan in jobshops. *Computers and Operations Research* 33(5), 1345-1367

He Z., Yang T., Tiger A., 1996. An exchange heuristic imbedded with simulated annealing for due dates job shop scheduling. *European Journal of Operational Research* 91(1), 99-117

Zhou X., Zhong M., 2005. Bicriteria train scheduling for high speed passenger railroad planning applications. *European Journal of Operational Research* 167(3), 752-771.