INTEGER PROGRAMMING BASED APPROACHES FOR THE TRAIN DISPATCHING PROBLEM

Güvenç Şahin[1], Ravindra K. Ahuja[2] and Claudio B. Cunha[3]

(Revised: February 23, 2008)

ABSTRACT

Railroads face the challenge of competing with the trucking industry in a fast-paced environment. In this respect, they are working toward running freight trains on schedule and reducing travel times. The planned train schedules consist of departure and arrival times at main stations on the rail network. A detailed timetable, on the other hand, consists of the departure and arrival times of each train in each track section of its route. The train dispatching problem aims to determine detailed timetables over a rail network in order to minimize deviations from the planned schedule. We provide a new integer programming formulation for this problem based on a space-time network; we propose heuristic algorithms to solve it and present computational results of these algorithms. Our approach includes some realistic constraints that have not been previously considered as well as all the assumptions and practical issues considered by the earlier works.

Keywords. Transportation, train dispatching, space-time network, integer programming, heuristics.

[1] Manufacturing Systems/Industrial Engineering, Faculty of Engineering and Natural Sciences, Sabanci University, Orhanli, Tuzla, 34956 Istanbul, Turkey, guvencs@sabanciuniv.edu. Corresponding Author.

[2] Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611 USA, ahuja@ufl.edu.

[3] Department of Transportation Engineering, Escola Politécnica, University of São Paulo, São Paulo, SP, Brazil, cbcunha@usp.br.

1. INTRODUCTION

Railroads play a vital role in a country's economy by providing efficient and cost-effective freight services for the transportation of products and goods. In order to remain competitive, railroads continually seek to improve customer service, reduce transit times and operating costs, increase asset utilization, and minimize capital investments. Therefore, the rail transportation industry faces many planning and scheduling problems that can be modeled and solved with mathematical optimization techniques. If solved, these problems promise significant potential savings in costs and improved levels of service (see, for example, Assad 1980 and 1981, and Cordeau et al. 1998). However, the literature devoted to railroad optimization has experienced a slow growth in the past and, until recently, most contributions were dealing with simplified models or small instances that fail to incorporate the characteristics of real-life applications.

Freight railroads usually publish their train schedules with respect to an ideal timetable that concerns only the departure and arrival times at main stations, including origin and destination stations, as well as at intermediate stations where any type of scheduled service should occur. The *train dispatching problem*, also known as the *train meet-and-pass problem* or the *train timetabling problem*, aims to determine detailed train movements and timetables over a rail network under various constraints in order to minimize train deviations from the ideal planned schedule. Deviations or delays occur when trains traveling on the same track line either in opposite directions or in the same direction meet, thus requiring one of the trains to be pulled over for the other to cross or to overtake it, which is called a meet-pass. The rail network is composed of *corridors* that are sets of track lines connecting major stations. A corridor encompasses several intermediate points including *stations*, *sidings* and junctions, and single- or double-line track *sections* between such points. A single-line section is a track line segment that can accommodate only one train at a time. Similarly, a double-line section can accommodate up to two trains at a time, traveling either in the same direction or in opposite directions. A track section is delimited at each of its extreme endpoints by a station or a siding. A siding can accommodate at least one train waiting for crossing or overtaking while a station serves as a yard or a terminal where passing trains can wait for crossing or overtaking and where trains can depart from or arrive at.

It should be noted that these conflicts that lead to delays tend to occur in railroads where the dominant (or total) traffic corresponds to freight trains. Since these delays, due to trains being pulled over, are much less tolerated (or not at all) by passengers, sufficient infrastructure capacity is provided in order to avoid passenger trains to be delayed due to crossings and overtakes. On the other hand, for freight railroads the additional investment in tracks required to avoid or eliminate delays usually cannot be translated into increased prices to the railroad users in order to generate enough revenue to cope with these additional investments. Freight railroad users accept that delays due to traffic may occur and railroads tend to be slower than other modes of transportation in terms of total transit times (due to several factors, including delays) but can offer competitive and attractive lower prices for the services provided. As discussed in details later, this is a key issue at the strategic capacity planning level.

The detailed timetable (dispatching plan) consists of the departure and arrival times of trains at each intermediate point on their routes. Therefore, it also comprises the detailed information of all trains that are delayed due to conflicts with other trains, delay locations, and the start and end times of intermediate stops.

A feasible timetable honors the rail traffic constraints such as:

- track capacities,
- ensuring that train crossings and overtakes happen at sidings or stations,
- station and siding capacities, and
- minimum headway safety requirements.

In addition to the rail traffic constraints, freight railroads have to account for several practical constraints such as:

- Trains may not be delayed more than a certain time (referred as maximum delay allowances) in order to attain consistent schedules.
- Insufficient track length or the nature of the freight being carried may restrict the trains that can be stopped at a specific siding or a station (for instance, a train of chemicals or other hazardous products may not be allowed to pull over near a populated area).
- Some trains cannot be pulled over at some of the sidings because they may not have enough horsepower available to resume moving due to the siding's ascending slope.
- Lack of adequate infrastructure for crews may limit the maximum time a train can wait at a siding.
- Some trains such as passenger trains have to meet exact timetables including specified intermediate stops and may not be pulled over at all.
- Trains with same origin and destination stations arrive at their destinations in the same sequence as they depart.

The train dispatching problem arises in several contexts. In strategic planning, it relates to investment decisions in expanding railroad infrastructure such as building new sidings, extending current siding lengths to accommodate longer trains, and upgrading single-line segments to double-lines. In tactical planning, it consists of finding the best master schedule on a regular basis (weekly, monthly, or quarterly) with respect to train dispatching and timetabling decisions, which concerns determining when changes in the train schedules are required and when new trains should be scheduled (e.g., during season peeks) on the same railroad line. In real-time scheduling, it helps the dispatchers who are controlling the traffic to make sound decisions about which trains to stop and where, as updated data about train positions becomes available. In operational planning, dispatching plans that are updated every 2-12 hours provide the dispatchers with a guideline to aid in day-to-day operations. All these planning perspectives except for the real-time scheduling are closely related to each other at an abstract level in an attempt to represent the problem with a mathematical model. Nonetheless, the same mathematical model can be used for different purposes when enhanced with appropriate decision support mechanisms. For instance, it can be used as a what-if analysis tool to evaluate different investment options for strategic planning purposes. The same model can be used by a railroad to analyze the feasibility of a new schedule or the impact of inserting one or more new trains to the existing schedule at the tactical planning level. On the other hand, the dispatcher would like to achieve an optimal or near-optimal solution in the case of day-to-day operations using the same mathematical model.

In this paper, we report the development of a new modeling approach and related algorithms for solving the train dispatching problem that can yield high quality or near-optimal solutions in reasonable computational times. Our approach is flexible enough to incorporate a variety of practical constraints. We make the following contributions to the field:

1. We develop a new integer programming formulation of the problem based on a space-time network. The contribution with this formulation has two folds. First, it encompasses all relevant assumptions and issues considered in previous works. Second, it allows us to obtain optimal or near-optimal solutions for real problems in reasonable running times while considering practical constraints that have not been considered before.

2. We propose heuristic algorithms, based on the space-time network formulation that can be employed to solve some types of train-dispatching problems that have been considered earlier in the literature, especially those related to railroads whose predominant traffic corresponds to freight trains.

3. The heuristic algorithms are designed to handle relevant constraints for freight railroads that appear in practice. Some of these constraints have not been previously considered. Particularly, current state-of-the-art heuristic algorithms are not able to handle maximum allowable delays, restricted stopping stations for individual trains and constraints imposed by the restrictions of the physical infrastructure.

4. We present computational results of our approaches when applied to problems similar to those found in practice as well as some randomly generated problems. The heuristic algorithms allow us to obtain good or near-optimal solutions for both types of problems. On average, one of our heuristic algorithms provides solutions that are within 3% of the best solution for some heavily congested problems, and significantly better solutions than those that can be obtained by a commercial solver in much longer times. The outcomes are additionally confirmed by the results obtained for a real-world problem of a congested freight railroad.

To summarize, we present an integer programming formulation for the train dispatching problem that as a special case subsumes the previous formulations. We also use this formulation as a basis to develop several heuristic algorithms that capture most of the commonly used practical constraints. This study develops a framework for modeling and solving the train dispatching problems of generalized railway infrastructures; we demonstrate our findings for the case of bidirectional corridors.

2. Literature Review and Scope of the Problem

Several papers in the past are devoted to analytical line models whose main aim is to estimate train delays caused by interferences on a rail line due to dispatching policies, traffic distribution, and physical track topology. A good summary review of these papers can be found in Cordeau et al. (1998).

The following authors study the issues related to computerized train dispatching and computer-aided tools that assist planners in constructing dispatching plans: Sauder and Westerman (1983), Rivier and Tzieropoulos (1984, 1987), Peterson et al. (1986), Smith (1990), Churchod and Emery (1987), and Jovanovic and Harker (1991). Sauder and Westerman (1983) describe a computer-aided dispatching system developed at Norfolk Southern Railroad that consists of a partial enumeration scheme for generating effective train schedules on a single-track line. Jovanović and Harker (1991) propose the SCAN I system, a decision support model for scheduling of trains and track maintenance operations that help design reliable schedules in the sense that they are robust under unpredictable changes in operating conditions. The model, which considers a 24-hour time-

horizon, can deal with single-line and double-line tracks, and generate feasible, but not necessarily optimal meet-pass plans.

Carey and Lockwood (1995) study the timetabling problem in a setting that focuses on the passenger trains of European railway systems. This first article proposes algorithms and strategies for rail network structures with multiple one-way lines. In two related articles, Carey (1994a) and Carey (1994b) extend the findings of this study. The former extends the previous model to include the choice of sections and waiting platforms during the itinerary of the train. The latter considers two-way lines and generalizes the existing assumptions to those that are typical to North American railway systems.

Kraay and Harker (1994) propose a model for optimizing freight train schedules over the entire rail network with the intention of using them as part of a real-time control system. Its goal is to help coordinate train dispatchers by determining the target time for each train at major points of its itinerary. These findings can then be used in such dispatching models as the SCAN I system. The model, which is a large nonlinear, mixed-integer problem, directly considers the current position and relative importance of each train. The authors additionally propose a simple heuristic procedure and local search methods.

Higgins et al. (1996, 1997) study the problem of dispatching freight trains in a single-line track aiming to minimize the total weighted travel times. They formulate a mixed integer linear problem in which the arrival and departure times are modeled as continuous decision variables and the conflicts resolution as binary variables. In their first paper, they propose a branch-and-bound method; in the second one, they develop local search heuristics, genetic algorithms, tabu search, and related hybrid algorithms.

Cai et al. (1998) propose a greedy construction heuristic for timetabling and dispatching trains on a single-track railroad. This heuristic is an extension of a previous work of Cai and Goh (1994). It can handle the physical backups of the trains when necessary. In both works, the heuristic approach aims to resolve the capacity conflicts that arise when the trains follow their planned schedule by selecting the train(s) to traverse the conflict segment and the train(s) to be pulled over according to a greedy local criterion. The authors' approach is based on assigning trains to a position-time pair attributes at each time instant of the schedule horizon.

Adenso-Diaz et al. (1999) develop a real-time decision support system, which they designed for the Spanish National Railway Company. This study considers a passenger train system with a focus on service level; they attempt to satisfy the demands of a single corridor with three major lines while also solving the scheduling and dispatching issues.

Şahin (1999) develops a heuristic algorithm for rescheduling trains in conflicting situations on a single-track railway. The method is based on a systems approach in which conflicts with two trains are solved as they appear in time. To resolve a conflict, the algorithm selects to stop one of the trains by comparing a look-ahead measure that considers the potential conflicts and expected arrival times of the trains. This measure is based on analytical models that estimate average interference delays.

Kroon and Peeters (2003) develop models to construct cyclic timetables for the major Dutch operator of passenger trains. The system under consideration typically has a high frequency of repeated passenger trains running on one-way multiple track lines. The proposed models extend the existing models to consider variable trip times.

From a broader perspective, problems similar to the one we consider in this paper are addressed by Brännlund et al. (1998) and Caprara et al. (2002), since both deal with train timetabling in the presence of traffic conflicts that affect capacity. Brännlund et al. (1998) propose an integer programming formulation to obtain a profit maximizing timetable in which profit is measured by estimates of the value of running different services at specified times, and solve it using Lagrangian relaxation. Unnecessary waiting along the tracks are penalized and trains may not get scheduled at all. Caprara et al. (2002) consider the problem of determining periodic timetable for a set of trains traveling solely in the same direction on a corridor that consists of a single-line one-way track; the authors propose a time-space network representation of the problem that is similar to ours. However, some key differences to both articles should be highlighted: the approach proposed by Caprara et al. (2002) implicitly considers a scenario in which the predominant traffic corresponds to passenger trains (in fact, in their experiments they consider six different types of passenger services, from high-speed Eurostar to local trains) in the presence of some freight trains, all traveling in the same direction. Thus, this approach cannot be applied to the scheduling of a freight railroad, in which freight trains usually travel in both directions on a single-track line and crossings are frequent. According to Caprara et al. (2002), their problem is similar to the one addressed by Brännlund et al. (1998), which also implicitly focuses on situations in which the traffic of passenger train is predominant. In both cases, the business rules and the constraints differ quite significantly from a congested, single line freight railroad, in which trains must be pulled over and consequently delayed much more often than the typical scenario of passenger railroads, with trains running on multiple-line, single direction corridors. In this case, each corridor or direction could be dealt with independently. Therefore, although the mathematical model in Caprara et al. (2002) is more comprehensive and generalizes the approach by Brännlund et al. (1998), and both solution methods are suitable for real-life instances, they are not targeted to handle the several and frequent train meet-pass conflicts in opposite directions that arise in freight railroads. In addition, we claim that our network representation and the corresponding integer programming formulation not only allow the generalization of traffic constraints in Caprara et al. (2002), that is, track capacity in overtaking and time window constraints, but also incorporate several additional practical considerations for the more complex scenario of railroads in which the traffic of freight trains is dominant.

Most recently, Zhou and Zhong (2007) study a single-track environment in the context of a high-speed passenger rail line in an existing network in which the both the expected waiting times for high-speed trains and the total travel times of high-speed and medium-speed trains should be minimized. The authors propose branch-and-bound procedures with enhanced lower-bounding and novel upper-bounding schemes. Törnquist and Persson (2007) study the re-scheduling decisions under disturbances in real-time passenger train dispatch management for unrestricted number of segment on bidirectional railway lines. Both formulation approaches rely on the traditional and older version of assigning track usage rights to trains during a given time period rather than the newer network representation approach in Caprara et al. (2002). Törnquist and Persson (2007), in addition, provide a very compact analysis of the literature with a very precise classification scheme that organizes the previous studies based on the type of railway infrastructure, evaluation approach and the type of problems modeled and tested. According to their classification scheme, Caprara et al. (2002) should be classified as modeling a line corridor with unidirectional single-track segments and testing the same type of railway infrastructure while our study should be classified as modeling a network-type railway with bidirectional and unrestricted number of tracks on a segment with experimental tests performed on line corridors.

To summarize, the algorithms developed so far for the freight train dispatching problem rely on simplifications of the general problem in order to solve specific instances. Several more recent studies deal exclusively or predominantly with problems of scheduling or timetabling passenger trains, in which the nature of the problem, the function to be optimized and their constraints differ quite significantly from freight train

scheduling. For freight train dispatching problems, different mathematical formulations have been proposed as linear or non-linear optimization problems. In general, these problems have integer decision variables related to resolution of conflicts and continuous decision variables representing the departure and arrival times of each train at each meet-point. These problems, however, turn out to be large and intractable. The feasible solutions that are obtained with simplifications on the formulations are sometimes not even implementable.

In the remainder of the paper, we impose *maximum delay allowances* for trains, which play a significant role in running consistent and reliable schedules especially for congested single-line freight railroads, in which some passenger trains may be present among the predominant traffic of freight trains.. Maximum delay allowances specify a time windows for not only the departure (arrival) time of the train from (at) its origin (destination) but also the departure and arrival times of the train at all stations on its route. Maximum delay allowances have never been explicitly considered in previous studies, and they help us build the space-time network discussed in Section 3. Our modeling approach can also be used for problems where train speeds are not given a priori but rather selected from a set of discrete choices, and where the timetables are cyclic, although these issues are not necessarily related to our planning perspective as we describe previously. In general, the problems tackled in the literature consider a railroad line linking two major points and comprising single- and double-line segments. None of the problems explicitly consider the more general situation where a rail network may be composed of different track lines or the interactions between different trains traveling from different origins to different destinations that share parts of their routes. Decomposition approaches that deal with a network of rail lines may lead to good, near-optimal partial solutions for each track line. However, this may result in a poor global solution since the timetable in each track is made independently and it does not consider their impacts on the conflicts for other tracks. This may be particularly significant to network configurations with a main, central line that concentrates a significant amount of the traffic from different interconnecting lines (e.g. Y-shaped or fishbone-like). Similarly, some of the practical operational constraints mentioned above have never been considered, probably due to the difficulty of incorporating them into a mathematical formulation as well as into solution strategies, in addition to the fact that they do not necessarily arise in passenger railroads. On the other hand, it should be noted that our modeling approach does not aim to handle specific constraints that may arise in railroads in which the traffic of passenger trains is predominant, such as routing trains among different lines and paths, allocating trains to platforms, scheduling different types of traffic (intercity, local, etc.), reducing passenger inconvenience due to delays and transfers, and robust scheduling with respect to minor disturbances. To summarize, we believe that there are major opportunities for modeling real-world freight train dispatching problems to determine efficiently near-optimal meet-pass plans that consider the additional practical constraints, and are implementable. This study attempts to meet this need.

3. MATHEMATICAL PROGRAMMING FORMULATION

In this section, we first describe a space-time network representation. The train dispatching problem is then formulated as a multi-commodity flow problem on this network. For the sake of clarity, we first consider a corridor composed of both single- or double-track sections and a solution that honors maximum delay allowances and all traffic constraints except for the headways. Later, we discuss how other constraints are easily represented with the space-time network and do not even require any change in the mathematical model.

It should be noted that generalizing to a network of lines requires no major change. As illustrated in Figure 1, for the part of a railway network, which is not necessarily a corridor and contains both single- and double-track sections, we can still identify the sidings and physical connections between stations as we do in a single-

track corridor-type network. In addition, each train to be scheduled has a pre-defined route in terms of sequence of stations and track sections to follow. In most cases there is only one path linking each train's origin and destination; otherwise, each train's path is clearly defined in terms of stations and sections to follow. Therefore, contrarily to some passenger train problems, spatial issues like train pathing are not part of the decisions to be made. Therefore, generalization of the procedures for a corridor-type network can be accomplished by as a simple and straightforward extension of the network data structures we use. The mathematical model is based on the network model we next discuss and the implementation of the solution procedures requires no conceptual change at all for the generalized case.

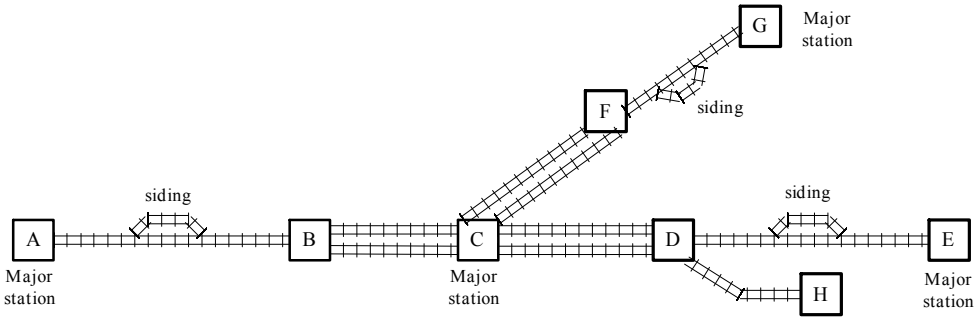

Figure 1. Part of a rail network with single and double-line tracks.

The network representation we propose is based on the concept of a train graph diagram, which is a standard method for displaying the resulting train schedules, meeting points, and associated delays. Many freight railroads still rely on train schedules that are made manually by skilled dispatchers using this kind of diagram. Figure 2 illustrates a sample train-graph schedule-diagram for a single-line track that links end- stations A and I. Intermediate meet-points are located at all the stations from B to H. The horizontal and vertical axes represent time and space, respectively. There are two northbound trains that move from A to I, and one southbound train. The southbound train is delayed twice, at meet-points E and C, as a possible solution for the conflicts with the two northbound trains. In other words, these delays allow northbound trains to meet and pass.

Even in this simple example, there are many possible combinations of sidings and times for trains to be pulled over in order to allow meets and passes, not to mention eventual changes in the departure times from the intermediate stations. Therefore, the train meet-pass problem is a very large-scale combinatorial optimization problem. When several trains in both directions are scheduled, many conflicts may arise. Each conflict involves trains moving either in opposite directions or in the same direction. Depending on the chosen solution for a conflict involving two trains (i.e. which train pulls over for the other to pass or overtake), the location and the time of later conflicts may change, new conflicts at different locations and times may arise, and existing conflicts may cease to exist. Thus, the number of feasible solutions to train meet-pass conflicts can be very large.

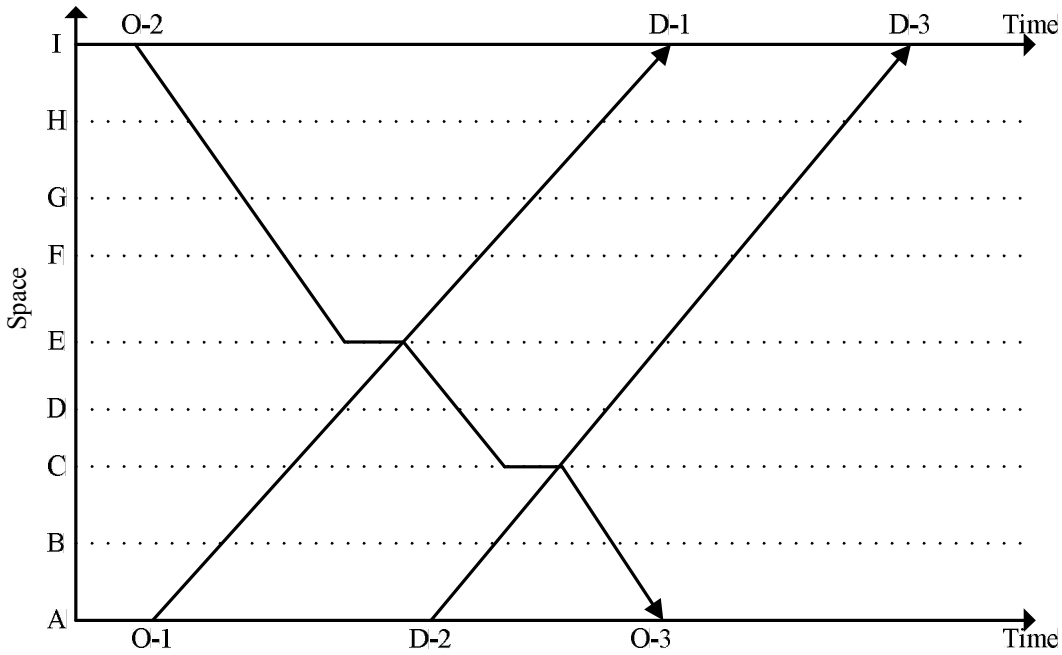

Figure 2. A train graph schedule diagram.

3.1 Notation and Definitions

On a single-line corridor, let $S = \{0,\ldots, s\}$ represent the set of stations and sidings, numbered according to the order in which they appear along the corridor from north to south. Let T be the set of trains traveling along S. Each train $t \in T$ may depart from any station $s_1 \in S$ and arrive at any other station $s_2 \in S$ ($s_1 \neq s_2$), which means that the trains may travel not only between the endpoints of the line (0, s) in any direction, but also may depart from and arrive at any intermediate station. The trains traveling from lower numbered stations to higher numbered stations (from north to south) are called outbound trains and represented with $T^{out} \subset T$. Correspondingly, the set of inbound trains is represented with $T^{in} \subset T$. Obviously, $T = T^{out} \cup T^{in}$. We define $L = \{0,\ldots, l\}$ as the set of sections (line segments) to represent the tracks between consecutive stations or sidings, where $l=s-1$ and s_i and $s_{i+1} \in S$ are the endpoints of section l. As mentioned previously, a single-line corridor is considered solely for the sake of clarity of the proposed network representation. The modifications for double-line track segments, as well as a network of tracks interconnecting with each other, are straightforward and require no modification in the proposed problem representation and in the formulation. We associate the following data with each train $t \in T$:

- d^t: departure (origin) station for train t,
- a^t: arrival (destination) station for train t,
- ed^t: departure time for train t according to the planned ideal schedule,
- md^t: maximum allowable delay for train t,
- ld^t: latest possible departure time for train t (considering the maximum allowed delay time),
- sa^t: scheduled arrival time for train t,
- la^t: latest possible arrival time for train t,
- f_i^t: travel time of train t along the track segment i (e.g., travel time from station i to station $i+1$ if t is an outbound train ($t \in T^{out}$); travel time from station $i+1$ to station i if t is an inbound train ($t \in T^{in}$)),

- *IS(t)*: the set of intermediate stations or sidings $s \in S$ where train $t \in T$ is allowed to pull over to wait for crossing or overtaking,
- *P(t)*: the ordered set of consecutive sections (d^t–s_1, s_1–s_2,.., s_k–a^t) that compose the path of train $t \in T$ from its departure station d^t to its arrival station a^t.

Obviously, $d^t \in S$, $a^t \in S$, $IS(t) \subset S$ and $P(t) \subset L$. Also, sa^t and la^t can be derived directly from ed^t, ld^t, md^t and f_i^t as $ld^t = ed^t + md^t$, $sa^t = ed^t + \sum_{l \in P(t)} f_l^t$ and $la^t = sa^t + md^t$, $\forall t \in T$.

It should be noted that travel times are assumed to be fixed and known a priori. These times are determined by ideal speeds that allow maximum efficiency in terms of fuel consumption for each train type and size on each track segment.

3.2 The Space-Time Network

We formulate the train dispatching problem as a multicommodity network flow problem (Ahuja et al. 1993) with additional constraints on the space-time network. The space-time network representation has been traditionally used to model problems when there exists a time dimension. Similar network representations have been discussed in Caprara et al. (2002) and Schrijver (1993) in the railroad context. Each train functions as a commodity in the network. We represent the space-time network as $G = (N, A)$, where N denotes the node set and A denotes the arc set. The space-time network contains three types of nodes. *DepNodes* represents the set of artificial nodes in which the outflow generate departure of a train from its origin. There exists a departure node for each train $t \in T$ denoted as Dep^t. Therefore, the supply of each node in this set is one. The *ArrNodes* set corresponds to the set of artificial nodes in which the inflow represents the arrival of a train to its destination. In a similar way, the demand of an arrival node Arr^t is set to one for all $t \in T$. *StatTimeNodes* is the set of nodes with two attributes: place and time. Time is discretized into discrete time instants with equal time intervals between consecutive time instants. Let the set of time instants be defined as $Q = \{1, 2, \ldots, q\}$. For instance, if a daily schedule (24 hours) is discretized in 5 minute periods, then q=288. Nodes of the set *StatTimeNodes* correspond to the copies of each station node $s \in S$ in each time period $k \in Q$, in other words, *StatTimeNodes* corresponds to the set $S \times Q$. An element of this set is denoted as i_k, where $i \in S$ and $k \in Q$.

The set A of arcs is composed of five subsets: origin arcs, destination arcs, outbound travel arcs, inbound travel arcs, and waiting arcs. The arcs in the set of origin arcs emanate from nodes in *DepNodes*, and destination arcs enter the nodes in *ArrNodes*. To represent the departure of a train t from its origin station d^t, we create arcs from node Dep^t to nodes d_k^t for each time period k where $ed^t \le k \le ld^t$. Similarly, to model the arrival of a train t at its destination station a^t, we create arcs from nodes a_k^t to node Arr^t for $sa^t \le k \le q'$, where $q' = \min\{q, la^t\}$.

The travel arcs (*TArcs*) and waiting arcs (*WArcs*) of the space-time network are created on a train-by-train basis. For train t, we follow the sequence of stations and sidings in *IS(t)*, and segments in *P(t)* on the train's route, starting from its origin station d^t through its destination station a^t. The set of outbound travel arcs include the arcs from node i_k to a consecutive node $(i+1)_l$, where $k+1 \le l \le q$. To model the possible movement of an outbound train t from station i to station $(i+1)$ at time k, we create an arc from each node i_k to node $(i+1)_l$,

where $l = k + f_i^t$ if i_k has an incoming arc of train t. Similarly, the set of inbound train arcs include the arcs from node i_k to node $(i-1)_l$, where $k+1 \le l \le q$. To model the possible movement of an inbound train t from station i to station $(i-1)$ at time k, we create arcs from each node i_k to node $(i-1)_l$, where $l = k + f_{i-1}^t$ if i_k has an incoming arc of train t. The set of waiting arcs for train t includes the arcs from node i_k to node i_{k+1} for all stations $i \in IS(t)$. Whenever i_k has an incoming arc that represents a possible movement of train t, we create consecutive waiting arcs on the same station level by considering the maximum possible delay of train t at that station. An example in Figure 3 shows a subset of the arcs defined for a train scheduled to depart from Station 0 at time t and arrive at Station 2 at time $t+2$ with a maximum delay allowance of 3 time periods.

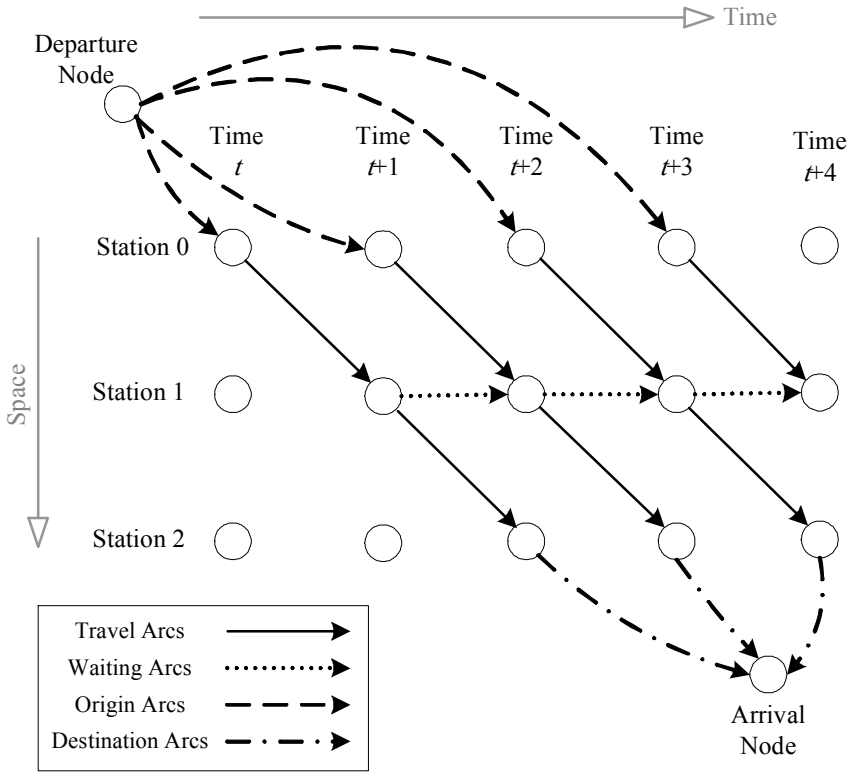

Figure 3. Time-space network representation.

All arc capacities are set to one since each arc represents flow of one train. The costs of waiting arcs are set to the length (in minutes) in which time is discretized since each arc represents a train waiting at a particular siding from time k to time $k+1$. Similarly, costs for origin arcs correspond to the respective lengths of the initial delays. In other words, the origin arc that emanates from the departure node (Dep^t) and enters a departure station node at time period k ($ed^t \le k \le ld^t$) corresponds to a delay of ($k - ed^t$) time periods multiplied by the length of the discretization in minutes. All other arc costs are zero. Delays can also be multiplied by specific train weights in order to reflect relative priority of certain trains.

3.3 The IP Formulation

The problem is formulated as an IP problem with integer (unit-) flow decision variables $x^t_{i_k j_l}$ representing the flow of train t on arc (i_k, j_l) where $t \in T$ and $(i_k, j_l) \in A$. Note that for a given train t, each flow variable is defined only for feasible moves corresponding to one of the five arc sets, as described in Section 3.2. The IP formulation is as follows:

$$\text{Minimize} \quad Z = \sum_{t \in T} \sum_{(i,j) \in A} c_{ij} x^t_{ij} \tag{1}$$

$$\text{subject to} \quad \sum_{ed^t \le k \le ld^t} x^t_{Dep^t d^t_k} = 1 \qquad \forall t \in T \tag{2}$$

$$\sum_{sa^t \le k \le la^t} x^t_{a^t_k Arr^t} = 1 \qquad \forall t \in T \tag{3}$$

$$\sum_{i \in StatTimeNodes} x^t_{ij} - \sum_{i \in StatTimeNodes} x^t_{ji} = 0 \qquad \forall t \in T,\ \forall j \in StatTimeNodes \tag{4}$$

$$\sum_{t} x^t_{ij} \le u_i \qquad \forall (i,j) \in WArcs \tag{5}$$

$$\sum_{t \in T^{out}} \sum_{l \le k-1} \sum_{m \ge k} x^t_{i_l (i+1)_m} + \sum_{t \in T^{in}} \sum_{l \le k-1} \sum_{m \ge k} x^t_{(i+1)_l i_m} \le 1 \qquad \forall i \in S, \forall k \in Q \tag{6}$$

$$x^t_{ij} \in \{0,1\} \qquad \forall (i,j) \in A,\ \forall t \in T \tag{7}$$

The objective function (1) minimizes the total delay of trains since the arc costs represent the delays in discretized time units. Constraint set (2) ensures that for each train t there is a unit outflow from its departure node to one of the nodes that is the copy of its origin station at a time within its possible departure time window. Similarly, constraint set (3) imposes that for each train t there is a unit flow from one of the nodes that is the copy of its destination node to its arrival node within its arrival time window. Constraint set (4) provides the flow conservation constraints for the nodes in *StatTimeNodes*, for which the demand and supply is zero. Therefore, any flow entering these nodes should leave. Constraint set (5) ensures that at any time interval, the number of waiting trains at a station or a siding do not exceed the capacity of the station or the siding, u_i, where $i \in S$. Constraint set (6) is the track capacity constraints. These constraints guarantee that at any time interval there is no more than one train traveling on a track section. This is accomplished by ensuring that, at most one of the arcs passing through the same time interval at a particular track section carries a unit flow. Figure 4 illustrates track section AB and arcs of the space-time network that represent different trains that can travel along this section during the time intervals from discrete time instant 0 to discrete time instant 6. For this section, we should ensure that for each time interval, the sum of the flows on arcs passing through the same time interval should not exceed 1. This results in six constraints, one for each time interval, the first three of which are exemplified as follows:

- for time interval 0-1: $x_{A0,B2} + x_{A0,B4} + x_{B0,A2} + x_{B0,A3} \le 1$,
- for time interval 1-2: $x_{A0,B2} + x_{A0,B4} + x_{A1,B3} + x_{B0,A2} + x_{B0,A3} \le 1$, and
- for time interval 2-3: $x_{A0,B4} + x_{A1,B3} + x_{A2,B5} + x_{B0,A3} + x_{B2,A3} \le 1$.

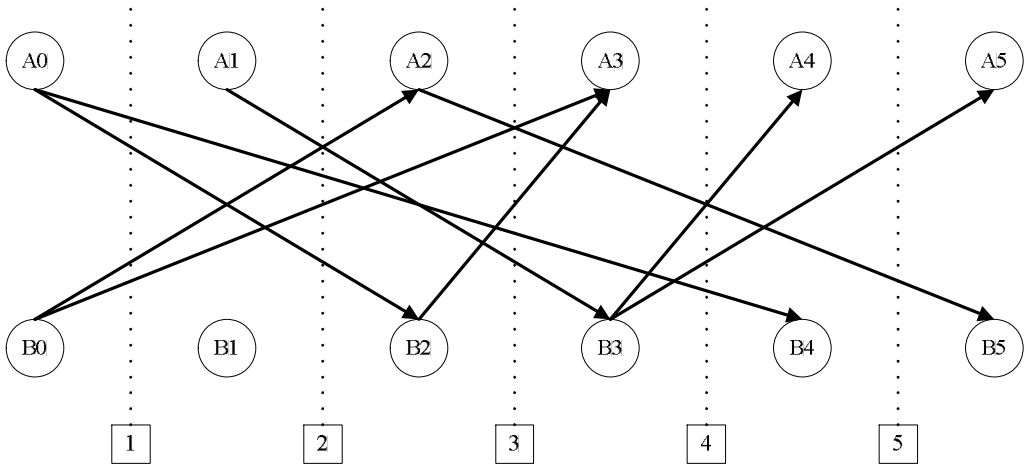

Figure 4. Track capacity constraints.

The first term in constraint set (6) represents the sum of flows on the arcs in the outbound direction (from station i to station i+1) during time period k. The second term represents the flows on the arcs in the inbound direction (from station i+1 to station i) during time period k. In total, we need to write such constraints as much as the number of tracks multiplied by the number of discrete time instants (i.e. $(|S| - 1)*|Q|$).

To this end, as mentioned before, in view of the literature, the problem that we are dealing with is the most generalized form of the train dispatching problem in which the focus is mainly on North American railway systems, usually freight trains. Carey (1994b) considers the problem in a similar setting as ours. However, the mathematical model in this study is based on a set of binary variables that represent the precedence relation of two trains on a link, which magnifies the number of variables when compared to our formulation. The space-time network representation here might share some characteristics with the problem representation in Caprara et al. (2002); however, as mentioned earlier, the authors address a single-line corridor along which trains travel in only one direction. Therefore, their resulting formulation cannot be generalized for the setting we study here. In addition, Caprara et al. (2002) focus on the passenger trains. In the next two sections, we discuss some constraints that are practical but skipped for the time being to provide a clear understanding of the network representation and the formulation, and some other so-called complicating issues and their remedies according to our modeling framework.

3.4 Practical Constraints

Some of the practical constraints in railroad practice are not explicitly addressed in the above formulation. Nonetheless, our network representation and the associated IP formulation provide a framework to incorporate these constraints without adding new constraints to the formulation. In the following subsections, we list such practical constraints and discuss how to handle them only with slight modifications in the network representation instead of adding new constraints.

3.4.1 Minimum Headways

The most crucial of these constraints are the so-called *minimum headways constraints*. For safety purposes, trains departing (arriving) from (at) a station one after each other and trains following each other have to keep a

minimum clearance distance between them which can be represented with respect to either time or track length. We here discuss how to handle these two types of minimum headways:

- If the headways are represented with respect to track length, say r miles, then we create artificial nodes with no train holding capacity every r miles between two intermediate points along the corridor as necessary while constructing the network. For instance, let $r = 20$ miles and three stations be at miles 40, 65 and 80 along the corridor. When we create an artificial node at mile 60, track capacity constraints would make sure that no train leaves the first station during a time interval when there is another train occupying the section between the first station and the artificial node at mile 60. On the other hand, after a train passes through the artificial node, a new train can depart from the first station to second station without violating the minimum headway. It is easy to observe that we do not need to create any artificial node between the second and the third station as the track length is shorter than 20 miles and the track capacity constraints would automatically satisfy the minimum headway.

- If the headways are represented with respect to time, say h minutes, then we only need to be careful in selecting the length of the time interval between discrete time instants of the network representation. If the selected equivalent time intervals between discrete time instants is larger than h, the track capacity constraints would honor the minimum headway requirements. In this case, the track capacity constraints would automatically not allow two trains in the same direction to travel on the same section with a time distance less than h. Otherwise, we need to write multiple additional constraints of type (6) for each section in each time interval following a simple enumeration procedure. Particularly, for each travel arc, we first enumerate all other arcs with which it cannot share the same time interval (due to the headway length), then add a constraint that prevents any of these other enumerated arcs (which represents other trains traveling at that time interval) to carry a positive flow. In this case, for any inbound (outbound) travel arc, the outbound (inbound) that pass through the same interval appear in its headway constraint. Among the same direction arcs, however, only those that emanate from a node of the same station but within a time interval less than h should appear in the constraint.

Minimum headway constraints for trains that meet can also be easily handled. In this case, one pulled over train cannot depart from a station before a given amount of time has elapsed since the departure of the other train traveling in the opposite direction. To handle this situation, we simply add constraints that prevent two travel arcs emanating from the same node at the same instant in opposite directions to carry flow, in a similar way to the track capacity constraints (i.e., constraints of type (6)). However, our experience shows that this type of "meet headway" constraint is unnecessary in most cases for freight railroads, since in a meeting, the train that is not pulled over does not stop at the station or siding, contrarily to passenger trains in which both trains, not only the one waiting, may have to stop momentarily at a given station. Also, freight trains usually travel at relatively low speeds (i.e.,below 60 km/h), and these headways, usually around 2-3 minutes for freight trains, are, in most cases, shorter than the minimum interval in which times are discretized. Thus, this constraint, which may arise in the context of real-time scheduling, may require some manual minor adjustments to the schedule by the dispatcher, with no major impact given the length of track segments and the corresponding travel times are elevated when compared to these headways.

3.4.2 Terminal Track Lengths, Siding Lengths and Restricted Stops

In practice, not all trains are allowed to stop at any station or siding on their route. The foremost reason for this may be due to infrastructure of stations and sidings. While waiting at a station, trains are held at the *terminal*

tracks of the station. These tracks vary in length from station to station. Therefore, some tracks may not be long enough to hold some trains. Similarly, sidings (usually designated as a single track) may not be long enough to hold all trains that pass through. To represent such restrictions in our model, we need to be careful in constructing the set of waiting arcs for a train. Particularly, if a train is longer than a siding length, we do not create any waiting arcs for that train at that siding. The same idea applies to stations with a single track or multiple tracks of same length. In the case of multiple terminal tracks with various lengths, a train may be short enough for some tracks at the station but not for the others. We can still handle this restriction within the network representation where we create one waiting arc for each track that is long enough and avoid creating waiting arcs for the shorter tracks. Considering the constraint set (5) for a station, instead of writing one constraint with the right-hand side u_i, we write a separate constraint for each terminal track at the station with right-hand side equal to 1.

Other types of restrictions prohibit particular train delays at particular stations. One restriction might be due to the type of material being carried by the train. For instance, trains carrying hazardous materials may not be allowed to stop at stations in proximity to populated areas. Another reason might be related to locomotives of restricted power capacity. A locomotive or a consist of locomotives, depending on the size and weight of the railcars it pulls, may not be able to resume moving the train after being pulled over at certain sidings or stations due to slope of the track lines. In short, various types of restrictions can be easily handled with the same idea discussed in the previous paragraph.

3.4.3 Explicit Time Windows and Implicit Time Windows Constraints

Caprara et al. (2002) discuss two types of constraints that might be considered in the set of practical considerations. Explicit time windows constraints require that a train arrives at a particular station no later than a given time, or depart from a station not earlier than a given time. Our method for creating travel arcs of a train at a station within a set of time intervals (imposed by the maximum delay allowances) generalize this type constraints for all stations on the train route. In the case of more specific time windows, the set of travel arcs emanating from or entering a station node is created in a more restricted manner.

Implicit time windows constraints, on the other hand, favor a time interval (or a set of time intervals) over the others among the set of possible arrival or departure times of a train at a station. Although this is usually not a choice of operating principle in freight railroads, we can still represent these considerations in our model. In this case, as some of the travel arcs are preferred over the others, we simply penalize the non-preferred ones by attaching positive arc costs. However, the penalty scheme should be applied carefully keeping in mind that the actual objective function penalizes only the delays. Therefore, instead of penalizing those arcs in proportion to their corresponding travel times, we may apply a scheme that downgrades the arc travel times to a particular ratio. A more general penalty scheme similar to the profit model in Brännlund et al. (1998) prefers a particular arrival or departure time and gradually penalizes the times that are earlier or later. For instance, a train may depart from a station at time instants $k_1, k_2, \ldots, k_m$ with respect to its maximum delay allowance. The most preferred departure time for this train at this station is given as k_i, $i \leq m$. Then, we can attach penalty costs to these arcs such as $c_i = 0$, $c_1 > c_2 > \ldots > c_i = 0$ and $c_m > c_{m-1} > \ldots > c_i = 0$.

3.5 Other Issues on Dispatching Plans

Although it is out of the scope of the planning perspective we deal with, there are other planning issues addressed in the literature and in railroad practice such as constructing cyclic timetables and allowing the possibility of trains with varying speeds.

To demonstrate how we can use the same network representation for cyclic timetables, let us take an example of 24-hour daily periodic timetable that starts and ends at midnight. Let us discretize the time in 5-minute periods. Since the planning horizon is cyclic, the first time instant and the last time instant are the same and equal to 12:00 AM corresponding to time instant 0. Then, in the time period set Q, q corresponds to 11:50 pm and is equal to 286. Consider a train that departs from station i at 11:30 pm (corresponding to time instant 282) towards station j of which the travel time is 1 hour, and arrives at 00:30 (corresponding to time instant 7). The corresponding travel arcs emanates from the station node with space attribute i and time attribute 282 and enters the station node with space attribute j and time attribute 7. Therefore, a cyclic timetable is easily represented with a little twist by matching the first time instant with the last, and we do not need to change the IP formulation at all.

To allow trains travel with varying speeds is again a matter of network representation. All we need to do in this case is to create an alternate set of travel arcs for a train out of a node in *StaTimeNode*. To go with an example again, let us assume a train may depart from station i towards station j at time instants $k_1, k_2, \ldots, k_m$. The travel time of this train along this section is given as f^1, f^2 and f^3 with respect to three discrete speed choices respectively v^1, v^2 and v^3, where $v^1 > v^2 > v^3$. In this example, we create travel arcs for this train from nodes with place attribute i and with time attribute k' to nodes with place attribute j and time attribute k'' if $k'' = k' + f^1$ or $k'' = k' + f^2$ or $k'' = k' + f^3$ as long as these nodes are among the possible station-time combination according to maximum delay allowances.

4. HEURISTIC ALGORITHMS BASED ON THE IP FORMULATION

Although each study in the literature focuses on a particular version of the problem, the train dispatching problem in its general form is designated as a NP-hard problem. As expected, solving the problem with a commercial IP solver reveals that even real-life problems cannot be solved to optimality with reasonable computational effort. In order to confirm this, we first attempt to solve some problem instances with CPLEX version 8.1, setting all MIP solver parameters to their default values. Our experiments show that it is highly unlikely to obtain optimal solutions even to moderate size problems. Thus, in this paper we present heuristic ideas that benefit from the IP formulation. Instead of conventional heuristic approaches similar to Cai et al. (1998) and Şahin (1999) that rely on greedy selection heuristics and LP-based techniques similar to Higgins et al. (1996), Brännlund et al. (1998) and Caprara et al. (2002), our heuristics rely on the IP formulation we develop. The first heuristic can be classified as a search technique, not necessarily according to general understanding, which progresses through feasible solutions of restricted versions of the problem. The second idea is inspired by the dispatcher's decision-making process and constructs a feasible solution by resolving the train conflicts in a chronological order.

4.1 An IP-Based Heuristic

This heuristic is based on our observation of the performance of CPLEX version 8.1, well-known commercially available optimization software, to solve different instances of the train dispatching problem. We realize that for moderately small instances that are measured in terms of the number of sidings, the length of the time horizon, the number of trains, and the number of potential conflicts, CPLEX performs well, reaching the optimal solution in a few minutes. An important aspect of the formulation is that larger maximum delay allowances lead to larger problems by increasing the number of arcs in the network as well as the complicating capacity constraints (i.e., constraint set 6) that compromises the network structure of the problem. Thus, decreasing maximum delay allowances tightens the formulation without affecting the feasibility. As the size of the problem increases and/or as longer maximum train delays are allowed, CPLEX fails to quickly deliver the optimal solution. We have also observed that a feasible solution in general can be found in a reasonably short amount of time, usually in less than a minute.

In their day-to-day operations, train dispatchers face an overwhelmingly dynamic environment; unexpected and unpredictable events occur all the time, causing trains to be delayed. Therefore, a major effort to determine a global optimal solution for all trains over the whole scheduling horizon (typically 1-2 days) may not be worth the effort. The dispatcher, under this scenario, sacrifices the impact of conflict resolution decisions in the long term. On the contrary, the sub-optimal decisions of the dispatcher favor the feasibility in the long term and good quality decisions in the short term. Our heuristic idea tries to repair this drawback by integrating this planning approach into the IP formulation. On one hand, we progressively tighten the formulation by shortening the maximum delay allowances. On the other hand, we constantly make sure that a feasible solution exists even with shorter maximum delay allowances, and pay special attention to short-term conflicts of the planning horizon.

The key idea behind this method consists of gradually reducing the maximum total allowable delays (md^t) for all trains based on new feasible integer solutions that are progressively obtained. Initially, we divide the scheduling horizon q into two periods, the short and the long terms. Let q_s be the length of the short term ($q_s \leq q$); consequently $q_l = (q - q_s)$ define the remaining length of the scheduling horizon (i.e. the long term). We define δ_s and δ_l as increments to the train delays obtained in a feasible solution for the short and long terms, respectively. Suppose we can quickly find a feasible solution in which a train t is delayed cd^t units of time ($cd^t \leq md^t$). The idea is to make the mathematical formulation progressively tighter, in order to make it converge faster to a good solution, possibly optimal or near-optimal. In this case, for train t the maximum allowed delay is reduced from md^t to the minimum of ($cd^t + \delta_s$) and md^t for the time periods within the short term. Similarly, maximum delay is reduced from md^t to the minimum of ($cd^t + \delta_l$) and md^t in the long term where $\delta_l \leq \delta_s$. Note that the maximum delay allowed for the short term is still superior to the total initial delay obtained along the whole scheduling horizon. Therefore, a feasible solution is always guaranteed and the problem is tighter. The optimization process is then re-started for this tighter problem and proceeds for a fixed running time period σ. With this method, the chances that a better feasible solution is found in a shorter amount of time are increased. The same process is repeated until either an optimal solution for the progressively tightened problem is found or a maximum total CPU time is reached. Figure 5 formalizes our IP-based heuristic for the train dispatching problem.

```
algorithm IP-Based Heuristic for TDP;
begin
    find the first initial feasible solution for the IP formulation using CPLEX;
    while the current solution is not locally optimal or CPU time limit is not reached do
        for each train t
        set md^t = min (d^t + δ_s , md^t) for the short term (periods 1-q_s);
        set md^t = min (d^t + δ_l , md^t) for the long term (periods q_s+1, q);
        end ; {for each train t}
        process CPLEX for σ seconds and find the current best feasible solution;
    end;
end;
```

Figure 5. The IP-based heuristic for the train dispatching problem.

4.2 A Simulation-Based LP-Greedy Construction Heuristic

This algorithm constructs a feasible solution so that the conflicts in the planned ideal schedule are solved in chronological order according to a selection criterion. To resolve a conflict between two trains the algorithm selects which train to stop and which train to pass/overtake, and this decision is based on the criterion. The algorithm proceeds maintaining a feasible schedule until the last resolved conflict, and it terminates when it finds a feasible schedule. However, as the conflicts are progressively resolved, the remaining conflicts in the original schedule are not valid anymore. In other words, as the train schedules are disturbed due to conflict resolutions, the conflicts in the schedule may change in time and space. The dynamicity of the schedule implies the dynamicity of the information with which the algorithm works; we need a technique to handle this. Similar to the approach in Şahin (1999), we use a method based on the discrete event simulation technique. Instead of score-based selection criteria, we use the LP relaxation of the IP formulation to predict which stopping decision leads to a better dispatching plan for the remainder of the planning horizon after this conflict. Before proceeding to the details of the algorithm, we need to state the differences between the previous algorithm by Şahin (1999) and ours.

First, in our problem definition, the trains have a maximum delay allowance that cannot be exceeded in a feasible schedule. The choice to resolve a conflict is not only dependent on the criterion used to resolve the conflict but also on feasibility. Consider the following situation. Trains t_1 and t_2 meet at section s at time k. However, neither train can be any further delayed, because they both have consumed their maximum delay allowances according to the feasible schedule maintained up to that time. Therefore, it is not possible to obtain a feasible schedule for the rest of the planning horizon since none of the trains can be delayed anymore. Our algorithm is designed in such a way that when it discovers an infeasible conflict, the algorithm goes back on the timeline to the time corresponding to the last conflict involving either train t_1 or train t_2 (say time $k_1 < k$). It then reverses the simulation procedure moving all trains backwards until time k_1, and then changes the resolution of the conflict at time k_1, and finally (re)proceeds forward solving the conflicts that follow starting from time k_1.

Second, to resolve a conflict, Şahin (1999) projects the conflicts after the current conflict time according to the different choices. We make the projection by approximating the likelihood of happenings in the future. Particularly, we employ an approximate optimization scheme, the LP relaxation of the real problem, to two future scenarios based on two present choices. In our case, projecting the future would be a computational burden because of the maximum delay allowances. We use two special data structures: an event list that handles

the simulation process and a conflict list that provides the simulation with the ability to go backwards on the timeline.

The steps of the algorithm where a generalized selection rule is applied can be summarized as follows:

Step 0. Set the current conflict number to n=0. Initialize the event list with the departure event for each train from its origin station at the scheduled origin time;
0.1. If there are trains that have not yet arrived at their final destinations, proceed to *Step1*;
0.2. Otherwise, stop.

Step 1. Using the event list, advance trains on the timeline until either a pending conflict is found or all the trains have arrived at their destinations;
1.1. If no conflict is found and all trains have reached their destinations, stop; a feasible solution has been found.
1.2. Otherwise, set the current conflict to n=n+1. Check if this conflict can be solved;
1.2.1. If at least one of the two trains can be pulled over without violating maximum delay allowances, proceed to *Step2*.
1.2.2. Otherwise, proceed to *Step3*.

Step2. Select the train to be pulled over based on the selection criterion;
2.1. If no feasible solution can be found for the current conflict, go to *Step3*.
2.2. Otherwise, update the departure time of the delayed train in the event list and return to *Step1*.

Step 3. Reverse the event list back in time until the most recent prior conflict involving any of the trains in the current conflict n is found, say conflict r ($r < n$). Is changing the choice for delayed train in conflict r feasible?
3.1. If yes, return to *Step1*.
3.2. Otherwise, restore the most recent prior conflict as the current conflict ($n = r$) and repeat *Step3*.

The algorithm is flexible in the sense that it allows the use of various alternate selection criteria in *Step2* to resolve a conflict involving two trains that meet (i.e, crossing or overtaking). We only discuss a LP relaxation-based greedy selection rule. Let the trains in the conflict at time t be i and j.

- Construct an LP-relaxation of the IP formulation given in Section 3.
- Set the decision variables fixed to 0 or 1 according to the feasible part of the schedule (i.e. the maintained schedule up to time t).
- Solve the LP relaxation, LP_i, for which train i is given priority and can occupy the conflict section at time t while train j is delayed. Let the resulting objective function value of LP_i be Z_i.
- Similarly, solve the relaxation LP_j in which train j is given priority; thus, the corresponding variables that allow train j to occupy the conflict section at time t are set accordingly. The corresponding objective function value is Z_j.
- If $Z_i \leq Z_j$, delay train j. Otherwise, delay train i.

We emphasize that LP-based selection rule also allows a later infeasible conflict to be detected if both LP_i and LP_j are infeasible. This feature, however, would not exist if the selection were based on a scoring scheme of the trains involved in the conflict.

5. COMPUTATIONAL RESULTS

In this section, we report the results for some experimental problems using the solution methods that we propose. We report results for three different set of problems:

1. *Congested test problems*: We test our heuristic methods on randomly generated, congested problem settings. We use this set to analyze how the size of the problem (determined by the number of trains, number of stations and the magnitude of maximum delay allowances) affects the difficulty level.

2. *Synthetic test problems*: We generate a second set of problems that are representative of the real-life problems with three different congestion levels. By keeping length of the planning horizon constant over all instances, we test the performance of the heuristic algorithm for the different congestion levels and different choices of the time interval selection in constructing the space-time network.

3. *A real-life problem*: We solve a real-life problem and compare our results with the existing solutions.

In addition to computational results for the IP formulation with CPLEX and proposed heuristics, we report results for the IP formulation where CPLEX running time is limited. This is mainly done to help us understand the performance of the heuristic algorithms when it is very difficult to obtain even a feasible solution with the methods that are used currently in practice. All CPU times are in seconds.

5.1 Congested Test Problems

We have created five problems varying by number of trains and stations. All of these problems are highly congested compared to real-life cases in the sense that the congestion can easily be observed throughout the entire timeline by the number of conflicts that arise once the original schedule diagram is observed. An important characteristic of the problem corresponds to the maximum delay allowances that change the size of the IP formulation together with the number of trains and number of stations along the corridor. The maximum delay allowances of the five original problem settings are created randomly based on a known feasible schedule for each of these problems. These maximum delay allowances vary from 25% to 75% of a train's ideal total travel time. Then, we modify these problems by fixing the maximum delay allowances of the trains to much longer maximum allowed delays, that is, 50% and 100% of their ideal travel time. In total, we have five different problem settings (number of trains varying from 17 to 30, and number of stations varying from 9 to 20), and each problem is modified with respect to the maximum delay allowances in three different ways.

Tables 1 and 2 show the results of the experiments with this set of randomly generated problems. All computational times represent the actual times spent by a Pentium 4, 3.0 GHz PC with 1,024 MB of RAM. Heuristic gaps are calculated as the percentage deviation from the best integer feasible solution (not necessarily optimal) obtained from the CPLEX solution of the IP formulation. All gap figures are calculated based on the optimal (or the best feasible if the optimal solution is unavailable) objective function value. In Tables 1 and 2, a problem instance is described by a triple "*t-s-%*," where *t* corresponds to the number of trains, *s* corresponds to the number of stations, and *%* corresponds to the degree of the maximum delay allowances (where *r* represents random allowances from 25 % to 75 %). The column 'Solution' in both tables correspond to the objective function value in discrete time-periods of 15-minute length, which are compatible with 24-hour horizon usual in the planning level.

In Table 1, the column "Infeasibilities" denotes the number of times that the simulation-based approach detects infeasibility (as mentioned in Section 4.2) and goes back on the timeline. This figure should be used as a reference for unexpectedly high computational times when they are different from zero. We test the IP-based heuristic with different times per iteration from 1 second to 20 seconds, as presented in Table 2. The first row shows the time per iteration in parentheses for each column.

The results for the IP formulation show that the tightness of the problem significantly affects the performance and the size of the network as well. As the maximum delay allowance is increased, the network grows larger and the problem loosens. This makes it difficult for CPLEX to confirm an optimal solution even when it is reached early in the branch and bound procedure. However, when we follow the progress of branch and bound iterations, we observe that most of the time is spent verifying the optimality. In most cases, an unverified optimal solution is found in less than half the time CPLEX takes to prove the optimality of the solution. Therefore, we also provide results on the performance of CPLEX when the running time is limited. We chose the limit on the maximum running times based on the number of trains and number of stations of the problem.

Since the LP-formulation is also based on the time-space network representation, the same conclusion is valid for the IP-formulation; see the column "LP-Greedy Construction" for the results of the simulation-based construction approach using the LP-greedy selection rule. The results show that it takes significantly longer for the algorithm to reach a feasible solution when the maximum delay allowances are larger, irrespective of the tightness of the problem. The gap between the best/optimal solution and the heuristic result varies from 4.76% to 35.71%. Whenever the algorithm reaches an infeasible solution and has to go back in time, the heuristic result is significantly worse when compared with the IP-formulation. For smaller problems, the time for the heuristic is almost identical with the IP-formulation. For larger problems, the time difference between the heuristic and IP-formulation is significant.

We test the IP-based heuristic by varying the time per iteration (σ) from 1 second to 20 seconds. The IP-based heuristic is superior to the simulation-based LP-greedy construction in terms of both the quality of the solution and the time. Out of 15 problems, the IP-based heuristic finds the best solution for 14 of them (breaking the ties in favor of the shorter time). For problem 30-20-r, the LP-based simulation approach finds a better quality solution with a gap of 5.00% from the best solution obtained by the IP formulation. The results for the best solution found for an instance by any of the heuristics are printed in bold. It should be noted that the performance of the IP-based heuristic is based on selecting the length of the short term (q_s) and the time per iteration. Our preliminary results show that setting the ratio of q_s/q between 1/4 and 1/6 is a good choice. Lower or higher ratios decrease the possibility for improvement in a single iteration. The results in Table 2 use a ratio of 1/5; we only report the results based on different choices of time per iteration. When the different times per iteration are compared, longer iterations are expectedly better, since they significantly outweigh the shorter iterations with better quality solutions for larger problems. The results do not matter for smaller problems, as we do not need to go further than a single iteration in most cases. However, the individual results for larger problems are satisfactory with longer iteration times of the IP-based heuristic.

5.2 Synthetic Test Problems

We have generated three instances in this set. The number of stations and sidings, and the physical infrastructure (spacing between stations and/or sidings) of the rail network, as well as train speeds mimics a

typical large North American railroad corridor, which has 40 stations and sidings. The typical load of this corridor is 35 to 40 trains in a repetitive fashion, which is above the average traffic of a typical single track congested railroad similar to, for instance, an iron ore corridor found in Brazil. We first set up a base instance with 40 trains, which is representative of the frequency of trains between the pairs of end stations. Then, we generate a larger instance with 50 trains by increasing the frequency of trains on expectedly larger demand origin-destination pairs; we generate a smaller instance by eliminating the shortest 10 trains from the 40-train schedule. The instances still consider the same planning horizon of two-days, which corresponds to the usual strategic planning horizon and is also much above the usual operational planning horizon.

Although we cannot use real data for the speed of trains and their exact schedules, the targeted departure and arrival times help us roughly assign maximum delay allowances at 50% of their ideal total travel times calculated at an average speed expectation. From a practical perspective, the length of 15-minutes is considered accurate enough by the railroad practitioners to represent the resolution of possible conflicts, especially at strategic and tactical planning levels. In order to understand the sensitivity of the solution approaches to the choice of time discretization, we solve each instance with 10, 15 and 20 minutes of discretization.

The choice of the time discretization is an important issue that affects the capacity of the model in representing real-life situations. It also depends on the purpose of the model. At strategic and tactical planning levels, discretization in 10 or 15 minute periods may be sufficient for the level of accuracy and certainty of all input data, as well for the type of result usually aimed. At the operational level, a shorter time step would be required; in most cases, a 5 minute period may suffice given the low speeds of freight trains, as well as the length of track segments. In general, an interval of less than 3 minutes would never be required, even in highly congested freight railroads, or in the presence of passenger trains. If the operational planning should consider a longer planning period (e.g. 24 hours or more), the size of the network may increase and affect the performance of the proposed approaches to reach a good solution in short running times. One alternative to overcome this difficulty is to consider different time intervals for the short and long terms, similarly to the approach adopted in the IP heuristic with respect to the maximum allowable delays. In other words, given the uncertainty of the schedule in the long range (due to future disruptions and other unpredictable events that may cause trains to get delayed) it may not be necessary to consider a short time interval for the long range, since a longer interval (e.g. 10 minutes) may be enough to ensure feasibility in the long term and good quality solutions in short term. This can be accomplished in an easy and straightforward manner, posing no difficulties or any changes in the modeling, but ensuring the managability of the IP formulation with the IP-based heuristic and the LP relaxations, as evidenced by our experiments with the synthetic test problems.

In Table 3, we only report results for the IP-based heuristic and pure CPLEX runs. The LP-greedy construction algorithm performs poorly when compared to the IP-based heuristic. Particularly, since the size of this set of problems are larger than those of Section 5.1, the solution times increase unfavorably. A problem instance from this set is described by a triple "*t-s/p*" where *t* represents the number of trains, *s* represent the number of stations, and *p* is the length of the time interval for discretization. We note that the number of discrete time instants is equal for those with the same value of *p*. The second and third columns of Table 3 report the objective function value and time to solve the problem to optimality if an optimal solution is known for the problem. The next two columns report the same for the IP-based heuristic; the time per iteration is set to 20 seconds and the ratio of q_s/q is set to 1/5 for all instances. The results clearly show that finding an optimal solution in a reasonable time is almost impossible when the problem is large, the horizon is long and the discretization interval is too small. Yet, the IP-based heuristic finds very good quality solutions in a reasonable amount of time. Considering that this type of problem may be solved daily (or at the beginning of each

dispatcher's shift), the time for 50-50/10 might be too high. Yet, the quality of the solution is significantly better than that of CPLEX. Nonetheless, 50-50/15 solves the same problem with less accuracy; and the solution quality is still superior to CPLEX alone while the time is in reasonable limits. The last four columns report the performance of the IP-based heuristic with respect to the solutions obtained by CPLEX in a limited time (1 hour, 2 hours and 3 hours) and the best solution we know for the instance. The negative gap figures show that the IP-based heuristic solution is better than the feasible solution obtained by CPLEX with the time limit.

It is quite apparent from the results that the IP-based heuristic performs significantly better than what CPLEX can do even under time limit restrictions (hence sacrificing from the optimality) for the test problems that represent real-life instances. The results also show that the number of trains (hence the frequency of arrivals and departures) with the same planning horizon and the same physical network has an expected effect on the solution time (either by CPLEX alone or by the heuristic) on the average. Similarly, when the results for the same instance are compared for different time discretization level, the less accurate (i.e., larger time interval) the problem setting is, the easier problem to solve.

5.3 A Real-Life Problem

The proposed algorithms are also applied to a real-world problem of a major railroad in Brazil. We get all the necessary data from Leal et al. (2004). The problem comprises a single-line track that links two major stations with 23 intermediate stations or sidings where trains can be pulled over; 25 scheduled trains are to be dispatched over a 24-hour period; trains run in both directions. The train routes, the travel times in each section, and the scheduled departure times are all known in advance. Our objective is to minimize the total unweighted delay of all the trains. For the trains traveling between the two endpoints, we set the maximum allowed delay equal to four hours; for the remaining trains, we set the maximum allowed delay as proportional to this value. We choose to discretize times in periods equal to five minutes in order to produce a precise timetable for the trains that could be used at the operational level.

Table 4 presents the results obtained by the IP formulation and the heuristics in this study for the problem. It reveals that all the results related to the proposed heuristics significantly improve those obtained manually by the dispatchers, as well as the previous results of Leal et al. (2004).

6. CONCLUSIONS

We present in this paper a new network based IP formulation and an application of two heuristic approaches for the freight train dispatching problem. We investigate the previous approaches and identify those issues that have not been considered earlier in the literature. We develop a novel IP formulation based on the space-time network representation of the problem. The heuristic approaches benefit from the IP-formulation and the available solver technology, and they build new ideas upon those that have been studied before.

The novelty of the IP formulation is due to its ability to handle several practical constraints without complicating the formulation further. The structure of the formulation does not change with the addition of these new constraints since the variables of the formulation are based on the space-time network representation rather than the conventional definition of the decision variables in the previous formulations that are based on selecting the train to be pulled over in case of a conflict. The complex infrastructures of railroad networks, such as double

lines and intersecting railroads, are easily handled with this type of network representation, which avoids the complications in the formulation.

These approaches should be evaluated from the perspective of a master planning tool that provides a guideline to dispatchers in their day-to-day operations. Nonetheless, we emphasize that the planning scope of the problem is indeed very large, and with appropriate decision support mechanisms, the same modeling approach can be used for different planning purposes.

Computational results reveal that each solution methods have its pros and cons. Our experiments indicate that with enough technology and advanced computational infrastructure that can support the most recent solver technology, the IP formulation may provide satisfactory results as a master planning tool for moderate size problems. On the other hand, the IP-based heuristic should be considered as a trustworthy master planning tool since the experiments confirm that it provides high quality solutions with a significantly less amount of computational effort particularly when the problem size is large and the accuracy of the representation (based on the time discretization) is better. The simulation-based LP-greedy algorithm is inspired by the dispatcher's decision-making process. Although, it is clear that the other methods outweigh the LP-greedy algorithm from a master planning tool perspective, it raises the question "Why should the dispatcher not use the LP-relaxation solutions at the time of a conflict resolution?". It is clear that solving the LP-relaxations of the problem is accomplished in a matter of seconds even for the largest problems, not to mention the possibility of detecting future infeasibilities.

We hope this study motivates the railroad industry to consider incorporating these optimization-based approaches into their efforts in order to tackle the train dispatching problem. We believe that our methods verify the feasibility of an effective solution using analytical and systems-based approach.

ACKNOWLEDGEMENTS

This paper was written during Dr. Cunha's sabbatical at the University of Florida's ISE department as a visiting faculty member. Dr. Cunha acknowledges Brazil's CAPES (Coordenadoria de Aperfeiçoamento de Pessoal de Nível Superior) for the grant that allowed his stay at the University of Florida between August 2003 and April 2004, during which this research was undertaken.

REFERENCES

Adenso-Diaz B, Gonzalez MO, Gonzalez-Torre P. On-line timetable re-scheduling in regional train services. *Transportation Research Part B* 1999; 33; 387-398.

Ahuja RK, Magnanti TL, Orlin JB. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall: New Jersey; 1993.

Assad AA. Analysis of rail classification policies. *INFOR* 1983; 21; 293-314.

Brännlund U, Lindberg PO, Nõu A, Nilsson J-E. Railway timetabling using lagrangian relaxation. *Transportation Science* 1998; 32(4); 358-369.

Cai X, Goh CJ. A fast heuristic for the train scheduling problem. *Computers and Operations Research* 1994; 21; 499-510.

Cai X, Goh CJ, Mees AI. Greedy heuristics for rapid scheduling of trains on a single track. *IIE Transactions* 1998; 30; 481-493.

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

Carey M. Extending a train pathing model from one-way to two-way track. *Transportation Research Part B* 1994b; 28; 395-400.

Carey M, Lockwood D. A model, algorithms and strategy for train pathing. *Journal of the Operational Research Society* 1995; 46; 988-1005.

Caprara A, Fischetti M, Toth P. Modeling and solving the train timetabling problem. *Operations Research* 2002; 50; 851-861.

Churchod A, Emery D. Computer-aided planning for major railway stations. In: Murthy TKS, Young FE, Lehmann S, Smith WR (Eds), Computers in railway installations, track and signaling. *Computational Mechanics Publications*: Berlin; 1987. p. 3-19.

Cordeau JF, Toth P, Vigo D. A survey of optimization models for train routing and scheduling. *Transportation Science* 1998; 32; 988-1005.

Higgins A, Kozan E, Ferreira L. Optimal scheduling of trains on a single line track. *Transportation Research Part B* 1996; 30; 147-161.

Higgins A, Kozan E, Ferreira L. Heuristic techniques for single line train scheduling. *Journal of Heuristics* 1997; 3; 43-62.

Jovanovic D, Harker PT. Tactical scheduling of train operations: the SCAN I system. *Transportation Science* 1991; 25; 46-64.

Kraay D, Harker P. Real-time scheduling of freight networks. *Transportation Research Part B* 1994; 29(3); 213-229.

Kroon LG, Peeters LW. A variable trip time model for cyclic railway timetabling. *Transportation Science* 2003; 37; 198-212.

Leal JE, Soares AC, Nunes LCN. A heuristic approach for the train scheduling problem on single railway tracks. In: Annals *of the XVIII ANPET Congress*. Brazil: 2004. p. 945-954 (in Portuguese).

Peterson ER, Taylor AJ, Martland CD. An introduction to computer aided train dispatch. *Journal of Advanced Transportation* 1986; 20; 63-72.

Rivier RE, Tzieropoulos P. Interactive graphic models for railway operational planning. In: Florian M. (Ed*), The Practice of Transportation Planning*. Elsevier Science Publishers: Amsterdam; 1984. p. 245-259.

Rivier RE, Tzieropoulos P. Computer-aided planning of railway networks, lines and stations. In: Murthy TKS, Lawrence LS, Rivier RE (Eds), Computers in railway management. *Computational Mechanics Publications*: Berlin; 1987. p. 3-16.

Schrijver A. Minimum circulation of railway stock. *CWI Quarterly* 1993; 6; 205-217.

Şahin I. Railway traffic control and train scheduling based on inter-train conflict management. *Transportation Research Part B* 1999; 33; 511-534.

Sauder RL, Westerman W. Computer aided train dispatching: decision support through optimization. *Interfaces* 1993; 13; 24-37.

Smith ME. Keeping trains on schedule: on-line planning systems for the advanced railroad electronics system (AERS). *Journal of Transportation Research* Forum 1990; 31; 17-24.

Törnquist J, Persson JA. N-tracked railway traffic re-scheduling during disturbances. *Transportation Research Part B* 2007; 41; 342-362.

Zhou X, Zhong M. Single-track train timetabling with guaranteed optimality: Branch-and-bound algorithms with enhanced lower bounds. *Transportation Research Part B* 2007; 41; 320-341.

Table 1. Results for the IP formulation and simulation-based approach with the LP-based selection rule.

	IP Formulation			IP Formulation with Limited Time			LP-Greedy Construction			
Problem	Solution	CPU Time	Gap (>)	Solution	Time Limit	Gap	Solution	CPU Time	Infeasibilities	Gap
17-9-r	24	3.6					27	9.0	0	12.50%
17-9-50	24	2.5					30	5.6	0	25.00%
17-9-100	24	5.3					27	9.6	0	12.50%
24-10-r	31	26.9					36	29.4	0	16.13%
24-10-50	34	10.1					43	25.2	0	26.47%
24-10-100	31	16.2					34	31.3	0	9.68%
24-16-r	42	197.6		42	180	0.00%	57	171.9	14	35.71%
24-16-50	42	990.9		42	180	0.00%	44	83.3	0	4.76%
24-16-100	41	757.0		42	180	1.00%	44	163.6	0	7.32%
24-20-r	43	1013.2		44	360	0.00%	48	103.7	1	11.63%
24-20-50	43	717.8		44	360	0.00%	48	158.0	0	11.63%
24-20-100	43	4153.5		48	360	11.00%	48	362.1	0	11.63%
30-20-r	82	> 7200	25.00%	n/a	720	n/a	84	499.7	3	5.00%
30-20-50	73	> 7200	25.00%	81	720	37.00%	78	677.1	0	6.85%
30-20-100	73	> 7200	25.00%	82	720	38.00%	82	1849.0	0	12.33%

Table 2. Results for the IP-based heuristic.

	IP Based Heuristic (20)			IP Based Heuristic (10)			IP Based Heuristic (5)			IP Based Heuristic (2)			IP Based Heuristic (1)		
Problem	Solution	CPU Time	Gap	Solution	CPU Time	Gap	Solution	CPU Time	Gap	Solution	CPU Time	Gap	Solution	CPU Time	Gap
17-9-r	24	1.9	0.00%	24	2.6	0.00%	24	2.1	0.00%	24	1.9	0.00%	24	4.5	0.00%
17-9-50	24	1.5	0.00%	24	1.7	0.00%	24	1.5	0.00%	24	1.4	0.00%	24	3.5	0.00%
17-9-100	24	4.6	0.00%	24	4.3	0.00%	24	4.3	0.00%	24	7.4	0.00%	24	5.4	0.00%
24-10-r	34	15.5	9.68%	34	18.3	9.68%	34	13.2	9.68%	34	24.3	9.68%	43	8.7	38.71%
24-10-50	34	9.1	0.00%	34	9.4	0.00%	34	11.9	0.00%	34	12.3	0.00%	34	13.2	0.00%
24-10-100	34	35.2	9.68%	34	24.8	9.68%	34	23.0	9.68%	34	13.9	9.68%	34	10.3	9.68%
24-16-r	42	131.5	0.00%	42	49.9	0.00%	42	36.5	0.00%	46	15.7	9.52%	46	11.6	9.52%
24-16-50	43	112.7	2.38%	42	75.1	0.00%	44	29.2	4.76%	44	17.2	4.76%	44	13.0	4.76%
24-16-100	41	136.0	0.00%	42	105.4	2.44%	42	53.1	2.44%	51	20.2	24.39%	51	15.9	24.39%
24-20-r	43	155.5	0.00%	43	54.8	0.00%	45	26.7	4.65%	45	17.7	4.65%	45	13.5	4.65%
24-20-50	43	144.7	0.00%	43	56.5	0.00%	44	41.2	2.33%	44	28.6	2.33%	44	24.4	2.33%
24-20-100	43	152.8	0.00%	45	168.1	4.65%	59	70.2	37.21%	59	58.9	37.21%	59	55.4	37.21%
30-20-r	92	220.0	15.00%	93	98.6	16.25%	96	75.6	20.00%	96	61.7	20.00%	98	62.8	22.50%
30-20-50	76	183.3	4.11%	76	82.2	4.11%	92	78.3	26.03%	117	45.4	60.27%	117	34.5	60.27%
30-20-100	75	128.9	2.74%	80	86.3	9.59%	85	30.9	16.44%	85	9.9	16.44%	85	8.1	16.44%
Average	-	-	2.91%	-	-	3.76%	-	-	8.88%	-	-	13.26%	-	-	15.36%

Table 3. Results of the synthetic problem set with the IP-based heuristic.

	Optimal		IP-Based Heuristic		Heuristic Gap (%)			
Problem	Solution	CPU Time	Solution	CPU Time	1 hour	2 hours	3 hours	Best
30-50/10	-	-	142	105.6	-5.96%	0.00%	0.00%	0.00%
30-50/15	98	1023.9	100	58	2.04%	2.04%	2.04%	2.04%
30-50/20	62	5.2	62	6.1	0.00%	0.00%	0.00%	0.00%
40-50/10	-	-	205	180.6	4.06%	4.06%	4.06%	4.06%
40-50/15	119	149.5	124	14.8	4.20%	4.20%	4.20%	4.20%
40-50/20	80	25.5	82	9.1	2.50%	2.50%	2.50%	2.50%
50-50/10	-	-	285	1205.4	-20.39%	-20.39%	-17.15%	-17.15%
50-50/15	-	-	175	196.8	-1.13%	-0.57%	-0.57%	-0.57%
50-50/20	110	548.2	110	38.6	0.00%	0.00%	0.00%	0.00%
Average					-1.63%	-0.91%	-0.55%	-0.55%

Table 4. Results of the real-world problem.

	Total Travel Time (minutes)	Total Delay (minutes)	CPU Time
Dispatchers Solution	14,119	6,311	-
Optimal IP Solution	8,240	432	820
IP-Based Heuristic	8,245	437	173
LP-Greedy Construction	8,573	765	136
Heuristic 1, Leal et al (2004)	8,766	958	-
Heuristic 2, Leal et al (2004)	8,755	947	-