An approach for the railway multi-territory dispatching problem

Conrado Borraz-Sánchez, Diego Klabjan
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208
{c-borraz-sanchez, d-klabjan}@northwestern.edu

Alper Uygur
Manager, Revenue Management Science at Carnival Cruise Line
Alper.Uygur@gmail.com

Abstract: In this research, we address the railway multi-territory dispatch planning (RMTDP) problem. The RMTDP problem is concerned with achieving optimal movement of trains along consecutive dispatch territories, and it is one of the major challenges that decision makers face on a daily basis. It ideally takes into account the correct placement of maintenance windows, remaining capacity of terminals and availability of train crews among other critical aspects such as locomotive balance, fueling locations and inspections. Although these train movement plans are made at the corridor level, which is comprised of several dispatch territories, when it comes to execution, the meet-pass decisions are made at the individual dispatch territories. This notion causes disruptions and misalignment at the boundaries of dispatch territories. The approach in this paper aims at finding a holistic conflict-free master plan by optimally matching train line-ups at territory boundaries and smoothly routing trains through bottlenecks.

We propose an efficient solution approach that iteratively constructs a master scheduling plan while minimizing the amount of train delays within a given planning horizon. This is accomplished by designing a time-space network model to identify feasible schedules and developing a mathematical programming-based heuristic to solve the underlying model. A thorough computational study shows the effectiveness of our heuristic approach, as we report reasonable average run times of 3.0 and 6.5 minutes to solve instances of moderate to large size problems, respectively.

The results obtained from the algorithm using test snapshots from a Class I Railroad company have been shown to assistant chief dispatchers and have received encouraging feedback for applicability.

Keywords: Railway dispatching, Integer Programming, Heuristics, Meet&Pass, Multitrack Territories

1. Introduction

The railway industry has been essential for the global economic growth. During the industrial revolution held in Europe in the 19th century, railways played a key role in supporting technological progress in transportation. Since then, the percentage of freight hauled by railroads have increased continuously and at this point in time there is very limited substitute for rail freight transportation. Due to the exponential increase of railway networks observed during the last decades, the railway industry has paid special attention to the quality of the rail traffic planning process. Improving this process is clearly necessary to meet and exceed customers' expectations. As a consequence, railway operators and dispatchers always strive for providing efficient plans, which include defining specific train line-ups and timing at control points, terminals and platforms. Hence, a robust master plan has to be created while optimizing train delays which occur due to meet-pass events and work events scheduled at the designated locations.

In this work we address the railway multi-territory dispatch planning (RMTDP) problem. The RMTDP problem refers to the train routing and scheduling problem over a set of dispatch territories adjacent to each other and complex terminals in between where a set of specific business rules are imposed. Our goal is to find a master schedule for a set of trains that traverse in both directions over a corridor which may have both single track territories and multi-track territories. For ease of exposition, terminals are also considered as dispatch territories in this work. There are, however, a few key distinctions between a terminal and a typical dispatching territory. For example, terminals are shorter in length yet much more complex to operate than a typical dispatching territory. For this exercise, we will only be considering the through tracks where trains actually move on to pass from one territory to another, have a crew change, get fuel or get inspection done. The yard tracks where switching of cars happen are not being considered as usable capacity in this study. Another difference between terminals and line segments is that although the through tracks are under the governance of the terminal dispatcher, he/she consults with the terminal trainmaster/yardmaster to actually line-up signals or to give authority to trains to move. Fig. 1 shows a terminal in between two dispatch territories. A typical network topology tackled in this work is shown in Fig. 2, where the entire corridor is composed of 3 dispatch territories (T1, T3, and T5) and two terminals (T2 and T4).

By defining a track section as a collection of tracks (i.e., main tracks, sidings, switches, crossovers) between two adjacent control points, in its broadest sense, the RMTDP problem concerns the optimization of route allocations and holistic train scheduling at intermediate control points. Route allocation refers to the assignment of trains on track sections and the specification of arrival and

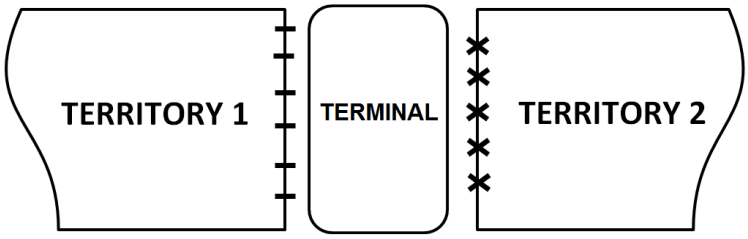

Figure 1 – Typical stitching area composed of two dispatch territories and a terminal in between.

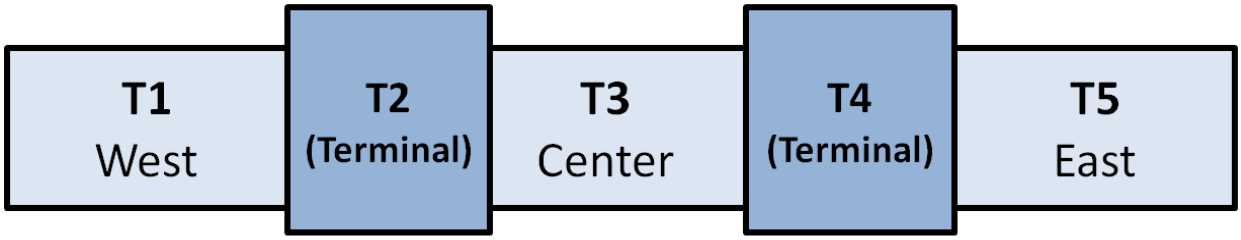

Figure 2 – A network instance of the multi-territory dispatch planning problem.

departure times at the end points of these sections. Holistic train scheduling requires finding the optimal set of train arrival and departure times at specific control points that make trains traverse smoothly from one dispatching-territory to the next.

The main challenge is posed by the fact that, initially, the dispatchers governing adjacent dispatch territories may very likely not have compliant plans. As a consequence, discrepancies in train line-ups would arise. For example, while constructing a plan for his/her shift, the dispatcher of Territory T3 may assume different times of arrival for what the dispatcher of Territory T1 actually plans for. Hence, the key idea of this project is to create a scheduling plan that benefits the entire corridor. The majority of the difficulties arise when (1) assigning tracks to trains inside the terminals due to work events scheduled for each train under the observance of a very limited through track capacity; and (2) enforcing adjacent dispatchers to bring trains at specific times into the terminals while satisfying their own set of business rules. In this work, terminals or any physical location between two adjacent territories are called stitching points. The stitching master plan is strongly constrained by a set of business rules imposed on the terminals and the internal business rules imposed by each independent dispatching territory.

In this work we propose an efficient solution approach to resolve train conflicts and infeasibility situations during the creation of a holistic master plan, aimed at minimizing the amount of train delays and other objectives within a given planning horizon. This is accomplished by developing a mathematical programming-based heuristic that iteratively provides matching train line-ups among

the independent territories' boundaries while enforcing a set of specific business rules imposed at the terminals. We assume the presence of an oracle that solves dispatching within a single territory (but not within a terminal). Such an oracle is separated from the rest due to complicated business rules and thus it provides a robust solution. Typically a railway first develops this oracle and later integrates it into a system for cross-territory dispatching, e.g. by using our approach.

Our mathematical-programming based approach consists of three phases. In phase one, we implement train line-ups matching procedures to obtain an initial state for the entire corridor. In phase two, we create both a time-space network model to find feasible train movements along the paths in the stitching network sections, and an integer programming (IP) model to capture the set of business rules imposed at the terminals. Our models are embedded in an iterative scheme to efficiently incorporate individual routing plans from independent dispatching-territories provided by the oracle, into the global master plan. The algorithms routes trains sequentially but allows to unwind decisions. Finally, in phase three, we apply a bisection method to minimize the overall delay amount of the entire railway network. This phase has finer time granularity.

The main contributions of this research are: (1) the development of a novel approach to solve the RMTDP problem, which makes use of heuristic techniques combined with mathematical programming; 2) the design of a new integer programming-based model for train routing inside terminals that is capable of producing an operationally feasible schedule (the model allows trains to be scheduled in time while enforcing a set of business rules imposed in practice by dispatchers), 3) the efficiency, scalability and robustness of the heuristic method for solving the overall problem, 4) and the study of a set of large-scale scenarios (the data comes from a Class–1 US railway, including all business rules and the oracle).

The paper is organized as follows. Section 2 provides the problem statement, in which the stitching network areas are described in detail. Section 3 presents the proposed heuristic approach by describing its key modules, including the time-space network and IP models. Section 4 reports the results of several computational analyses conducted on a set of large-scale case studies to show the capability of the proposed heuristic. This section also includes a thorough comparative analysis between our train line-up solutions against baseline projections provided by the Class–1 US railway. Finally, concluding remarks are given in Section 5. The literature review is presented next.

1.1. Literature review

The literature on railway scheduling optimization is abundant. However, the research in this area have been done on train planning and timetabling on a single dispatch territory, primarily focusing

on scheduling trains on single or multiple tracks within.

Research on a single territory can be traced back to early 1970s. Szpigel [15], for example, developed a linear programming model to determine the best meet and overtake locations given that the departure times and maximum velocities of the trains are known. Szpigel's work aimed at resolving the conflicts for a set of small problems while minimizing the sum of travel times by means of a branch and bound method. Petersen et al. [14] considered a similar approach by proposing a dispatch algorithm to calculate the segment transit times and to determine the meet-pass locations. Mees [13] models a single track territory as a network structure where each segment is an arc (a siding is considered as an extra arc), separated by nodes (considered as track intersections or stations). It is created as a time-space network with a fixed schedule time span, and headway separations are obtained by allowing only one train per arc at a time segment.

Research on railway traffic scheduling can be classified as tactical scheduling, operational scheduling and re-scheduling. While tactical and operational scheduling –with different time perspectives– are both treated as a timetabling problem where a schedule does not yet exist (i.e., plans are to be created not long before trains departure with the intention to be executed in real-time), the re-scheduling is considered a dispatching problem where a schedule already exists and is to be modified. T*ó*rnquist in [16] provides a good classification on train scheduling at tactical and operational levels. Jovanović and Harker [11] address the tactical train scheduling problem, which aims at determining train arrival and departure times at specific points (e.g. yards, terminals, junctions) along a track line. The results are then provided as timetables for marketing purposes.

Carey [4] extended the model for train pathing suggested by Carey and Lockwood [7] with choice of lines, station platforms and routes. Basically, the train pathing model focuses on single track rail lines with trains in both directions, and meets and passes taking place at sidings located at intervals along the single track line. In both papers, it is assumed that each track is normally dedicated to trains in one direction. In [4], Carey deals with the problem by applying two-way track headway constraints. Carey in [5] continues his previous work on the train pathing problem and proposes an IP model with multiple lines and platforms that handles general rail network planning policies. Basically, Carey tackles the model by adopting strategies analogous to those adopted by "expert" train pathers using traditional manual graphical methods. The simple strategy is to "path" the trains one at a time until all trains are routed once, and if necessary iteratively repath trains until an acceptable solution is found.

Alfonso and Bispo [1] address the meet and pass train scheduling problem over a single track line. They consider the fixed block signaling system, which allows trains to follow each other

with minimum safety headway, and propose a two-phase algorithm to solve it. First, a conflict detection loop is executed, and then, based on heuristic and search-based solution procedures, a conflict resolution procedure is applied to solve the detected conflicts. The authors test the algorithm over several instances obtaining encouraging results. Similarly, Boccia et al. [3] propose two heuristic approaches to solve the dispatching problem on a single multi-track territory by a mixed integer linear programming formulation. Lin et al. [12] develop a decision support system for train dispatching based on exact and heuristic algorithms that can be applied solely to individual dispatch territories.

Higgins et al. [10] solve the train scheduling problem on single track rail corridors. They focus on developing a decision support tool for train dispatchers to schedule trains in real-time, and propose a nonlinear IP model to help dispatchers perform train dispatching functions for short to medium term train planning. Their model is embedded in a branch and bound procedure, and tested on a single instance composed of a rail corridor containing 14 sidings and 31 trains. They also conducted a second experiment that consisted of scheduling additional trains at the already congested periods of the existing schedule. Run times of 10 minutes were observed.

There is also a research on scheduling of single stations. Carey et al. [6], for example, focus on a single large train station with multiple platforms and multiple conflicting in- and out-lines, yet they do not deal with matching times at intermediate points as required by the RMTDP problem. The reader is referred to the papers of Assad [2], Cordeau et al. [8] and Tórnquist [16], and more recently Corman [9], for complete surveys in this area.

Our research is different from these aforementioned papers since they are not concerned with multiple dispatching-territories or complex stations (terminals) or assume the latter have unlimited capacity. In particular, we differ from each of the above mainly by the RMTDP problem requirement that the master plan needs to take into account work events at certain intermediate terminals with finite capacity in between multiple dispatch territories.

2. The railway multi-territory dispatching problem

The RMTDP problem refers to the problem of scheduling a set of trains traversing multiple dispatch territories adjacent to each other while obeying a set of business rules. A dispatch territory can be formed by either single track segments where only a unidirectional movement can happen in between sidings in some period of time (more than 1 train can be trailing each other), or a combination of single-track segments and multi-track segments where multiple trains are allowed to move in different directions by utilizing parallel tracks. Either way, dispatch territories have tracks

called sidings along their main tracks where meet-pass events or specific train delays may occur. For the case of multiple tracks, it also includes cross-overs (i.e., specific locations where trains can move from one track to another) along the track lines.

The RMTDP problem aims at avoiding deadlocks (opposing trains coming to full stop and causing one of them backing up) as well as any other potential infeasible routing outcomes. The general goals are (1) to obtain a proper routing of trains within each dispatch territory; (2) to create a detailed schedule inside the terminals with limited capacity; and (3) to match holistically the train line-ups (i.e., arrival and departure times) at intermediate control points between each consecutive dispatch territories. Note that in (1), we obtain independent train schedules for a single dispatching-territory by means of an outside oracle. The input data fed into the oracle consist of the network configuration of a territory, train priorities, train departure times, terminal-want times (i.e., suggested train arrival times at the terminals), specific sets of track segments available for each train, and costs of using tracks and entering sidings.

Due to the meet-pass events, train delays may occur throughout the entire network. Consequently, these delays may be propagated as secondary delays to other trains because of their holistic interaction, thus affecting the entire network. Hence, the overall objective is to provide a complete and detailed train schedule for the entire network with the least amount of delay while adhering to train priorities.

Typically, sidings are placed every 10 to 15 mile interval along a track line. Most sidings can accommodate only one train to conduct meet-pass events. Whenever the plan requires to have two trains meet with each other, the trains are run as far as they can and then one takes a siding (usually the one with low priority.) In general, train priority plays a key role in routing them through the network. Business rules indicate the priority of the trains.

When creating a master scheduling plan, our algorithm's robustness covers a wide set of business rules imposed by the dispatchers. For example, minimum dwell times for trains to perform their work events inside the terminals, a minimum separation rule (i.e., trains are obligated to meet certain minimum safety headway when following each other), and trains' priorities are to be considered when allocating delays along a track line.

Note that in practice, dispatchers create a plan for solely those trains that are expected to be in their territories during a given planning horizon. The dispatchers collect all the information with regards to their trains, including the expected arrival time into their territories by means of a real-time scheduling (RTS) system. The RTS system provides individual train-based projections based on history, without considering the current state of the railroad. Therefore, when downstream

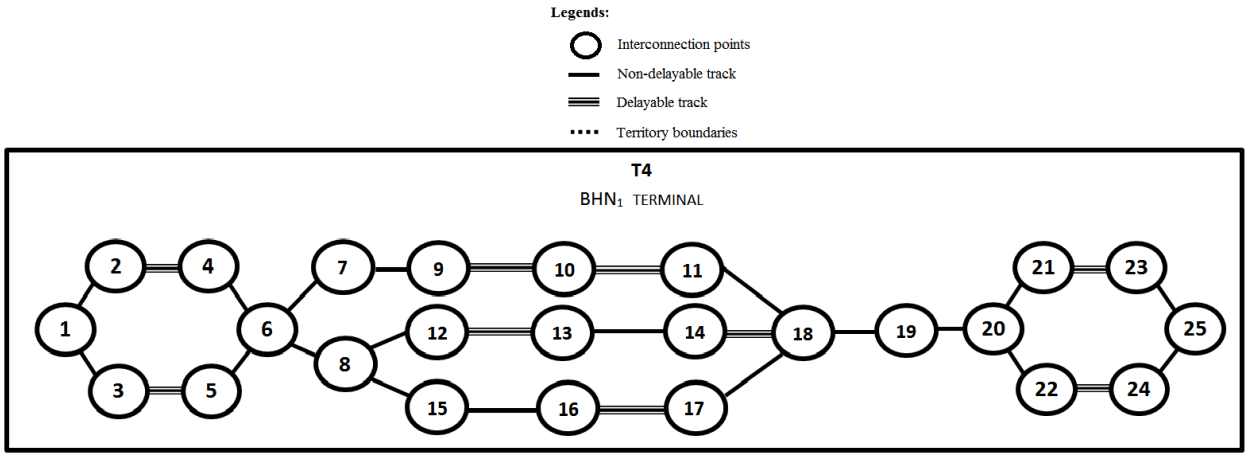

Figure 3 – Network configuration for BHN_1 (a stitching network area)

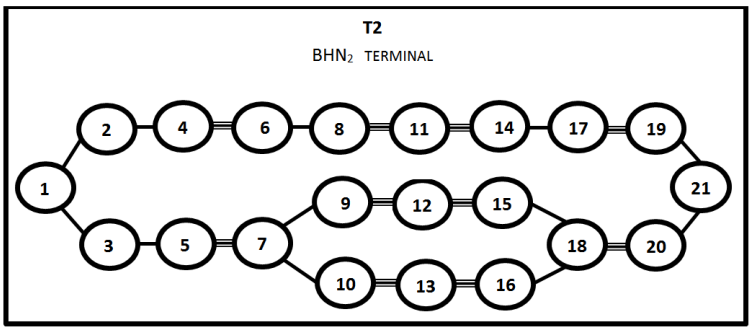

Figure 4 – Network configuration for BHN_2 (a stitching network area)

dispatchers check for estimated arrival times, that information does not come from the dispatchers, and thus the actual line-ups may be different than what the RTS system suggests.

2.1. Stitching network areas

In presence of 3 dispatch territories, there are two stitching network areas that are intrinsically dependent on each other, namely BHN_1 (see Fig. 3) and BHN_2 (see Fig. 4.) Each one is defined by a specific type of a network configuration that includes a complex terminal with limited capacity.

These stitching areas are crucial for the routing and accurate timetabling of trains through the overall railway network. Note that due to the safety regulations, we impose a train separation rule at the arc level within the stitching areas for trains traversing in any direction. Our goal is to find an optimal set of arrival and departure matching times at these stitching areas that make trains traverse smoothly from one dispatching territory to the next.

3. Solution procedure

The proposed stitching algorithm is based on a methodology that consists of three phases.

(A) Pre-processing: This phase resolves immediate infeasible situations (e.g., time discrepancies among the independent scheduling plans) and provides initial train line-ups. It neglects all requirements inside the stitching areas.

(B) Stitching: This phase focuses on creating train paths through the terminals (T_2 and T_4, see Fig. 2). It starts with the solution from preprocessing. This is accomplished by creating (a) an individual time-space network for each train and (b) solving the underlying IP model for each train. The key idea of this phase is to execute steps (a) and (b) iteratively for each train until a stopping criterion is met.

(C) Post-processing: This phase improves the final feasible solution by applying a bisection approach to reduce possible slack. It is required due to coarse time discretization in the stitching phase.

Taking into account the entire railway network depicted in Fig. 2, the algorithm basically aims at creating train paths for a set of trains traversing the stitching network areas T_2 and T_4. To route trains through T_1, T_3, and T_5, we make use of the outside oracle, which we call the movement planner (MP) to get independent scheduling plans for each territory. MP takes, among a large set of input data, the network configuration of a territory, entry times, train priorities, costs associated with the use of track segments and access to siding stations, and terminal want times (TWT) as input, and it provides arrival times at the terminals as output.

Note that MP is called iteratively in all of the main phases of the algorithm to solve T_1, T_3, and T_5 as many times as required. In any iteration we obtain updated train line-ups at the boundaries of T_2 and T_4 by algorithmically updating the input data to MP in each territory. The challenge is to match the entry and exit times of the independent train line-ups provided by MP while meeting all the business rules imposed in T_2 and T_4.

The following notation is employed to describe the main procedures of the stitching algorithm.

Sets:

$t \in \mathbb{T}$: Set of territories, $\mathbb{T} = \{1, .., 5\}$

$i \in \mathbb{I}$: Set of trains

$e \in \mathbb{E} \subseteq \mathbb{I}$: Set of eastbound trains

$w \in \mathbb{W} \subseteq \mathbb{I}$: Set of westbound trains

Parameters:

α_i^t : Unimpeded path time of train $i \in \mathbb{I}$ through territory $t \in \mathbb{T}$

Δ_i^t : Required minimum dwell-time for train $i \in \mathbb{I}$ in territory $t \in \mathbb{T}$

L^t : Train line-up in territory $t \in \mathbb{T}$

FT_i^t : Fitting time of train $i \in \mathbb{I}$ in territory $t \in \mathbb{T}$. This parameter (given in minutes) is the total time allocated for a train to perform its scheduled work events inside a terminal in addition to potential delays required due to feasibility reasons, e.g., meet-pass events. This value is calculated by subtracting the train's scheduled arrival time into the terminal from its estimated departure time from the terminal.

p_i^t : Entry time to territory $t \in \mathbb{T}$ for train $i \in \mathbb{I}$

q_i^t : Exit time from territory $t \in \mathbb{T}$ for train $i \in \mathbb{I}$

Network and math model definitions:

$\mathcal{G}$: Stitching network for BHN_1

$\mathcal{H}$: Stitching network for BHN_2

$IP_{\mathcal{G}}$: IP model for a single train adopting all the business rules in $\mathcal{G}$

$IP_{\mathcal{H}}$: IP model for a single train adopting all the business rules in $\mathcal{H}$

3.1. Pre-processing phase

The pre-processing phase basically provides feasible line-ups by resolving all the immediate discrepancies provided by the initial line-ups and neglecting the routes inside the terminals. The goal is to obtain a feasible set of matching times in the stitching areas by sequentially solving all the adjacent dispatch territories and updating trains' entry and exit times as required. We start by solving those territories that have only a single adjacent territory (T1 and T5 in our case) and move forward to those located in the middle. Note that we start with the outermost territories for two reasons. First, these territories contain fixed entry times, and second, the trains originating in these territories are arguably the ones causing the major infeasible situations when traversing the entire railway network.

Moreover, by working with the most accurate entry times for the current state of the system, we are allowed to compute approximate entry times to the adjacent territories by using Eq. (1) to calculate the fitting time for each train inside the terminals:

$$FT_i^{t'} = q_i^t + \alpha_i^{t'} + \Delta_i^{t'}, \forall i \in \mathbb{I}, \text{for adjacent territories } t \in T \text{ and } t' \in T. \tag{1}$$

Eq. (1) defines the fitting time value for any train inside terminal t' and it equals to train i's exit time q from the previous territory t plus the train's traversal time α and minimum dwell time Δ

Algorithm 1 $PreProcessing(\mathbb{T})$

1: Call MP to solve T_1 and T_5

2: Compute $FT_e^2, \forall e \in \mathbb{E}$ and $FT_w^4, \forall w \in \mathbb{W}$ as follows:

$$FT_e^2 = q_e^1 + \alpha_e^2 + \Delta_e^2; \qquad FT_w^4 = q_w^1 + \alpha_w^4 + \Delta_w^4$$

3: Update $p_i^3, \forall i \in \mathbb{I}$ needed for MP on T_3

4: Call MP to solve T_3

5: Compute $FT_w^2, \forall w \in \mathbb{W}$ and $FT_e^4, \forall e \in \mathbb{E}$ as follows:

$$FT_w^2 = q_w^3 + \alpha_w^2 + \Delta_w^2; \qquad FT_e^4 = q_e^3 + \alpha_e^4 + \Delta_e^4$$

6: Update $p_w^1, \forall w \in \mathbb{W}; p_e^5, \forall e \in \mathbb{E}$ which are required for MP on T_1 and T_5

7: Call MP to solve T_1 and T_5

8: Call the $ConflictRefiner$ procedure to verify feasibility of the train separation rule at arc level and integrity of FT values

9: IF a conflict is detected,

Make the proper adjustments and go to step 1

ELSE

Pre-processing stage finished.

OUTCOME: Set of matching times in stitching areas

through terminal t'.

Alg. 1 shows the main steps conducted in the pre-processing phase. In summary, at every iteration the algorithm sequentially solves all of the dispatch territories, recalculates FT values through the stitching areas, updates the entry and exit times accordingly, and makes a call to the $ConflictRefiner$ procedure to confirm feasibility of train separation and the integrity of FT values.

For illustration purposes, let us consider the network instance shown in Fig. 2. Alg. 1 starts by calling MP to solve the outermost left and right territories T_1 and T_5, respectively. It then proceeds to compute the FT values through the terminals T_2 and T_3 as given by Eq. (1), updates T_3's new entry times, and calls MP in T_3. As a result, we may observe a significant impact on all of the other territories due to the high discrepancy in the trains' entry and exit times caused by solving T_3. The algorithm keeps track of the direction of the trains to update their times on the adjacent territories accordingly by recalculating the FT values. This allows the algorithm to update the entry times in T_1 and T_5, and to call MP again to solve the territories with the newly updated times. This

Algorithm 2 *Stitching*($\mathbb{T}$)

1: Create $\mathcal{H}$ and $\mathcal{G}$
2: Sort L^2 and L^4 based on trains' priority and then exit times.
3: Let A and A' be the first un-routed trains in L^2 and L^4, respectively.
4: Call *LongestPath* algorithm to obtain time bounds at nodes of $\mathcal{H}$ and $\mathcal{G}$.
5: Construct time-space networks for A and A' based on FT values.
6: Create and solve $IP_{\mathcal{H}}$ and $IP_{\mathcal{G}}$ for A and A'.
7: Individually evaluate models' status, if infeasible:
8: Adjust infeasible train on the corresponding stitching area by calling Alg. 3.
9: Call $ConflictRefiner$ to confirm feasibility.
10: Whenever a conflict occurs, make adjustments and call MP as required.
11: Sort L^2 and L^4 as described in step 2.
12: Repeat steps 4–8 until a stopping criterion is met.

OUTCOME: Global master scheduling plan for the entire railway network.

process repeats accordingly until feasible line-ups are obtained at each territory's boundaries.

To ensure that the final-state of this stage is conflict-free, our algorithm iteratively makes calls to the $ConflictRefiner$ procedure to detect possible conflicts within stitching areas. Note that at this stage, this procedure evaluates potential conflicts solely at the stitching networks' boundaries, in particular regarding the train separation rule. This evaluation is required because, although we algorithmically make strong suggestions to MP as to at what time to bring a train, or when it should depart from the stitching locations, MP might modify the suggested times since it still has to enforce specific business rules to its final scheduling plan, thus creating conflicts.

Whenever a conflict is detected, the $ConflictRefiner$ procedure makes the proper adjustments based on the trains' priority. For example, if two trains are in conflict while violating the train separation rule, the train with the lowest priority is retained at the entry point of the terminal, thus increasing its FT value. This change in the entry times also entails an update of train line-ups at the terminals. The $ConflictRefiner$ procedure resolves conflicts one at a time and performs an analysis of the train line-ups once a conflict is resolved. Alg. 1 repeats all the steps until a feasible set of matching times is found.

3.2. Stitching Phase

Alg. 2 shows the steps conducted in the stitching phase. It starts by creating the stitching network areas $\mathcal{H}$ and $\mathcal{G}$ as described in Section 2.1. (At this point, terminals T_2 and T_4 have a train line-up resulting from the pre-processing phase.) The main idea of the algorithm is to route trains one by one. We route an individual train by solving an IP. We sort the trains based on their priorities. We rearrange trains with the same priority based on their exit time (q_i) from the terminals. Note that this sorting stage is repeated iteratively whenever a change in trains' entry or exit times occurs. The trains are routed one by one by following this order. If a train cannot be routed, we backtrack and increase its FT value which triggers changes in the order of trains.

The algorithm proceeds by selecting unrouted trains A and A', each taken from the top of the two different line-ups. We construct the time-space networks and solve the underlying IP models for trains A and A' as defined later in Sections 3.2.2 and 3.2.3, respectively.

To limit the size of the networks, we apply a $LongestPath$ algorithm to obtain upper bounds on the arrival times at every node of the stitching network. This procedure is applied to each individual train based on the expected exit time from the stitching network. Basically, we traverse the stitching network in a backward fashion and compute the maximum time for a train to arrive at a particular node so as to create feasible routes. By doing so, we avoid the construction of the time-space network over the entire planning horizon, thus minimizing the size of the IP model.

The algorithm then proceeds with an individual evaluation of the models' outcome. If the model is feasible (i.e., a feasible route has been found), the algorithm moves forward to the next unrouted train in the line-up and the process repeats. If the model is infeasible (i.e., no feasible route was found for A or A'), the algorithm calls Alg. 3 to make the proper adjustments to the infeasible trains.

Alg. 3 is an adjustment procedure that takes as an argument the current number λ of unsuccessful attempts for routing a train. If λ is smaller than a given maximum number, as a strategy to provide the train with a greater flexibility while being routed, the procedure increases the train's fitting time, updates the input files of those territories being impacted by the new entry and exit times, and calls MP; otherwise, the algorithm resets the train's fitting time to its original value and asks MP to bring the train certain time later into the terminal to circumvent the congested network. Note that the algorithm never cycles because either trains increase their FT values, thus increasing the feasible operating domain, or they are brought to the entry point of the stitching network once the area becomes available. In the worst case, the trains might reach the end of the planning horizon, yet leading to a feasible solution.

When a model turns out infeasible, Alg. 2 also makes a call to the $ConflictRefiner$ procedure

Algorithm 3 *Adjustment*(Infeasible train i, λ_i, terminal $\bar{t}$)

1: IF $\lambda_i <$ max *attempts per train*

- Increase $q_i^{\bar{t}}$ in δ minutes based on train type.
- $\lambda_i = \lambda_i + 1$.

ELSE

- Enforce MP to bring train i δ' minutes later than its original $p_i^{\bar{t}}$
- Update $p_i^{\bar{t}}$ properly.
- $\lambda_i = 0$.

2: Update input files of territories being impacted by $q_i^{\bar{t}}$ and $p_i^{\bar{t}}$

3: Call MP to solve impacted territories properly.

4: **return** Updated entry and/or exit times for trains on terminal $\bar{t}$.

to confirm feasibility of entry and exit times and to determine whether or not MP needs to be called again to resolve any other territory where a time discrepancy or conflict among trains is observed.

The algorithm keeps on routing trains one by one while stitching all the dispatch territories until one of the following stopping criteria is met: reaching a maximum number of iterations or a given time limit, or once all trains have been successfully routed and a master plan can then be produced.

The core of Alg. 2 relies on two key components, namely: a time-space network (step 5), and an IP formulation (step 6). These components are constructed for each train in the system in a systematic way as the algorithm converges and are discussed next.

3.2.1. Model definitions

Let $G = (V, E)$ be a directed graph representing a stitching network area (e.g., networks such as $\mathcal{H}$ and $\mathcal{G}$ introduced in Section 2.1), where V and E are the node (junctions and control points) and arc (track) sets, respectively. Fig. 5 shows an example of G describing its main components, namely delayable and non-delayable arcs and traversal time τ.

Let $D_i \subset \tilde{E}_i \subseteq E$ be the set of delayable arcs inside the terminal for train $i \in \mathbb{I}$, where $\tilde{E}_i$ is the set of physical (directed) arcs allowable by train i. Let $\Gamma_i^1 \subseteq D_i$ be a set of unique arcs authorized for crew-change, fueling and similar required operations for train i. Similarly, let $\Gamma_i^2 \subseteq D_i$ and $\Gamma_i^3 \subset \tilde{E}_i \subseteq E$ be two sets of unique arcs authorized for inspection and yard-work exit points, respectively, for train i. Note that we differentiate among these unique sets because in practice the aforementioned

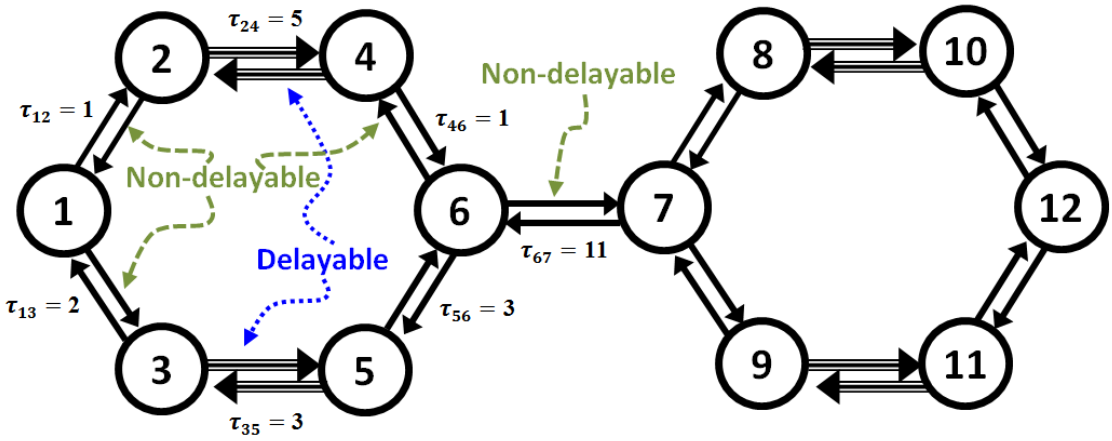

Figure 5 – Example of $G = (V, E)$ with delayable and non-delayable arcs and traversal time values.

work events are performed at distinct points within the stitching areas and delay trains with a considerable variation from one work event type to another.

The discrete-planning horizon interval is given by $H = [\min_{i \in \mathbb{I}} p_i, \max_{i \in \mathbb{I}} q_i]$, where p_i and q_i are the entry and exit times for train $i \in \mathbb{I}$, respectively.

Let τ_a^i and c_a^i be the traversal time and cost on arc $a \in \tilde{E}_i$, respectively, for train $i \in \mathbb{I}$. Let $\tilde{c}_a^i$ be the bonus of delaying train i on arc $a \in D_i$. Entry and exit nodes of train $i \in \mathbb{I}$ in G are given by α_1^i and α_2^i, respectively. By convention, for any node $u \in V$ and $i \in \mathbb{I}$, let $V_{iu}^- = \{v \in V : (v, u) \in \tilde{E}_i\}$ be the set of all tail nodes of incoming arcs, and let $V_{iu}^+ = \{v \in V : (u, v) \in \tilde{E}_i\}$ be the set of all head nodes of outgoing arcs. In addition, let ω_1 be the train separation parameter and let ω_2 be the crew-change, fueling and possible other operations parameter, both being defined by the dispatcher (typically, they are between 4 and 50 minutes). Similarly, let ω_3 and ω_4 be the parameters defined by the dispatcher for trains to perform their yard work (if any) and inspection, respectively (typically, ω_3 is a few hundred minutes while ω_4 is tens of minutes).

3.2.2. Time-space network construction

For each train $i \in \mathbb{I}$ we construct a directed network $N^i = (U, A_i)$, where U and A_i are the node and arc sets, respectively. Formally, we define $U = \{(u, h) \mid u \in V, h \in H\}$ and $A_i = A_i^1 \bigcup A_i^2$, with:

$$A_i^1 = \left\{ (u,h), (v, h + \tau_{uv}^i) \,\middle|\, h \in H, (u,v) \in \tilde{E}_i \backslash D_i \right\} \text{ (non-delayable arcs)},$$

$$A_i^2 = \left\{ (u,h), (v, h + \tau_{uv}^i + k) \,\middle|\, h \in H, k = 0, 1, 2, \cdots, (u,v) \in \tilde{E}_i \bigcap D_i \right\} \text{ (delayable arcs)}.$$

Fig. 6 shows an example of the resulting time-space network $N^i = (U, A_i)$ for a train originating at node 1 at minute 405.

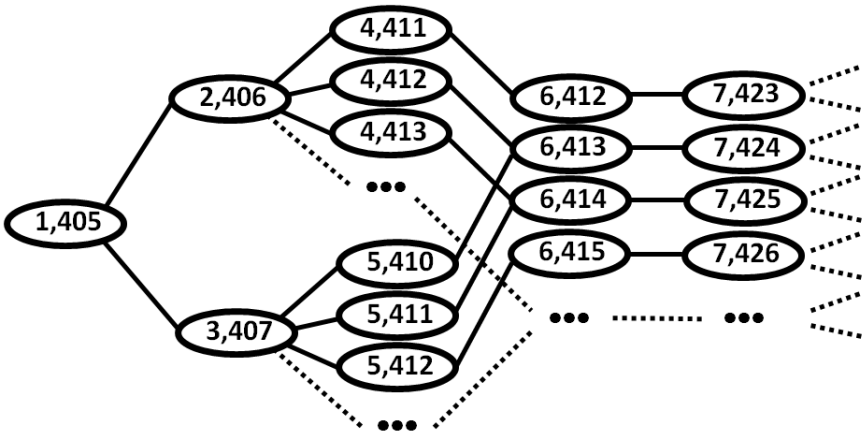

Figure 6 – Example of a time-space network with a train entering at node 1 at minute 405.

3.2.3. A multi-commodity formulation

Here we formulate the railway multi-territory dispatching problem for single train $i \in \mathbb{I}$ as an IP model. To do so, we make the following assumptions. We assume that trains' speed and physical location are known only at control points and select other locations. Acceleration and deceleration time losses are neglected. Specific work events to be conducted inside the terminals and the minimum dwell times for all the trains in the system are given in advance. Trains' type, priority, direction, and time and point of origin are also known. A finite network capacity within the terminals and a set of delayable arcs are assumed.

The following two types of decision variables are considered in the formulation.

$$y^i = \begin{cases} 1 & \text{if train } i \in \mathbb{I} \text{ is successfully routed through the stitching network} \\ 0 & \text{otherwise.} \end{cases}$$

$$x_a^i = \begin{cases} 1 & \text{if arc } a \in A_i \text{ is used by train } i \\ 0 & \text{otherwise} \end{cases}$$

We start by defining the following constraints for the construction of feasible train paths:

$$\sum_{\substack{a=((u,h_1),(v,h_2))\in A_i,\\ u=\alpha_1^i, v\in V_{iu}^+}} x_a^i = y^i \tag{2}$$

$$\sum_{\substack{a=((u,h_1),(v,h_2))\in A_i,\\ v=\alpha_2^i, u\in V_{iv}^-}} x_a^i = y^i \tag{3}$$

$$\sum_{\substack{a=((u,h_1),(v,h_2))\in A_i,\\ v\in V_{iu}^+}} x_a^i - \sum_{\substack{a=((v,h_1),(u,h_2))\in A_i,\\ v\in V_{iu}^-}} x_a^i = 0, \ \forall u \in U \backslash \{\alpha_1^i, \alpha_2^i\}. \tag{4}$$

As described in Section 2.1, there are specific sets of business rules imposed on the terminals.

Based on the time-space network defined in Section 3.2.2, these business rules can be mathematically modeled as follows.

Minimum Dwell Time Constraint:

$$\sum_{\substack{a=((u,h_1),(v,h_2))\in A_i \\ (u,v)\in D_i}} (h_2 - h_1 - \tau_a^i) \cdot x_a^i \geq \Delta_i \tag{5}$$

Crew-Change, Fueling and Other Operational Constraint:

$$\sum_{\substack{a=((u,h_1),(v,h_2))\in A_i \\ (u,v)\in \Gamma_i^1,\, (h_2-h_1-\tau_a^i)\geq \omega_2}} x_a^i \geq 1 \tag{6}$$

Inspection Constraint:

$$\sum_{\substack{a=((u,h_1),(v,h_2))\in A_i \\ (u,v)\in \Gamma_i^2,\, (h_2-h_1-\tau_a^i)\geq \omega_4}} x_a^i \geq 1 \tag{7}$$

Yard Work Constraint:

$$\sum_{\substack{a=((u,h_1),(v,h_2))\in A_i \\ (u,v)\in \Gamma_i^3\ (h_2-h_1-\tau_a^i)\geq \omega_3}} x_a^i \geq 1 \tag{8}$$

Eq. (5) imposes the minimum dwell time requirement given by Δ_i. Eq. (6) imposes the crew-change due time requirement given by minimum time ω_2 for train i, where Γ_i^1 is the set of delayable arcs inside the terminal designed for train i to perform its crew change. Eqs. (7) and (8) are defined analogously for inspection and yard work requirements, respectively.

Objective Function:

$$\max_{i\in\mathbb{I}} \lambda_i y^i - \sum_{a=((u,h_1),(v,h_2))\in A_i} c_a^i x_a^i + \sum_{a=((u,h_1),(v,h_2))\in A_i} \tilde{c}_a^i x_a^i \tag{9}$$

Eq. (9) states the objective function that minimizes the total cost incurred by traversing G. It consists of three terms.

- The train satisfaction factor denoted by λ_i captures the preference to route the train.
- The total penalty cost incurred by train i while traversing the stitching network. Basically, there is a high penalty c_a^i imposed on train i for entering/using siding arc a.
- The total bonus $\sum_{a\in A_i} \tilde{c}_a^i x_a^i$ for performing train's work events, e.g., inspection, crew change, fueling and other operations, on preferred arcs inside terminals and for delaying a train on most preferred arcs based on its priority and minimum dwell time values.

The underlying models corresponding to stitching network areas $\mathcal{H}$ and $\mathcal{G}$ are denoted by $IP_{\mathcal{H}}$ and $IP_{\mathcal{G}}$, respectively.

Algorithm 4 *PostProcessing*(Input: set of routed trains)

for each train i **do**
  if ($i \in W$ and $t = t_2$) or ($i \in E$ and $t = t_4$) **then**
    $\beta = (a_i - b_i)/2$
    Set $p_i^t = \beta$ and call MP to solve t
    IF no changes to any other train occurred in t
      Call *LongestPath* algorithm to get time bounds
      Construct corresponding time-space network
      Create and solve the underlying IP model
      If IP model is feasible:
        Update solution accordingly
        $a_i = \beta$
      ELSE
        $b_i = \beta$
    Repeat steps 3-13 until $|a_i - b_i| < \epsilon$ for train i

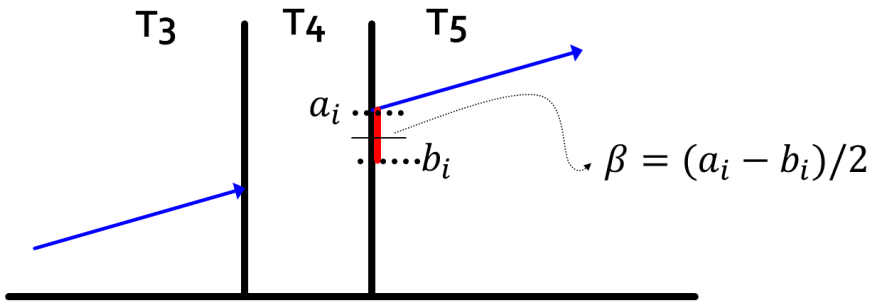

Figure 7 – Notation of a single iteration of the bisection approach.

3.3. Post-processing Phase

As described in the previous sections, FT values may be increased in δ units to strategically provide trains with enough time to perform their work events, or to be routed while having meet-pass events inside the terminals. As the algorithm iterates, due to the continuous increase in FT values, the resulting dwell times might observe a significant unnecessary slack. Hence, we apply a post-processing phase to minimize FT values inside the terminals, and thus also decreasing the slack of dwell times. This is accomplished by retrieving the feasible set of routed trains found in the stitching phase and applying a *bisection* approach to trains $w \in W$ and $e \in E$ traversing $\mathcal{H}$ and $\mathcal{G}$, respectively. Fig. 7 provides the notation of the bisection approach and its main components.

Alg. 4 shows the steps conducted in the post-processing phase. For each train $w \in W$ and $e \in E$ traversing $\mathcal{H}$ and $\mathcal{G}$, respectively, we iterate the following. Based on train i's current exit time $q_i^t = a_i$ from terminal t and its most optimistic potential exit time b_i, calculate the middle point $\beta = (a_i - b_i)/2$. Then assign β as the new potential entry time p_i^t to adjacent territory t for train i, and call MP to solve the territory. If no changes are observed in any other train's entry and exit times in t, we get time bounds by means of the *LongestPath* algorithm, we construct the time-space network and solve the underlying IP model. If the model is feasible, then we update the solution accordingly and move to the lower section of the bisection area (see Fig. 7), i.e., we set $a_i = \beta$; otherwise, we move to the upper section by setting $b_i = \beta$. We stop when $\mid a_i - b_i \mid < \epsilon$ for train i.

4. Study results

In this section we report our computational experiments on two test beds:

Type-1: Instances with only one terminal ($\mathcal{G}$) and two dispatch territories (T_3 and T_5, see Fig. 2). This set is composed of 15 test cases with a network configuration that includes the stitching area BHN_1 (see Section 2.1.)

Type-2: Instances with two terminals ($\mathcal{G}$ and $\mathcal{H}$) and three dispatch territories (T_1, T_3 and T_5, see Fig. 2). This set is composed of 4 test cases characterizing challenging scenarios: (1) maintenance of way (MOW) windows imposed on territories adjacent to $\mathcal{H}$, impacting eastbound arrivals at the terminals (scenario A); (2) MOW imposed on territories adjacent to $\mathcal{G}$, impacting westbound arrivals at the terminals (scenario B); (3) a scenario with severely infeasible line-ups (scenario C); and (4) a scenario with severe slacks in the line-ups (scenario D). Note that this set corresponds to a network topology that includes the stitching areas BHN_1 and BHN_2 (see Section 2.1.)

The total number of trains observed in the entire network of the instances found in the two sets above, ranges from 12 to 27. The planning horizon is set to 12 and 18 hours for type-1 and type-2 instances, respectively, with time discretization given in minutes. The instances were provided by our partner railway.

The algorithms are implemented in Java and the IP-models are solved with ILog Concert Technologies, IBM ILOG CPLEX Optimizer 12.3 library. All experiments were conducted on an Intel Xeon CPU 2.80-GHz server with 32 GB of RAM under a 64-bit MS-Windows Server operating system.

Next, we summarize the performance of the algorithm, and then in Section 4.2 we compare the algorithm versus manual benchmarks.

Table 1 – Computational results on type-1 instances.

Test case	# Iters	Dwell times			CPU time
		minimum	final	slack	
1	19	515	539	24	3.4
2	18	568	606	38	3.0
3	16	162	215	53	2.4
4	31	551	607	56	1.7
5	62	1026	1284	258	4.7
6	31	327	547	220	4.0
7	15	823	879	56	1.4
8	11	227	227	0	1.3
9	76	242	518	276	7.5
10	34	390	584	194	4.6
11	13	390	749	359	1.3
12	21	390	553	163	5.6
13	32	321	388	67	1.7
14	19	321	430	109	1.4
15	5	661	883	222	2.3

4.1. Computational results

Tables 1 and 2 show the computational results on type-1 and type-2 instances, respectively. In the tables, for each test case shown in column 1, the total number of iterations, the minimum and final dwell times, and the corresponding slack (for the corresponding stitching network area $\mathcal{H}$ and/or $\mathcal{G}$), and the total CPU time (in minutes) are given in the subsequent columns. Note that dwell-time values correspond to accumulative values per instance, i.e., a total value given by the sum of all dwell times across all trains at the terminals.

From Table 1, we observe that the algorithm required less than 3 minutes to solve 9 out of 15 cases while it took no longer than 7.5 minutes for the remaining instances. In summary, the maximum, minimum and average CPU times required by the algorithm were 7.5, 1.3 and 3.0 minutes, respectively. Regarding the dwell time slack, we observed a maximum, minimum and average values of 359, 0 and 139 extra minutes, respectively.

Table 2 – Computational results on type-2 instances.

Test case	# Iters	Dwell times for $\mathcal{H}$			Dwell times for $\mathcal{G}$			CPU time
		minimum	final	slack	minimum	final	slack	
A	13	840	858	18	676	753	77	3.0
B	35	790	919	129	90	124	34	3.5
C	41	1210	1587	377	120	315	195	5.2
D	17	670	1280	610	75	124	49	8.3

From Table 2, we observe that the algorithm required roughly 5 minutes to solve 3 out of 4 cases while it took 8.3 minutes to solve the remaining instance. In summary, the maximum, minimum and average CPU times required by the algorithm were 8.3, 3.0 and 5.0 minutes, respectively. Regarding the total dwell time slack per instance, we note a maximum, minimum and average values of 610, 18 and 288 extra minutes, respectively, for $\mathcal{H}$, and 195, 34 and 91 extra minutes, respectively, for $\mathcal{G}$.

On average, the total percentage of CPU time consumed by MP to obtain individual routing plans was 54% in type-1 instances, and 72% in type-2 instances.

4.2. Infeasible line-up analysis: Baseline discrepancies

In this section we present a thorough analysis on infeasible train line-ups if no stitching procedure is applied exclusively for type-2 instances. We classify trains per instance into two groups based on trains' departure times from terminals located along the railway network. Note that trains departing from these intermediate terminals are projections provided to individual dispatchers to route trains through their corresponding dispatch territories.

Group 1: Trains whose departure time from an intermediate terminal (stitching network area) given by the RTS projections are infeasible, i.e., the observed departure times are earlier than the actual departure times due to trains' traversal time through the terminal plus the enforced minimum dwell time to perform work events at the terminal.

Group 2: Trains whose departure time from an intermediate terminal (stitching network area) given by the RTS projections are later than the actual departure times due to the required amount of time for trains to perform their work events plus the time lost at the terminal because of congestion and traversal times.

This is accomplished by comparing train line-ups from (a) the RTS projections against (b) the final feasible line-ups provided by the stitching algorithm proposed in this work. Note that type-(a) line-ups are the original, initial line-ups provided by the real-time scheduling RTS system that dispatchers use to obtain train line-ups for a given territory independently from other territories. By investigating the discrepancies among the different line-ups, we identify the degree of infeasibility of RTS projections and the usefulness of the algorithm proposed in this work to provide more accurate arrival and departure estimations at the terminals.

The departure-time deviations (in minutes) of RTS projections from the stitching algorithm's projections are shown in Figs. 8–11 for scenarios A, B, C and D, respectively. The bar graphs represent the upward and downward shifts by train –represented by negative and positive values,

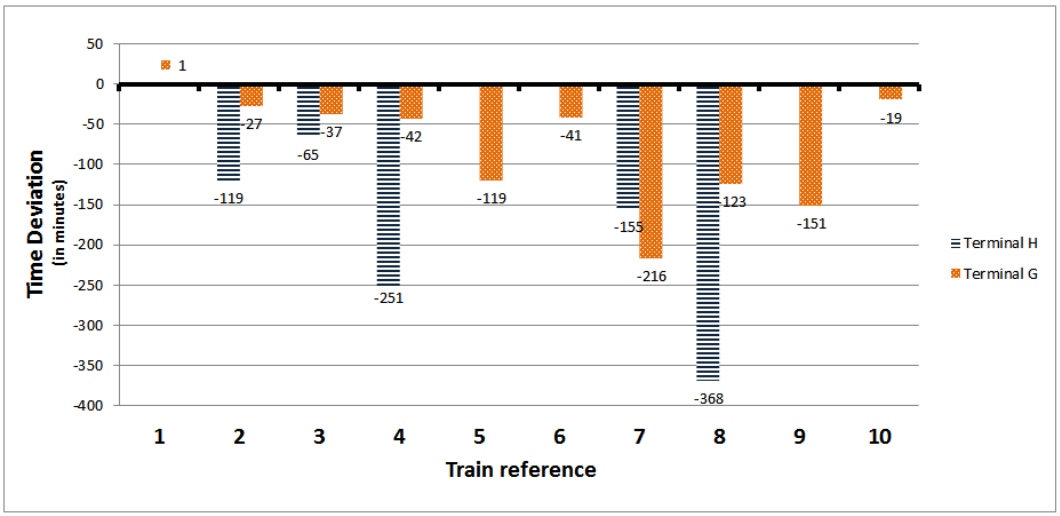

Figure 8 – Scenario A: Deviations of RTS projections from the stitching algorithm's projections at terminals H and G (by train)

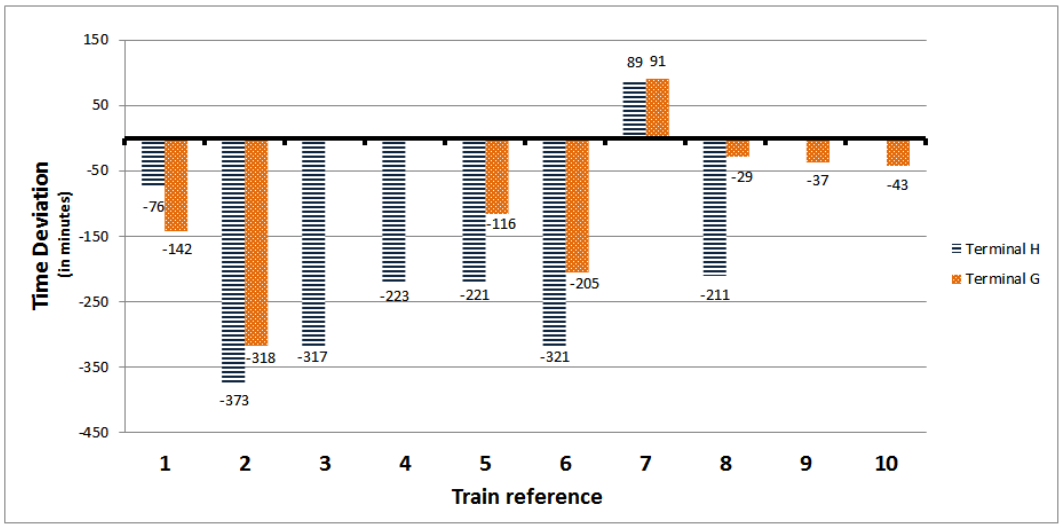

Figure 9 – Scenario B: Deviations of RTS projections from the stitching algorithm's projections at terminals H and G (by train)

respectively– that have to happen due to the congestion, meet-pass events and work events conducted inside terminals $\mathcal{H}$ and $\mathcal{G}$. Note that only those trains that actually exit the terminals before the maximum planning horizon is reached are included in the charts, thus the figures might contain bars for only one of the two stitching networks ($\mathcal{H}$ and $\mathcal{G}$).

The classification of trains per case into Groups 1 and 2 for the discrepancies observed between RTS projections and the stitching algorithm's projections are shown in Tables 3 and 4, respectively. In the tables, for each instance shown in column 1, a group of 4 columns is provided for each

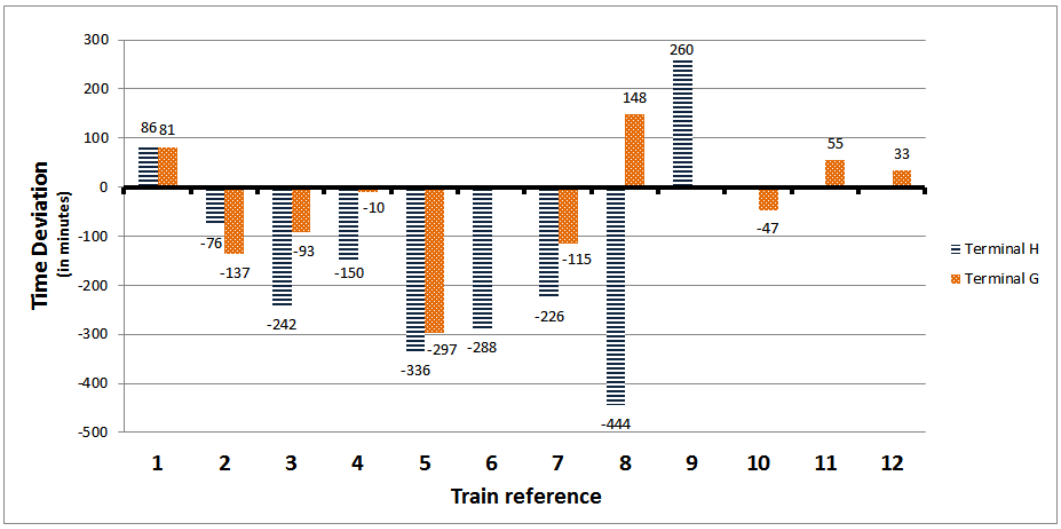

Figure 10 – Scenario C: Deviations of RTS projections from the stitching algorithm's projections at terminals H and G (by train)

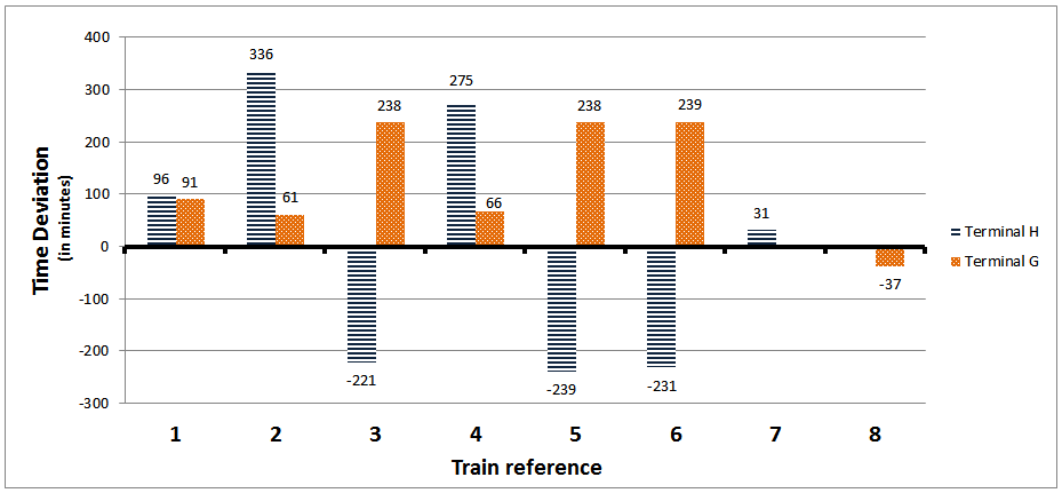

Figure 11 – Scenario D: Deviations of RTS projections from the stitching algorithm's projections at terminals H and G (by train)

stitching network $\mathcal{H}$ and $\mathcal{G}$. The columns show the number of trains having an infeasible schedule and the maximum, minimum and average amount of time (in minutes) of upward (Group-1) and downward (Group-2) shifts that occur due to the final intermediate departure times provided by the stitching algorithm's projections.

By taking into account the mean values among all the scenarios, from the results shown in Table 3, we observe that 6 out of 10 trains traversing the terminals are scheduled to be departing 3.8 hours later than the RTS projections in $\mathcal{H}$, and the same number of trains are departing 1.5

Table 3 – Group 1's summary for RTS projections vs. the stitching algorithm's projections

Test case	Upward shifts in $\mathcal{H}$ (in minutes)				Upward shifts in $\mathcal{G}$ (in minutes)			
	# trains	max	min	avg	# trains	max	min	avg
A	5	368	65	191	9	216	19	86
B	7	373	76	249	7	318	29	127
C	7	444	76	252	6	297	10	117
D	3	239	221	230	1	37	37	37

Table 4 – Group 2's summary for RTS projections vs. the stitching algorithm's projections

Test case	Downward shifts in $\mathcal{H}$ (in minutes)				Downward shifts in $\mathcal{G}$ (in minutes)			
	# trains	max	min	avg	# trains	max	min	avg
A	0	–	–	–	1	1	1	1
B	1	89	89	89	1	91	91	91
C	2	260	86	172	4	148	33	80
D	4	336	31	185	6	238	61	156

hours later in $\mathcal{G}$. We also observed up to 7.4 hours and 5.3 hours of upward shifts at $\mathcal{H}$ and $\mathcal{G}$, respectively. Analogously, from the results shown in Table 4, we observe that 2 out of 10 trains are departing 2.5 hours earlier in $\mathcal{H}$, and 3 out of 10 trains are departing 1.4 hours earlier in $\mathcal{G}$. In this comparison, we observe the maximum values of downward shifts of more than 4 hours at either terminal.

Two concluding remarks can be drawn from the summaries given in Tables 3 and 4. First, while observing large amounts of time of upward shifts between the two projections, we point out the infeasible departure times projected by the RTS system since it has no consideration of preceding meet-pass events, terminal capacities within the stitching networks, and congestion (due to work events) at the terminals. Second, it is also noticeable that whenever the RTS system projects extra amounts of dwell time inside the terminals (beyond the minimum dwell times), we observe an increase of the overall velocity in the whole system by suggesting shorter dwell times at the terminals, thus tighter schedules.

5. Conclusions

We have proposed a novel algorithm to efficiently tackle the railway multi-territory dispatch planning (RMTDP) problem. Regarding effectiveness, our algorithm shows average run times of 3.0 and 6.5 minutes to solve instances with 1 and 2 terminals, respectively.

The results obtained from the infeasible train line-ups analysis presented in Section 4.2 lead to the conclusion that using our algorithm provides more accurate projections of train arrivals at the terminals so as to plan their operations accordingly. In addition, since both stitching networks $\mathcal{H}$ and $\mathcal{G}$ (see Section 2.1) are crew change points, crew planners can call their crews according to

the updated line-ups. Moreover, the algorithm proposed in this research also acts as a point of interaction between dispatchers and chief dispatchers to resolve disputes, which is also attesting to the fact that without a proper train line-up at the terminals, the plans coming from MP may not reliable as MP does not consider interactions between dispatch territories.

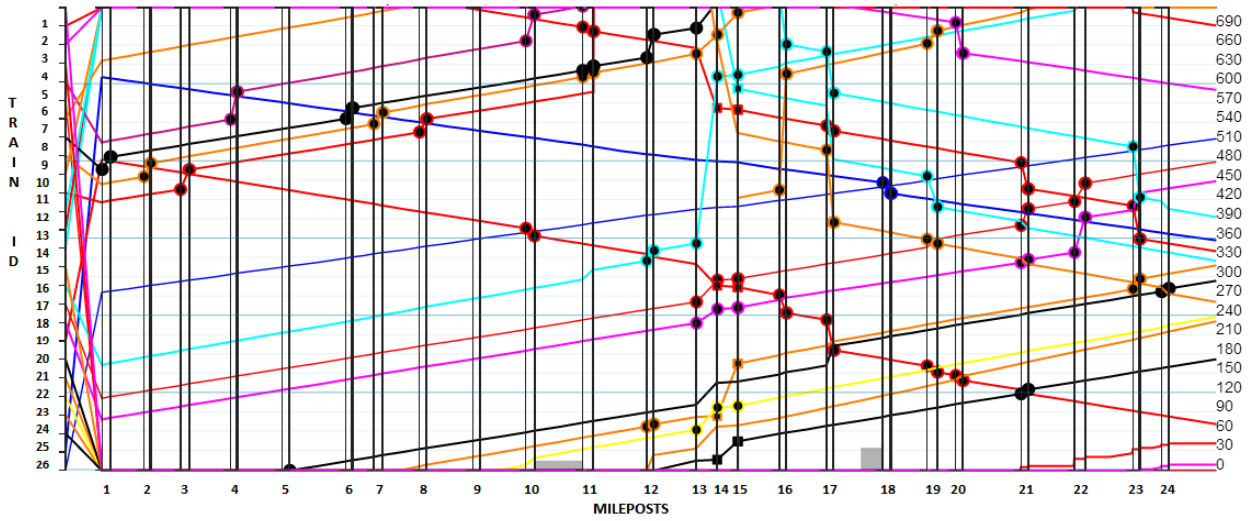

Figure 12 – Web-based GUI example: Scenario 5.

A web-based GUI application is also developed to analyze the stitching solutions produced by our algorithm. This application uses the x-axis to represent locations along the track line (e.g., siding stations) and the y-axis to represent discrete time values within the given planning horizon. Figure 12 shows an example of the web-based GUI application while displaying the stitching solutions for test case 5. In the figure, for each train, its path is shown by straight line segments that follow a train-type-based color pattern; its meet-pass events and required delays are represented by dots at specific locations; and MOW windows are represented by translucent grey rectangles.

Acknowledgements

Dr. Borraz-Sanchez and Professor Klabjan acknowledge Dr. Pooja Dewan, Dr. April Kuo, and Mike Swindler from BNSF for the overall insights and discussions provided to understand the problem and business needs.

References

[1] Alfonso, P.A., Bispo, C.F. Railway traffic management: Meet and pass problem. In: Proceedings of the LISS 2011 1st International Conference on Logistic, Informatics and Service Science, pp. 249, Beijing Jiaotong University, China (2011)

[2] Assad, A.A. Models for rail transportation. Transportation Research Part A, Vol. 14A, pp. 205–220 (1980)

[3] Boccia, M., Mannino, C., Vasilyev, I. The dispatching problem on multitrack territories: Heuristic approaches based on mixed integer linear programming. Networks. Vol. 62 (4), pp. 315–326 (2013)

[4] Carey, M. A model and strategy for train pathing with choice of lines, platforms, and routes. Transportation Research B, Vol. 28(5), pp. 333–353 (1994)

[5] Carey, M. Extending a train pathing model from one-way to two-way track. Transportation Research B, Vol. 28(5), pp. 395–400 (1994)

[6] Carey, M., Carville, S. Scheduling and platforming trains at busy complex stations. Transportation Research Part A, Vol. 37, pp.. 195–224 (2003)

[7] Carey, M., Lockwood, D. A model, algorithms and strategy for train pathing. Journal of the Operational Research Society, Vol. 46 (8), pp. 988–1005 (1995)

[8] Cordeau, J.F, Toth, P, Vigo, D. A survey of optimization models for train routing and scheduling. Transportation Science, Vol. 32(4), pp. 380–404 (1998)

[9] Corman, F. Rail-time railway traffic management: Dispatching in complex, large and busy railway networks. Ph.D. Thesis, TRAIL Thesis Series T2010/14, the Netherlands TRAIL Research School, Netherlands. (2010)

[10] Higgins, A., Kozan, E., Ferreira, K. Optimal scheduling of trains on a single line track. Transportation Research Part B. Vol. 30(2), pp. 147–161 (1996)

[11] Jovanović, D., Harker, P. Tactical scheduling of rail operations: The SCAN I system. Transportation Science, Vol. 25, pp. 46–64 (1991)

[12] Lin, S., Balakrishnan, A., Uygur, A. An optimization based decision support system for train dispatching. Working Paper, University of Texas at Austin, Austin, Texas. (2012)

[13] Mees, A.I. Railway scheduling by network optimisation. Mathematical Computer Modelling, Vol. 15, pp. 33–41 (1991)

[14] Petersen, E.R., Taylor, A.J., Martland, C.D. An introduction to computer assisted train dispatch. Journal of Advanced Transportation, Vol. 20, pp. 63–72 (1986)

[15] Szpigel, B. Optimal train scheduling on a single line railway. Operations Research, Vol. 72, pp. 344–351 (1973)

[16] Tórnquist, J. Computer-based decision support for railway traffic scheduling and dispatching: A review of models and algorithms. In: Proceedings of 5th Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS'05), Palma de Mallorca, Spain, October (2005)
DOI: 10.4230/OASIcs.ATMOS.2005.659