Omega 116 (2023) 102796

Contents lists available at ScienceDirect

Omega

journal homepage: www.elsevier.com/locate/omega

Optimal Train Rescheduling in Oslo Central Station☆

Carlo Mannino[b,a], Andreas Nakkerud[a,b,*]

[a] Department of Mathematics, P.O box 1053 Blindern, 0316 OSLO, Norway
[b] SINTEF, P.O box 124 Blindern, 0314 OSLO, Norway

ARTICLE INFO

Article history:
Received 6 August 2021
Accepted 2 November 2022
Available online 13 November 2022

2010 MSC:
90B06
90B20
90C06
90C08
90C11
90C90

Keywords:
Integer programming
Optimization
Rail transport
Dispatching
Scheduling
Routing

ABSTRACT

Real-time train dispatching (i.e., rescheduling and replatforming) in passenger railway stations is a very important and very challenging task. In most major stations, this task is carried out by hand by highly trained dispatchers who use their extensive experience to find near-optimal solutions under most conditions. With several simultaneous deviations from the timetable, however, the traffic situation may become too complex for any human to handle it far beyond finding feasible solutions. As part of a prototype for a dispatching support tool developed in collaboration with Bane NOR (Norwegian rail manager), we develop an approach for Optimal Train Rescheduling in large passenger stations. To allow for replatforming, we extend the standard job-shop scheduling approach to train-scheduling, and we develop and compare different MILP formulations for this extended approach. With this approach, we can find, in just a few seconds, optimal plans for our realistic instances from Oslo Central Station, the largest passenger train hub in Norway. The prototype will be tested by dispatchers in the greater Oslo area, starting from the fall of 2021.

1. Introduction

Like all management of critical infrastructure, train dispatching is heavily regulated. Under the current system, all dispatching decisions must be made by highly trained human dispatchers. Therefore, the only practical way to introduce optimization into the process is through decision support tools. This work is part of the GOTO project [1] with Norwegian rail manager Bane NOR. The GOTO project aims to deliver an optimization-based decision support tool for dispatching trains in Oslo Central Station and other large passenger train stations. While the tool we develop is aimed at Oslo Central Station, the algorithms we present are general and not tailored to this station. The layout of Oslo Central Station (Figure 10) is typical of large passenger train stations.

In order to have a decision support tool accepted, we must make sure that our approach can at least match or, better, outperform the human dispatchers under normal traffic conditions. Under these conditions, human dispatchers can use their expert intuition to evaluate suggested solutions and compare them to the near-perfect solutions they produce. Based on evaluations under normal traffic conditions, dispatchers may come to trust the suggested solutions in heavily congested traffic situations where no human can expect to capture the complete picture. It follows that we must model the infrastructure and business rules with a very high level of detail to produce high-quality solutions under any conditions.

The need for automated decision support can only be expected to grow with the introduction of new technology in railway signaling. Currently, almost all dispatching is based on a fixed division of the track infrastructure by signals. Level 3 of the European Train Control System (ETCS) introduces *moving blocks* (see [2]), where trains are protected by safe zones determined by breaking distances rather than by signals at fixed locations. As these new control systems are introduced, they will increase the flexibility of train rescheduling and make it even harder to solve the train dispatching problem optimally by hand. In order to make full use of the flexibility introduced by moving blocks, we will require fine-grained scheduling approaches.

An extensive research effort has gone into real-time train rescheduling problems (see, for example, survey papers [3–5]), but the research primarily covers simple railway network designs (for

☆ Area: Production Management, Scheduling and Logistics. This manuscript was processed by Associate Editor Prof. Benjamin Lev. This work was funded by The Research Council of Norway [grant numbers 237718, 267554, 300509].

* Corresponding author.
E-mail address: andreas.nakkerud@sintef.no (A. Nakkerud).

https://doi.org/10.1016/j.omega.2022.102796
0305-0483/

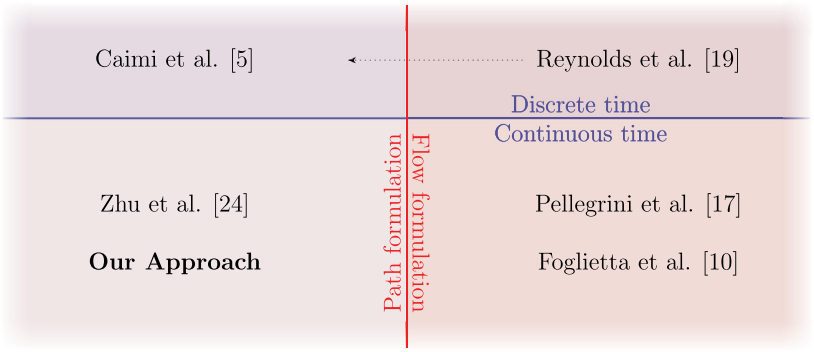


Fig. 1. Categorization of some recent papers on replatforming and rescheduling in railway stations.

recent works on rescheduling problems on a single line, see for example [6,7]). Only a few works are devoted explicitly to dispatching trains in (large) railway stations. The works typically make use of Mixed Integer Linear Programming (MILP) formulations [8]. We can identify two major classifications according to the way scheduling and platforming (routing within the station area) are represented in the models.

- For the scheduling part, two main streams of MILP models are applied in the literature: *big-M* formulations and *time-indexed* formulations [9]. In both models, the path of a train through the stations or lines is subdivided into smaller segments (sometimes down to the physical *track circuits*, which are the smallest regions in the train detection systems). In big-*M* formulations, for each train and each segment in its path, we have a continuous variable representing the time in which the train enters the segment. The drawback of this approach is that we need to introduce a disjunctive constraint to represent the order in which two trains travel through a contended track. These, in turn, are translated into linear constraints by introducing binary variables and the so-called big-*M* constraints, i.e., constraints containing some very large coefficients–notoriously weakening the formulation [9]. In time-indexed models, the planning horizon is discretized into small time periods. A binary variable is associated with each train, each segment in its path, and time period. The resulting formulations are typically stronger than their big-*M* counterparts, but they have a much larger number of variables and constraints, slowing down the solution process. The smaller the time period, the larger the number of variables: on the other hand, large time periods lead to a poor approximation of the train movement through the station, which may end up generating suboptimal solutions, or even in solutions that cannot be implemented in practice [10].
- For the platforming part, we can identify two major categories according to how paths through the station are represented. In *multicommodity flow* approaches [11], a binary variable x is associated with each train and each segment of its path, and $x = 1$ denotes that the train will run through the segment. In this class of approaches, the train path is constructed directly by the model. The drawback of this approach is that the model must incorporate flow constraints to represent paths through the station. In *path-based* approaches, we have a binary variable associated with each train and each potential path of the train through the station. The drawback of this approach is that the number of paths may grow exponentially with the station's size.
  Finally, the two approaches may be combined using Dantzig-Wolfe decomposition and column generation (see [12]). With this technique, a path-based MILP is constructed iteratively by solving a sequence of single-commodity flow subproblems.

Figure 1 shows how some recent papers on the topic are divided according to the classifications described above.

Outside these major classes of approaches, there are some simulation-based and heuristic approaches.

Reynolds et al. [13] present a time-indexed multicommodity flow model for rescheduling and replatforming. In their approach, they then transform their formulation into a path-based one by Dantzig-Wolfe decomposition. They then solve this formulation by branch-and-price and column generation [14]. Reynolds et al. apply their approach to solve instances with up to 32 trains (1 hour of traffic) in a large station area. They use a 15-second time step for their time-indexed formulation and 30-second margins to account for business rules that their model does not take into account.

Caimi et al. [15] present a discrete-time path-based formulation for rescheduling and replatforming. They introduce blocking stairways, which detail the speed profile of a train and when the train blocks different segments. The trains are then assigned to available blocking stairways for entry to and exit from the station. If no feasible solution can be found, additional blocking stairways are generated in a column generation fashion. The blocking stairways allow the track infrastructure to be modeled to the level of track circuits, but many blocking stairways may need to be generated. Caimi et al. apply their approach to instances of the central station of Berne, Switzerland, where they solve a whole operational day (roughly 1500 trains) in about 11/2 hours.

Pellegrini et al. [16] present a continuous-time multicommodity-flow-like approach. In their approach, like in that of Raynolds et al. [13], the routing is left to the solver. They also model the track infrastructure at the level of track circuits, which is the highest available resolution for train position detection in most current signaling systems. Pellegrini et al. apply their approach to Lille-Flandres station, where they solve instances with up to 47 trains, but only for up to 450-second (71/2-minute or 1/8-hour) periods.

Zhu et al. [17] present a continuous-time path-based formulation similar to the one we present here. However, their formulation is slightly simplified and is used to solve smaller instances in order to support an overarching agent-based approach. Zhu et al. show detailed analyses of computational results for MILP instances with four trains. He et al. [18] also present a similar path-based formulation, but they use it as part of a simulation-based approach rather than as a MILP formulation.

To our knowledge, although tested on real-life or realistic instances, none of the above approaches have been implemented in control centers and tested or adopted by operative dispatchers. Foglietta et al. [19] present a *heuristic* approach that was in operation to support dispatchers in Roma Tiburtina. While their paper also describes an exact, flow-based IP model, this model required a commercial solver and was not applied in the station.

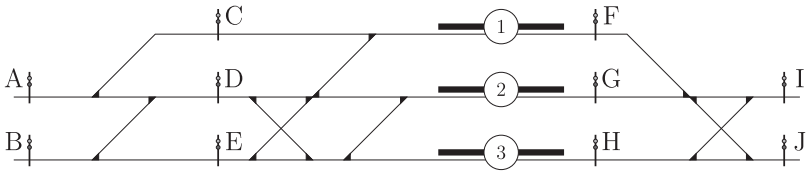


Fig. 2. We represent a station by a set of signals and a set of interlocking routes connecting pairs of signals. All train movements in the depicted area is from left to right.

As this paper is part of the research project GOTO in collaboration with Bane NOR, our modeling requirements have been guided by the use-cases at Bane NOR. In particular, we want to focus on real-time dispatching support for Oslo Central Station. This station acts as a hub, connecting traffic bound for the south-east and the south of Sweden; for the east and central Sweden; for the north; for the south-west; and for the west. Delays in Oslo Central Station can have knock-on effects on the entire Norwegian rail network, both for passengers and cargo. Since much of the Norwegian network is made up of single-track lines with limited numbers of meeting point, it is often difficult to make up delays. It is therefore very important to resolve delays at Oslo Central Station as quickly as possible.

Trains entering and leaving Oslo Central Station must be highly coordinated since they often share track resources. To avoid causing unnecessary delay, we must model the infrastructure and business rules of the station very accurately, and we must avoid adding catch-all buffer times. In late 2019, the Director for Customers and Traffic of Bane NOR, the Norwegian rail manager, told Norwegian newspaper Aftenposten [20] that to increase punctuality for 2020, trains will close their doors 20 seconds before their scheduled departure. This statement indicates that the flow of passenger-train traffic is sensitive to very small delays and that we must aim for a very fine time resolution.

In order to achieve sufficient accuracy in conflict detection, we model the track infrastructure on the level of track circuits [2], which offer the finest resolution of train location in fixed-signal based train control. For the schedule, we use signal-level resolution. Under normal operations, the signals define the finest resolution for train control. Then, we extend the alternative-graph big-*M* formulations of Mannino et al. [21–23] to include path selection, and we develop continuous-time formulations for the train dispatching problem in the large, hub-like passenger station Oslo Central Station in Norway.

We adopt the path-based approach for platforming and assign collections of possible paths to each train. Candidate paths are preselected based on observations of traffic and discussions with dispatchers in Oslo Central Station. As a result, we can offer the flexibility in routing expected by dispatchers while also finding optimal dispatching solutions in a reasonable time for real-time applications. Using 1 hour of observed rush-hour traffic, we craft instances of 6 hours of rush-hour traffic (330 trains) and instances with 24 hours of traffic with morning and afternoon rush-hours (1032 trains). In both cases, we solve all instances to optimality in a reasonable amount of time for real-time applications in dispatching.

We use long planning horizons to stress test the model. While we have access to timetables and detailed infrastructure data for Oslo Central Station, we lack the detailed timing data needed to ensure we are solving exactly the same problem as the dispatchers. We therefore craft a set of large instances designed to push the algorithm. In the real-time traffic information systems used in the prototypes delivered by the GOTO project, trains are entered up to 24 hours in advance. This is why we choose 24 hours as the longest horizon in our tests.

A support tool built on our algorithm will be field-tested by dispatchers at Oslo Central Station, starting in the fall of 2021. This field test is part of the ongoing GOTO project, which has delivered a line dispatching prototype already in active testing on the lines incident to the station.

2. The Optimal Dispatching Problem

In this section, we give a formal description of the optimization problem tackled in this paper. We consider the simultaneous rescheduling and replatforming of passenger-train traffic through large passenger stations. The combination of rescheduling and replatforming is the typical task of dispatchers. The Optimal Dispatching Problem (ODP) is the task of assigning tracks and schedules to trains in a way that minimizes delays or maximizes passenger utility. In this paper, we aim to minimize the (weighted) sum of delays for all trains. For our computational experiments, we solve ODP for Oslo Central Station, the largest hub for passenger-train traffic in Norway. We consider a scheduling horizon of up to 24 hours.

2.1. Track Infrastructure: Signals and Interlocking Routes

On the most basic level of scheduling, we represent the track infrastructure of a station as a set of signals and a set of *interlocking routes*, which are the track sections connecting two successive signals. The movement of a train can be decomposed into a sequence of elementary movements, one for each interlocking route of its path. This decomposition is of particular practical interest since, under normal operations, the interlocking routes are at the highest level of precision in scheduling train movements in signal-based train control systems [2]; dispatchers control trains on the level of signals.

Figure 2 shows an example of a station with three platforms. Often, an interlocking route is uniquely determined by the signals it connects, but not always. In Figure 2, there are two possible interlocking routes connecting signals *D* and *G*. An interlocking route cannot pass a signal, so the remaining interlocking routes are uniquely defined by the signals they connect. Signals are directional, and signals for opposite directions need not be placed at the same point along the tracks. Therefore, there need not be any correspondence between interlocking routes in opposite directions.

2.2. Paths and Station Platform Tracks

The station in Figure 2 has three platforms, drawn as solid, black rectangles with a circle containing the track number. Each train passing the station will have a set of permitted paths through the station, where a path is a sequence of interlocking routes. One or more paths may be preferred for a specific train, for example, paths using one of the tracks adjacent to a given platform. We may associate a cost with the choice of path.

In principle, any physically connected sequence of interlocking routes can be a feasible path. Usually, however, only a few paths are actually available to a given train because of business rules and other operational considerations. In a large station like Oslo Central Station, it is typically required that a train stops at its designated track or, possibly, the track opposite on the same physical platform.

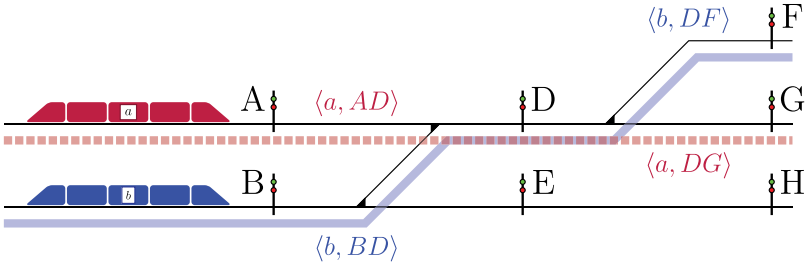


Fig. 3. Train *a* (purple) is about to enter interlocking route *AD*, and train *b* (blue) is about to enter interlocking route *BD*. Since the two interlocking routes physically overlap, one of the trains must wait.

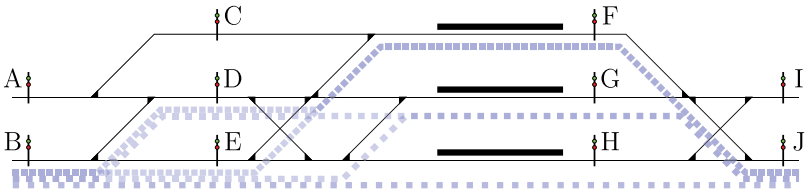


Fig. 4. An example collection of paths for a train going through the station from signal B to signal J. Not all possible paths are included in the collection.

2.3. Timetable and Delays

For each train, we are given a set of stations where the train is supposed to stop and the scheduled arrival and departure times at these stations. Together, these are referred to as the timetable. Given a station, our task is to decide, for each train, which path it will take and when it will pass each signal on its path. In general, we may be given a scheduled arrival time and an earliest departure time for any signal.

When we reschedule, the *delay* of a train in a station is the difference between the (re-)scheduled arrival time and the arrival time in the timetable, or 0 if the difference is negative. We assume that the timetable is independent of the choice of path through the station. That is, scheduled arrival and departure times do not depend on the choice of platform track.

2.4. Scheduling Conflicts

When two trains are set to use the same interlocking route (or a pair of physically overlapping interlocking routes), we have a *potential scheduling conflict* as shown in Figure 3. In Figure 3 we identify two potential scheduling conflicts. The first is between train *a* entering interlocking route *AD* and train *b* entering interlocking route *BD*, and the other is between *a* entering *DG* and *b* entering *DF*. Generally, a pair of interlocking routes is *incompatible* if some physical restriction or business rule limits their simultaneous use. An interlocking route will always be incompatible with itself.

A potential conflict is *realized* by a given schedule only if, according to the schedule, the two trains *simultaneously* occupy the interlocking route(s) generating the conflict. It is apparent that if a schedule realizes a conflict, then the schedule is not feasible. We will use the term *candidate schedule* when we want to emphasize that the schedule may be infeasible.

3. The Model

3.1. Route Nodes and Schedules

As discussed in the previous section, and illustrated in Figures 2 and 3, the movement of a train can be decomposed into a sequence of elementary movements, each through an interlocking route of the train's path. We assume the speed of a train to be constant through an interlocking route so that movement can be described by the entry time of the train in each interlocking route of its path. We assume unique entry signals to and exit signals from the modeled area for each train.

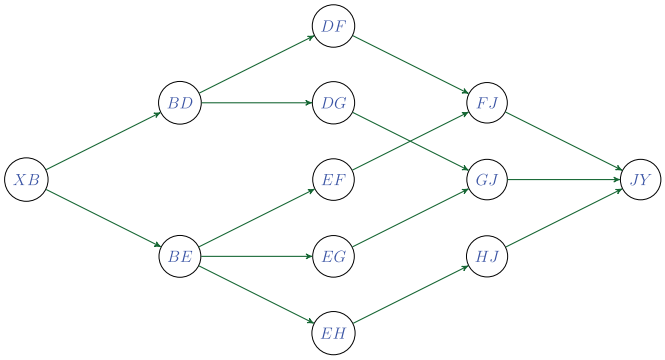


Fig. 5. The figure shows the directed graph associated with the interlocking routes of the paths highlighted in Figure 4. *XB* is the entry interlocking route leading to signal *B*, and *JY* is the exit interlocking route leading away from signal *J*.

Figure 4 shows a collection of five paths from signal *B* to signal *J* through the station in Figure 2. Each path consists of three interlocking routes. Figure 5 is a graph representation of how the interlocking routes in Figure 4 are connected into paths through the station. Each directed edge in Figure 5 represents a permitted transition from one interlocking route to another, and each directed path in the figure represents an available path through the station. The nodes labeled *XB* and *JY* are the entry and exit interlocking routes, respectively. Note that Figures 4 and 5 represent the same possible collection of path options available to a train passing the station. Different trains may have different path options, and the picture in the figures is not a complete representation of all the path options in the station.

For each train *a*, and each interlocking route *r* that may be used by *a*, we define the *route node* $\langle a, r\rangle$. We let $\mathcal{N}(a)$ be the set of all route nodes of train *a*. Informally, a route node $\langle a, r\rangle \in \mathcal{N}(a)$ represents the occupation of the interlocking route *r* by train *a*. A path for *a* through the station corresponds then to a subset of route nodes in $\mathcal{N}(a)$.

We let $\mathcal{A}$ be the set of all trains, and define $\mathcal{N} = \{o\} \cup \bigcup_{a\in\mathcal{A}} \mathcal{N}(a)$. That is, $\mathcal{N}$ is the set of all route nodes associated with the trains, plus a special node *o* which represents the origin of the planning horizon. We let $\mathcal{N}_S \subset \mathcal{N}$ be the set of sink (or terminal)

route nodes, i.e., the route nodes representing the end of the journey of a train (in the modeled area).

A *schedule* is a function $\mathbf{t} : \mathcal{N} \to \mathbb{R}$. We let $t_u = \mathbf{t}(u)$. A schedule associates a time to each route node, and $t_{\langle a,r\rangle}$ is the time train a enters interlocking route r, if r belongs to the path chosen for a. Note that, since there are alternative paths available, an interlocking route r available for a may not be chosen. In this case, $t_{\langle a,r\rangle}$ may assume any value.

The time t_o associated with the origin is the start time of our planning horizon, and we have

$$t_u \geq t_o \qquad u \in \mathcal{N} \tag{1}$$

For ease of explanation, through this section, we assume that the path through the station, i.e., the sequence of interlocking routes, is fixed in advance for any train a. In this case, the graph of available interlocking routes (Figure 5) reduces to an oriented, simple path and the set $\mathcal{N}(a)$ to the nodes in this path.

3.2. Release Times and Free Running

When we consider the schedule of an individual train in isolation, without any interaction with other trains, we say we are considering the *free running* of the train. In free running, a train's schedule is only determined by the train's time to traverse interlocking routes and by constraints on departure times.

If $u = \langle a, r_i\rangle$ is a route node of train a, and r_{i+1} is the route following r_i on the path of a, then we let $u + 1 = \langle a, r_{i+1}\rangle$. That is, $u + 1$ is defined for all $u \in \mathcal{N} \setminus \mathcal{N}_S$. We let $L_{\langle a,r\rangle}$ be the time it takes a to traverse r, so we have the following *traversal time constraint*

$$t_{u+1} - t_u \geq L_u \qquad u \in \mathcal{N} \setminus \mathcal{N}_S. \tag{2}$$

Trains typically follow a public timetable and cannot depart from a station before the officially scheduled time. Furthermore, we need to specify when trains enter the dispatching area being modeled. At the beginning of a train's path or at a station platform, we limit the train's earliest departure time, which is the earliest the train may enter the following interlocking route. We let $\Gamma : \mathcal{N} \to \mathbb{R}$, where $\Gamma_u = \Gamma(u)$ is the *earliest release time* of u relative to t_o, and get the *release time constraint*

$$t_u - t_o \geq \Gamma_u \qquad u \in \mathcal{N} \tag{3}$$

If the train is subject to a *no-wait* condition, the inequalities (2) and (3) may become equalities. E.g., if the train is not allowed to stop at a certain signal or the time a train enters the controlled area is fixed. If there is no earliest release time for u, we make the constraint (3) redundant by setting $\Gamma_u = 0$; by (1), we already have $t_u \geq t_o$.

3.3. Timetable and Objective

A standard way to assess the quality of a schedule $\mathbf{t}$ is by comparing it to the published timetable T. The timetable will specify target arrival and departure times at specific points in the network, called *timing points*. From the point of view of dispatchers, these points are normally the home signals and exit signals of stations with scheduled stops. The times at these points are denoted as, respectively, *arrival time* (at the station) and *departure time* (from the station). The quality of schedule $\mathbf{t}$ is measured by a cost function $c(\mathbf{t}, T)$, which typically penalizes delays of trains at their timing points. For passenger trains, the cost function may only penalize delays at arrival since these are the ones that most influence passenger utility; a delayed departure is not a problem if the train catches up by the next station.

More formally, we let $\mathcal{N}_T$ be a set of route nodes designated as *timing nodes*. A *timetable* is a function $T : \mathcal{N}_T \to \mathbb{R}$. We let $T_u = T(u)$, so that $T_{\langle a,r\rangle}$ is the *target time* (or *target entry time*) of train a in route r (or at $\langle a, r\rangle$). We define the *delay* at each timing node $u \in \mathcal{N}_T$ as $t_u - T_u$ if $t_u > T_u$, and 0 otherwise, and introduce the *delay variable* η_u, with

$$\eta_u = \max(0, t_u - T_u) \qquad u \in \mathcal{N}_T \tag{4}$$

Note that, depending on the route r, the target (entry) time may be an arrival or departure time. If T_u is the target entry (resp. arrival, departure) time at u, then t_u is the scheduled entry (resp. arrival, departure) time at u and η_u is the delay in entry (resp. arrival, departure) time at u.

3.4. Potential Conflicts and Selection Constraints

Until now, all our discussions have been about the free running of trains. In order to take the interaction between trains into account, we must now consider all potential scheduling conflicts between trains. Figure 3 shows an example of two trains crossing paths.

As described in Section 2, a potential conflict exists when two trains a, b make use of two incompatible (e.g., overlapping) interlocking routes r, q, respectively. In this case, we say a potential conflict exists between route notes $\langle a, r\rangle$ and $\langle b, q\rangle$. In Figure 3, $\langle a, AD\rangle$ and $\langle b, BD\rangle$ are in potential conflict, and so are $\langle a, DG\rangle$ and $\langle b, DF\rangle$.

Conflicts cannot occur in an actual schedule, and so we must decide which train goes first. If a goes first, then b can enter q only after a has left r. Vice versa, if b goes first, then a can enter r only after b has left q. This disjunctive precedence condition translates into a disjunctive constraint on the schedule of suitable route nodes on the paths of a and b.

We let $\mathcal{K}$ be the set of pairs of route nodes in potential conflict. Then, the following disjunctive constraint must be satisfied by every feasible schedule:

$$\begin{array}{l} t_v - t_{u+1} \geq \delta_u \\ \quad \text{or} \\ t_u - t_{v+1} \geq \delta_v \end{array} \qquad \{u, v\} \in \mathcal{K} \tag{5}$$

where δ_u for $u = \langle a, r\rangle$ is the time it takes the length of train a to pass the signal at the end of interlocking route r, thus clearing the way for the next train.

4. Path Selection

In this section, we show how to extend our model to consider the existence of alternative paths for a train through the station. Different paths may exist from the entry point to the platform track and from the platform track to the exit point. Even the choice of platform (or platform track) may not be fixed in advance, although the official timetable may indicate a preferred platform (or platform track). Each path is a sequence of interlocking routes, as pictured in an example with two stations in Figure 6. As the figure shows, the number of possible paths can grow exponentially with the number of locations where multiple routing options are available.

Figure 7 shows the station from Figure 4 with some of the possible paths drawn in. Paths 5–9 are the paths shown in Figure 4. The paths entering from signal A and exiting through signal I are shown as solid, purple lines. The paths entering from signal B and exiting through signal J are shown as dashed, blue lines. We note that the interlocking routes DG and EG are not uniquely defined by their end signals. We define DG and EG as they are drawn in Figure 7. The alternative interlocking routes (with the detour to the lower track for DG and the earlier change to the middle track for EG) could be added under different names, which would increase the number of possible paths.

We now extend our model to allow for path selection. As before, we let $\mathcal{A}$ be the set of trains, and for train a, we let $\mathcal{N}(a)$

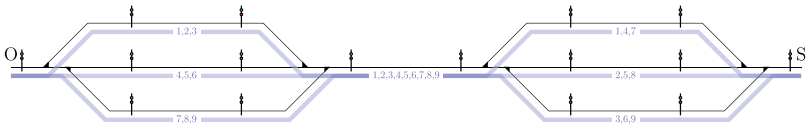


Fig. 6. There are 9 possible paths from O to S. For a train travelling through a sequence of stations with multiple paths through each station, the total number of paths is, in the worst case, exponential in the number of stations.

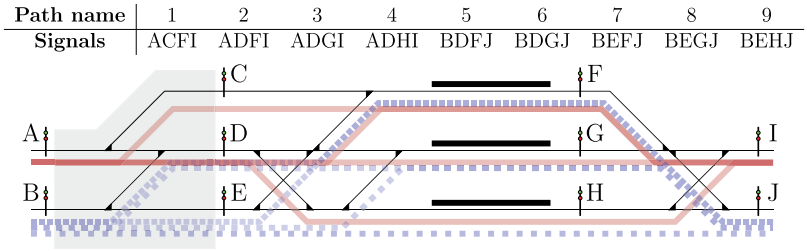


Fig. 7. A collection of paths through the station in Figure 2. Each path is made up of three interlocking routes, identified by the adjacent pairs of signals in the path definition. The interlocking routes are uniquely defined by the names of their end signals, with the exception of DG and EG, which we define as they are drawn in the figure.

be the set of route nodes for a and $\mathcal{P}(a)$ be the set of paths available to a. Any path $p \in \mathcal{P}(a)$ is an ordered sequence of route nodes in $\mathcal{N}(a)$. If u is a route node of p, we denote by u_p^+ the route node which follows u on p (if it exists). For a node $u \in \mathcal{N}(a)$, we let $S(u) \subseteq \mathcal{N}(a)$ be the set of potential successors of u. That is, if $v \in S(u)$ then there is at least one path $p \in \mathcal{P}(a)$ such that $v = u_p^+$. For a node $u = \langle a, r \rangle \in \mathcal{N}(a)$, we denote by $\rho(u)$ the set of paths in $\mathcal{P}(a)$ which goes through (i.e., uses or contains) interlocking route r. Now, let $\mathcal{P} = \bigcup_{a \in \mathcal{A}} \mathcal{P}(a)$. We define the *path variable* $w_p \in \{0, 1\}$ for $p \in \mathcal{P}$, which is 1 if and only if path p is selected. Since each train must be assigned exactly one path through the modelled area, we get the *path selection constraint*

$$\sum_{p \in \mathcal{P}(a)} w_p = 1 \qquad a \in \mathcal{A} \tag{6}$$

Next, for all route nodes $u \in \mathcal{N} \setminus \{o\}$ we introduce a variable $z_u \in \{0, 1\}$ which is 1 if and only if (a path containing) u is selected. We get

$$z_u = \sum_{p \in \rho(u)} w_p \qquad u \in \mathcal{N} \setminus \{o\} \tag{7}$$

If $z_u = 1$ we say that node u is *active* (or selected).

Many of the constraints we have introduced in Section 3 now depend on the choice of path for each train. We generalize these constraints in the following sections.

4.1. Path-Dependent Free Running Constraints

The following set of inequalities is a generalization of (2) to the case with path selection:

$$t_{u_p^+} - t_u - L_u \geq M(w_p - 1) \qquad u \in \mathcal{N} \setminus \mathcal{N}_S, p \in \rho(u) \tag{8}$$

where M is a suitable large positive constant. When $w_p = 1$, then (8) reduces to (2), otherwise it becomes redundant (this is the *big-M trick*). It is well known that big-M constraints are weak, in the sense that they do not help improve the value of the linear relaxation of the MILP formulation [8].

In a station with parallel platform tracks (as in Oslo Central Station), one can show that the system of inequalities (8) is dominated by the following family:

$$t_v - t_u \geq L_u \sum_{p \in \rho(u) \cap \rho(v)} w_p \qquad u \in \mathcal{N} \setminus (\mathcal{N}_S \cup \{o\}), v \in S(u) \tag{9}$$

where L_u, when $u = \langle a, r \rangle$, is the time for train a to run through r. Note that when $w_p = 1$ for some p containing both u and v, then (9) reduces to (2). Otherwise, when either or both route nodes are inactive, the constrain reduces to $t_v \geq t_u$ (where $v \in S(u)$).

Similarly, we get path dependent release-time constraints

$$t_u - t_o - \Gamma_u \geq M(z_u - 1) \qquad u \in \mathcal{N} \tag{10}$$

Again, one can show that this constraint is dominated by the following system of inequalities:

$$t_u - t_o \geq \Gamma_u z_u \qquad u \in \mathcal{N} \tag{11}$$

which does not involve the big-M constant. Indeed, if $z_u = 1$, then (11) reduces to (3). If $z_u = 0$, then (11) reduces to (1).

Figure 8 shows the route-node graph G_b for the station in Figure 7. We have omitted the arc weights, and have instead labeled each arc with the path variables on which it depends, in accordance with (8) (or (9)).

4.2. Path-Dependent Objective Functions

This subsection shows how to assess the cost of a schedule (and path selection) when trains can follow different routes in the station. There are two major considerations. First, the timing points may depend on the path. Second, some paths may be preferred to others, e.g., when the official timetable establishes the (preferred) stopping platform.

Path-dependent timing nodes Let $\mathcal{N}_T(a) \subseteq \mathcal{N}_T$ be the timing nodes for train a. Then we rewrite (4) as follows

$$\begin{array}{ll} \eta_u = max(0, t_u - T_u) & \text{if } z_u = 1 \\ \eta_u = 0 & \text{otherwise} \end{array} \qquad u \in \mathcal{N}_T(a), a \in \mathcal{A} \tag{12}$$

We let $k_a \geq 0$ be the cost of 1 unit of delay of train a.

Path-dependent costs To account for this cost component, we let $c_p \geq 0$ be the cost of choosing path $p \in P$.

Path-dependent objective function The overall path-dependent objective function can be written as:

$$\min \sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}(a)} c_p w_p + \sum_{a \in \mathcal{A}} \sum_{u \in \mathcal{N}_T(a)} k_a \eta_u \tag{13}$$

In order to express (12) using linear constraints, we can use the big-M trick. This results in the following family of inequalities:

$$\begin{array}{l} \eta_u \geq t_u - T_u - M(1 - z_u) \\ \eta_u \geq 0 \end{array} \qquad a \in \mathcal{A}, u \in \mathcal{N}_T(a) \tag{14}$$

If we have $z_u = 1$, then $\eta_u \geq t_u - T_u$ and $\eta_u \geq 0$ hold together, and then the positive coefficient k_a in objective function will push

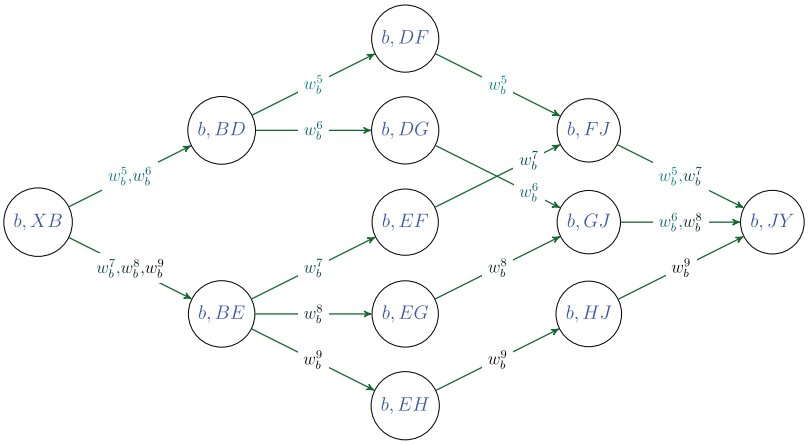


Fig. 8. The route-node graph G_b of train b. Nodes correspond to the interlocking routes available for b, and an arc (u, v) means that the route associated with v starts at the signal where the route associated with u ends. The variables used to label each arc (u, v) represent the paths containing both u and v. The route node $\langle b, XB\rangle$ is the origin or the route node for the interlocking route leading to signal B, while the route node $\langle b, JY\rangle$ is a sink node (if the train leaves the modelled area) or the route node for the interlocking route leading away from signal J. In the figure, we are assuming that neither $\langle b, XB\rangle$ nor $\langle b, JY\rangle$ are path dependent. Any path $p \in \mathcal{P}(a)$ corresponds to a directed path from $\langle b, XB\rangle$ to $\langle b, JY\rangle$ in G_b. However, not all paths from $\langle b, XB\rangle$ to $\langle b, JY\rangle$ in G_b need to belong to $\mathcal{P}(a)$.

the optimal value η_u^* down to $\max(0, t_u - T_u)$. When $z_u = 0$, only $\eta_u \geq 0$ holds (the other inequality becomes redundant), and we get $\eta_u^* = 0$.

Partitioning of path-dependent timing nodes We now consider the case where $\mathcal{N}_T(a)$, the timing nodes for train a, can be partitioned into $\{\mathcal{N}_T^1(a), \ldots, \mathcal{N}_T^{r^a}(a)\}$, such that

$$\sum_{u \in \mathcal{N}_T^i(a)} z_u = 1 \qquad T_u = T_v \qquad \text{for } u, v \in \mathcal{N}_T^i(a),\ i \in \{1, \ldots, r^a\} \tag{15}$$

That is, train a will use precisely one timing node in $\mathcal{N}_T^i(a)$, and all timing nodes in $\mathcal{N}_T^i(a)$ have the same time in the published timetable.

For passenger trains with multiple path options in a station, the partitioning condition (15) holds if arrival and departure times are independent of path selection. This is usually the case, and we can generally assume that the above partition exists. When it does, we introduce only one delay variable per set in the partition, namely $\eta_a^i, \ldots, \eta_a^{r^a}$.

Finally, using (15) and the assumptions behind (9), one can show that the constraints (14) can be replaced by the family of constraints

$$\eta_a^i \geq t_u - T_u \qquad \eta_a^i \geq 0 \qquad a \in \mathcal{A}, i = 1, \ldots, r^a, u \in \mathcal{N}_T^i(a) \tag{16}$$

which does not contain the big-M constant. We let k_a^i be the cost of delaying train a by one unit at the nodes in $\mathcal{N}_T^i(a)$ (e.g., the parallel timing points this represents), and get

$$\min \sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}(a)} c_p w_p + \sum_{a \in \mathcal{A}} \sum_{i=1}^{r^a} k_a^i \eta_a^i \tag{17}$$

4.3. Path-Dependent Selection Variables and Disjunctive Constraints

When we introduce alternative paths, potential conflicts become path-dependent, as they depend on whether certain tracks are used by certain trains.

More specifically, let $u = \langle a, r\rangle$ and $v = \langle b, s\rangle$ be two distinct route nodes, with $\{u, v\} \in \mathcal{K}$, where now $\mathcal{K}$ contains all pairs of route nodes in potential conflict, independently of whether or not the nodes are actually used by the trains. Then the potential conflict exists if and only if a path $p \in \rho(u)$ trough u for train a and a path $q \in \rho(v)$ through v for train b are selected, namely if node u and v are both active. In this case, we need to decide whether a precedes b or b precedes a in the contested track resource. The constraint (5) is extended as follows:

$$\begin{array}{l} t_v - t_{u_p^+} \geq \delta_u \\ \quad \text{or} \\ t_u - t_{v_q^+} \geq \delta_v \end{array} \qquad \text{if } z_u = z_v = 1 \qquad \{u, v\} \in \mathcal{K} \tag{18}$$

To linearize the above disjunctive constraint we introduce, for $\{u, v\} \in \mathcal{K}$, binary selection variables $y_{uv}, y_{vu} \in \{0, 1\}$ which, as in (5), decide which of the two terms of the disjunction must be satisfied by the schedule. In particular, if $y_{uv} = 1$, then u precedes v and the first term is the valid one; if $y_{vu} = 1$, then v precedes u and the second term is the valid one. Since (18) only holds if both u and v are active, we have for all $\{u, v\} \in \mathcal{K}$

$$y_{uv} \leq z_u, \quad y_{vu} \leq z_u, \quad y_{uv} \leq z_v, \quad y_{vu} \leq z_v. \tag{19}$$

In any case, for all $\{u, v\} \in \mathcal{K}$ at most one selection variable can be one:

$$y_{uv} + y_{vu} \leq 1 \tag{20}$$

Finally, for all $\{u, v\} \in \mathcal{K}$, when both u and v are selected, one selection variable must be one:

$$y_{uv} + y_{vu} \geq z_u + z_v - 1. \tag{21}$$

Note that if $z_u = 0$ or $z_v = 0$, (21) is redundant as the y variables are non-negative.

We are now ready to write the linear version of constraint (18), by exploiting once again the big-M trick:

$$\begin{array}{lll} (i) & t_v - t_{u_p^+} - \delta_{u_p^+} \geq M(y_{uv} + w_p - 2) & p \in \rho(u) \\ (ii) & t_u - t_{v_q^+} - \delta_{v_q^+} \geq M(y_{vu} + w_q - 2) & q \in \rho(v) \end{array} \qquad \{u, v\} \in \mathcal{K} \tag{22}$$

Let $u = \langle a, r\rangle$ and $v = \langle b, s\rangle$, and suppose $w_p = y_{uv} = 1$. It follows from (19) that, since $y_{uv} = 1$, both u and v are selected and train a precedes b. $w_p = 1$ implies that path p is selected for train a and, therefore, u_p^+ is the node following u for a. Since $w_p = y_{uv} = 1$, the r.h.s. of (22.i) is 0 and the constraint is active. On the other hand, since $y_{uv} = 1$, then $y_{vu} = 0$, and, with M suitably large, (22.ii) becomes redundant. A similar argument applies with the first and second term in (22) interchanged, when $y_{uv} = 0$ and $y_{vu} = w_q = 1$.

These path-dependent disjunctive constraints can be represented and visualized by the *path-dependent disjunctive graph* of in Figure 9, associated with two trains a, b. This graph contains as subgraphs the route-node graphs G_a, G_b of trains a and b, respectively, plus the origin, which is connected to the entry node of each train in the corresponding route-node graph. The green

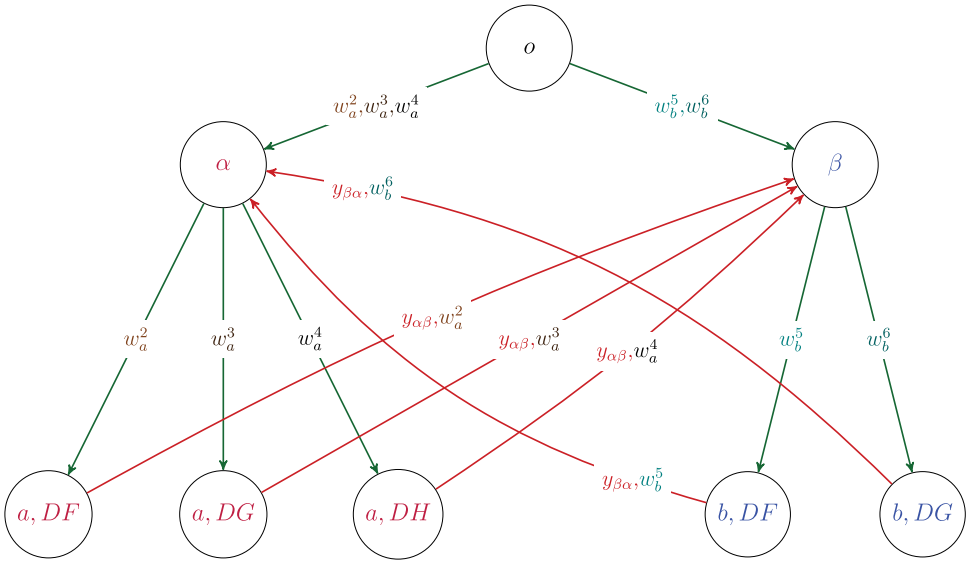


Fig. 9. A path dependent disjunctive graph. Each red arc represents one of the five terms in a disjunctive constraint (22). One such arc is thus associated with a selection variable y and a path variable w. The term becomes active, i.e. the arc is chosen, when both the associated y and w variables are 1.

Table 1
Listing of notation.

Symbol	Description
$\mathcal{A}$	Set of all trains
$\mathcal{N}$	Set of all route nodes
$\mathcal{N}(a)$	Set of all route nodes for train a
$\mathcal{N}_T$	Set of all timing nodes
$\mathcal{N}_T(a)$	Set of all timing nodes for train a
$\mathcal{N}_T^i(a)$	Element i of partition of timing nodes for train a
$\mathcal{N}_S$	Set of all sink nodes
o	Origin route node
u_p^+	Route node following u on path p
$\mathcal{P}$	Set of all paths
$\mathcal{P}(a)$	Set of all paths for train a
$\rho(u)$	Set of all paths containing route node u
$\mathcal{K}$	Set of all (potential) conflicts
$\mathbf{T}, T_u$	Timetable
$\mathbf{t}, t_u$	Schedule (variable)
η_u, η_a^i	Delay variable
y_{uv}	(precedence) selection variable
w_p	(path) selection variable
z_u	(route node) selection variable
k_a, k_a^i	Cost of unit delay to train a (in i)
c_p	Cost of selecting path p
$L_{\langle a,r\rangle}$	Time for train a to pass through interlocking route r
$\delta_{\langle a,r\rangle}$	Time for the length of train a to pass the signal at the end of r

arcs belong to the train-specific route-node graphs, and they can be associated with the path variables as in Figure 8. Each arc with one endpoint in one route-node graph, and the other endpoint in the other route-node graph, is associated with one of the terms in a disjunction. A term becomes active if both the corresponding y variable and w variable are 1. Note that the graph may contain parallel arcs. To simplify notation, we let $\alpha = \langle a, AD\rangle$ and $\beta = \langle b, BD\rangle$.

In Figure 9 the red arcs correspond to the following set of inequalities (which are an instance of (22)):

$$
\begin{aligned}
&t_\beta \geq t_{\langle a,DF\rangle} + \delta_{\langle a,DF\rangle} && \text{if} && y_{\alpha\beta} = w_a^2 = 1\\
&t_\beta \geq t_{\langle a,DG\rangle} + \delta_{\langle a,DG\rangle} && \text{if} && y_{\alpha\beta} = w_a^3 = 1\\
&t_\beta \geq t_{\langle a,DH\rangle} + \delta_{\langle a,DH\rangle} && \text{if} && y_{\alpha\beta} = w_a^4 = 1\\
&t_\alpha \geq t_{\langle b,DF\rangle} + \delta_{\langle b,DF\rangle} && \text{if} && y_{\beta\alpha} = w_b^5 = 1\\
&t_\alpha \geq t_{\langle b,DG\rangle} + \delta_{\langle b,DG\rangle} && \text{if} && y_{\beta\alpha} = w_b^6 = 1
\end{aligned}
$$

5. Solution Approach

Using the linearized constraints (8), (10), (14), and (22), we get the following MILP formulation for ODP. A listing of notation can be found in Table 1.

$$
\begin{aligned}
\textbf{min}\quad & \sum_{a\in\mathcal{A}}\sum_{p\in\mathcal{P}(a)} c_p w_p + \sum_{a\in\mathcal{A}}\sum_{u\in\mathcal{N}_T(a)} k_a \eta_u && && \\
\textbf{s.t.}\quad & (ia) && \sum_{p\in\mathcal{P}(a)} w_p = 1 && a\in\mathcal{A}\\
\textbf{s.t.}\quad & (ib) && \sum_{p\in\rho(u)} w_p = z_u && u\in\mathcal{N}\setminus\{o\}\\
& (iia) && y_{uv} + y_{vu} \geq \sum_{p\in\rho(u)} w_p + \sum_{q\in\rho(v)} w_q - 1 && \{u,v\}\in\mathcal{K}\\
& (iib) && y_{uv} + y_{vu} \leq 1 && \{u,v\}\in\mathcal{K}\\
& (iic) && y_{uv} \leq z_u && \{u,v\}\in\mathcal{K}\\
& (iid) && y_{uv} \leq z_v && \{u,v\}\in\mathcal{K}\\
& (iii) && t_u - t_o - \Gamma_u \geq M(z_u - 1) && u\in\mathcal{N}\\
& (iv) && t_{u_p^+} - t_u - L_u \geq M(w_p - 1) && u\in\mathcal{N}\setminus\mathcal{N}_S,\ p\in\rho(u)\\
& (va) && t_v - t_{u_p^+} - \delta_u \geq M(y_{uv} + w_p - 2) && \{u,v\}\in\mathcal{K},\ p\in\rho(u)\\
& (vb) && t_u - t_{v_p^+} - \delta_v \geq M(y_{vu} + w_q - 2) && \{u,v\}\in\mathcal{K},\ q\in\rho(v)\\
& (vi) && \eta_u - t_u + T_u \geq M(z_u - 1) && a\in\mathcal{A},\ u\in\mathcal{N}(a)\\
& && t_u \geq 0 && u\in\mathcal{N}\\
& && \eta_u \geq 0 && a\in\mathcal{A},\ u\in\mathcal{N}_T(a)\\
& && y_{uv}\in\{0,1\} && \{u,v\}\in\mathcal{K}\\
& && w_p\in\{0,1\} && p\in\mathcal{P}\\
& && z_u\in\{0,1\} && u\in\mathcal{N}\setminus\{o\}
\end{aligned}
\tag{23}
$$

In Section 4, we discussed how we can strengthen (23.iii), (23.iv), and (23.vi), replacing them with (9), (11), and (16), respectively.

5.1. Delayed Variable and Constraint Generation

Already in a moderately sized instance of ODP, the number $|\mathcal{K}|$ of potential scheduling conflicts can grow prohibitively large. Therefore, rather than generating a full instance with all constraints (23.v) from the start, we prefer to solve a sequence of

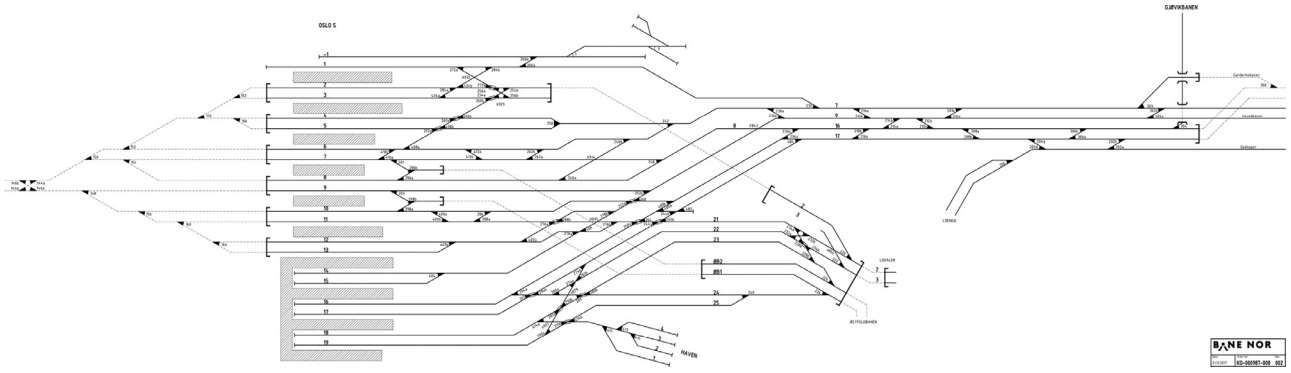


Fig. 10. Schematic track layout of Oslo Central Station. The station is laid out approximately east-to-west, so this schematic is oriented north-up. **Source:** Bane NOR (CC BY-SA 4.0)

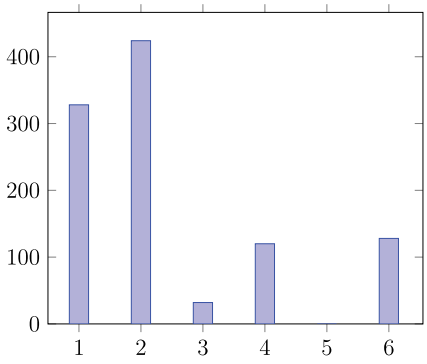


Fig. 11. Distribution of numbers of paths for the trains.

smaller instances by applying the delayed constraint generation approach. The idea is to start solving to optimality a model with much fewer constraints. We then check if any of the missing constraints are violated by the current optimal solution. If this is not the case, then the solution can be shown to be optimal for the full problem (with all constraints). Otherwise, the violated constraints are added to the model, and the process is iterated.

We start with a model $\mathcal{M}^0$, which is (23) with all constraints of type (23.ii) and (23.v) removed, and without any y-variables. Then, we use the algorithm outlined as follows.

1. Set $i \leftarrow 0$.
2. Find the optimal solution t^i, w^i, y^i, η^i to $\mathcal{M}^i$.

1. if there exist a potential conflict pair $\{u, v\} \in \mathcal{K}$ such that paths $p \in \rho(u)$, $q \in \rho(v)$ are chosen (i.e. $w^i_p = w^i_q = 1$) and constraint (18) is violated, create $\mathcal{M}^{i+1}$ by adding to the model the associated y-variables and constraints (19), (20), (21) and (22), and update $i \leftarrow i + 1$. Go to 2.
2. else the solution is optimal for (23).

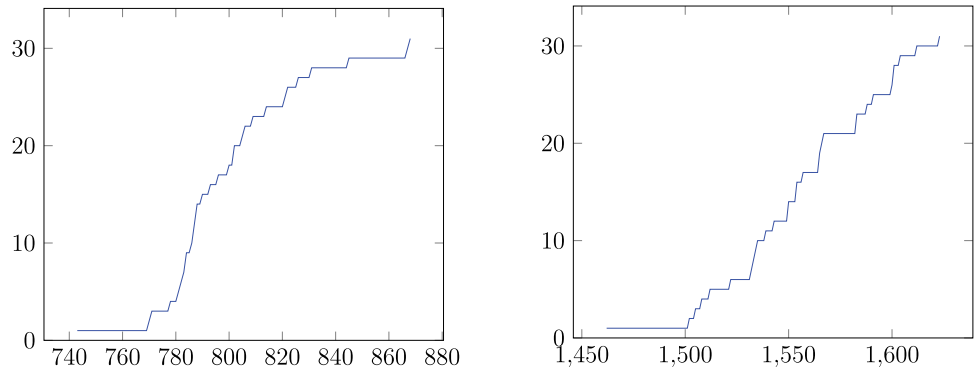
Checking for violated inequalities can be done very efficiently. In Appendix A we give more details on our constraint generation (conflict detection) algorithm. In our experiments, we have chosen to add variables and constraints associated with all potential conflicts where (18) is violated in the solution to $\mathcal{M}^i$.

6. Computational Results

To test our approach, we find the optimal dispatching solution in a variety of traffic instances in Oslo Central Station. The purpose of our experiments is to verify that our model can be used in real-time applications. Since we still lack detailed timing information except for at platform signals, we craft a set of instances that are at least as hard as the instances solved in normal operation. In an ongoing field-testing campaign in the GOTO project, dispatchers will judge the quality of the solutions produced by this algorithm. Figure 13 shows a screenshot of the prototype currently available to the dispatchers at Oslo Central Station. In this paper, we have focused on testing the limits of our algorithm.

Figure 10 show the layout of Oslo Central Station. The station has 19 platform tracks, 1 west-bound line, 4 east-bound lines, and 2 east-bound exits to technical areas. Oslo Central Station has 19 tracks. Track 1 and tracks 14–19 are east-bound only, while tracks 2–13 are both east-bound and west-bound. Tracks 2–8 are primarily used for west-bound traffic, while tracks 9–13 are primarily used for east-bound traffic. To the west, all tracks collect into a west-bound double-track tunnel. This tunnel is the main line connecting the east and the west of Norway by rail and is very busy. To the east, traffic can go north-east on Brynsbakken to one of the three lines Gjvikbanen, Hovedbanen, and Romeriksporten (Gardermobanen); southeast to stfoldbanen or the depot at Lodalen; or south to the yard at Haven. To the east, tunnels allow traffic to move at different levels in order to improve traffic flow. The tracks are divided into hundreds of track circuits, allowing effective use of sectional release (see Appendix A). In order to construct our experiments, we have been given insight into restricted-access documents detailing the infrastructure of Oslo Central Station and the surrounding area. In particular, we have been able to use the real track-circuit and signal layout, though we have had to estimate the exact sizes of each track circuit. In order to generate the timetable, we have used the public listing of arrivals and departures at the station. With permission from Bane NOR, we have published the infrastructure data for Oslo Central Station in a companion paper [24]. Our infrastructure model for the area has 198 track circuits, 254 interlocking routes, and 94 paths. Almost all potential conflicts are between trains on the same path, trains on merging paths and trains on crossing paths. There is one short single track segment with conflicts between meeting trains, and a few areas where some paths going in opposite directions can share a stretch of tracks, but these cases are relatively few.

All our instances are based on repeating the traffic scheduled between 4 p.m. and 5 p.m. on a weekday, during the height of the after-work commute out of Oslo city center. We test our approach under an extra heavy load by repeating this very busy hour instead of following the published timetable. The number of trains scheduled to move through the station this hour is 55, with some trains arriving from or departing to the neighboring depot. For traffic outside of rush hour (for longer simulations), we have selected 37 of the rush-hour trains. These are the trains scheduled to run hourly at off-peak hours, along with some of the train that have departures less frequently than each hour at off-peak times. The resulting number of trains is representative for off-peak hours during the day, and too high for off-peak hours during the night.



(a) Results in Table 3 (b) Results in Table 4

Fig. 12. Cumulative distribution of number of selection constraints for the experiments in Table.

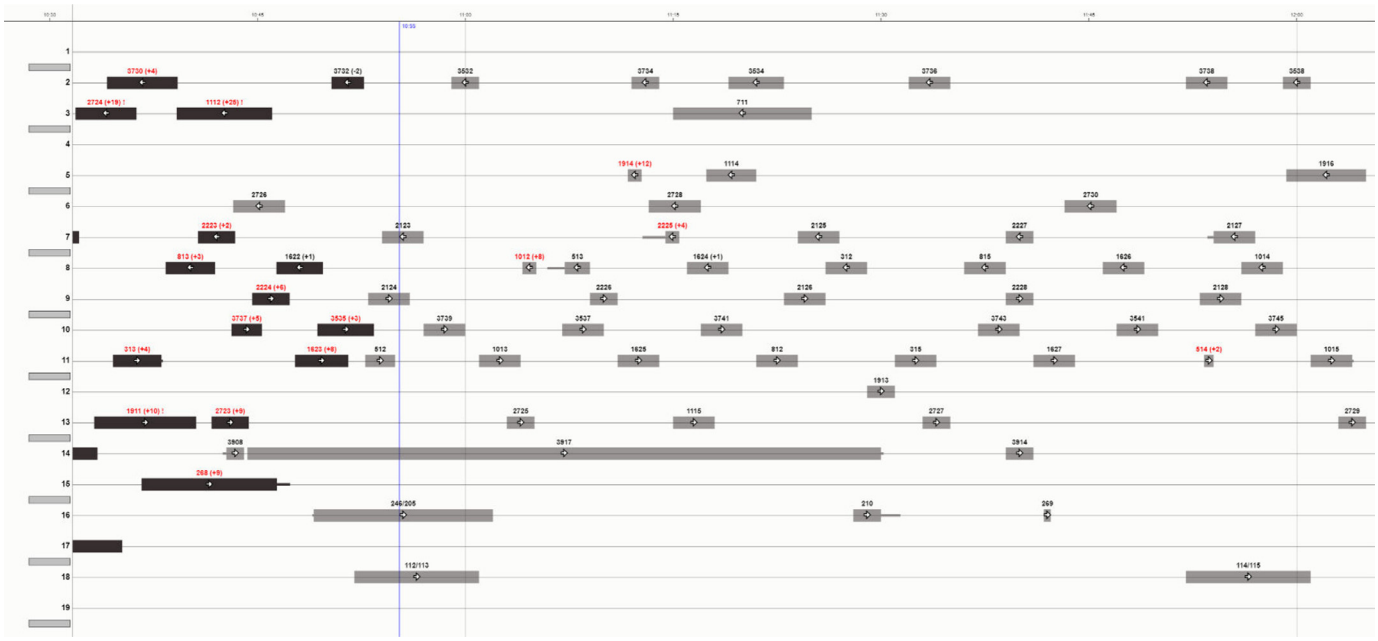


Fig. 13. Oslo Central Station dispatcher support tool prototype from the GOTO project. When working with a single station, the dispatching horizon can be as short as 2 hours.

Whenever a track connection existed, the trains were allowed to use both their scheduled track and the other track on the same platform. As possible paths, we allowed the paths trains are observed to take to their designated track and paths suggested as alternatives by dispatchers at Bane NOR. In order to minimize symmetry-breaking, we set a constant cost of delay $k_a = k_a^i = 1$, and give all paths cost $c_p = 0$. This is also in close accordance with practice at the station, since we only allow changing track to an adjacent track on the same passenger platform.

We ran our experiments on a MacBook Pro (15-inch, 2016) with a 2.9 GHz Quad-Core Intel Core i7 processor and 16 GB 2133 MHz LPDDR3 memory, running macOS 10.15, Python 3.8.2, and Gurobi 9.0.3 with default settings.

In order to compare the different formulations we present in Section 5, we have generated a set of 200 instances, each consisting of 4 consecutive hours of rush-hour traffic. In each instance, we randomly select trains and give them a random delay between 1 and 15 minutes. The number of trains selected for delay is between 1 and 30. We then solve all 200 instances using each of the formulations we compare.

The *big-M* formulation is the one given in (23). In rows labeled *modified*, the corresponding constraints have been replaced by their strengthened versions, described immediately following (23). In rows labeled *combined*, the corresponding constraints have been supplemented by their strengthened versions. Table 2 shows the average performance statistics. The columns of the table are (t_{total}) the total processing time, (t_{last}) the processing time of the last MIP to be solved, (t_{sep}) the time spent separating conflict constraints, (MIP) the number of MIP models solved, (Nodes) the total number of MIP nodes processed, (Vars) the number of selection variables generated, and (Constr) the number of selection constraints generated.

Table 2 shows a surprising result: replacing (23.iii) and (23.iv) with stronger, non-big-*M* versions slows down the solution process. Comparing the first three rows of Table 2 with the middle three rows, we see that the number of MIPs solved is smaller when using the strengthened constraints. This indicates that the intermediate solutions, and therefore the conflict detection process, might be better. However, the number of MIP nodes processed is much larger, indicating that the branch-and-bound process is less effec-

Table 2
Comparison between different MILP formulations for ODP using different versions of the constraints (23.iii), (23.iv), and (23.vi). For comparison, a set of 200 instances (220 trains over 4 hours with random delays) have been solved using each formulation, and the values shown in this table are the averages for each formulation. The fastest formulation has been highlighted.

(23.iii),(23.iv)	(23.vi)	t_{total}	t_{last}	t_{sep}	MIP	Nodes	Vars	Constr
big-*M*	big-*M*	13.63	1.00	0.25	17.07	21778.87	1035.29	517.64
big-*M*	modified	16.47	0.66	0.36	25.45	26405.58	1300.69	650.35
big-*M*	combined	15.11	0.63	0.34	25.43	22796.97	1273.94	636.97
modified	big-*M*	14.83	1.18	0.25	16.88	29430.74	1001.78	500.89
modified	modified	29.30	1.83	0.31	21.12	49842.10	1147.05	573.52
modified	combined	24.17	1.53	0.27	19.64	40426.82	1125.34	562.67
combined	**big-*M***	**12.73**	**0.94**	**0.24**	**16.66**	**20313.77**	**1036.53**	**518.26**
combined	modified	23.07	1.57	0.28	18.55	34165.39	1136.41	568.21
combined	combined	16.98	1.22	0.26	17.59	25044.31	1093.82	546.91

Table 3
Computational results for Oslo Central Station. Each instance is 6 hours of rush-hour traffic, with a total of 330 trains. N_d is the number of trains that has been delayed to create the random instance, and each delayed train was delayed by a random number of minutes between 1 and 15.

N_d	t_{total}	t_{last}	t_{sep}	MIP	Nodes	Vars	Constr
0	31.75	2.19	0.39	19	27842	1576	788
1	44.83	1.89	0.48	24	39388	1628	814
2	15.46	1.52	0.27	13	18871	1486	743
3	41.33	1.41	0.50	24	27607	1662	831
4	28.17	1.01	0.34	16	43545	1586	793
5	32.61	2.65	0.39	20	27394	1572	786
6	29.99	2.77	0.36	17	31249	1562	781
7	56.48	2.18	0.62	30	41290	1736	868
8	41.42	2.23	0.47	23	29717	1652	826
9	27.63	1.77	0.38	18	44345	1566	783
10	47.22	1.99	0.49	24	39800	1690	845
11	17.89	1.17	0.29	14	13660	1556	778
12	42.11	1.77	0.49	24	32576	1600	800
13	22.29	1.87	0.33	16	26874	1542	771
14	34.46	2.48	0.40	20	27559	1612	806
15	36.59	1.91	0.47	22	24433	1734	867
16	27.39	1.64	0.38	18	35316	1568	784
17	27.00	2.01	0.38	18	21565	1564	782
18	31.92	2.87	0.40	19	28043	1618	809
19	21.70	1.80	0.35	16	13584	1540	770
20	31.87	2.14	0.39	18	29370	1580	790
21	25.43	2.68	0.35	16	36534	1574	787
22	18.38	2.18	0.28	14	11386	1574	787
23	23.60	2.59	0.36	17	28187	1610	805
24	28.07	2.19	0.34	16	19580	1642	821
25	29.39	2.43	0.44	21	34381	1604	802
26	32.68	1.43	0.40	19	21195	1592	796
27	48.98	2.42	0.48	25	45070	1644	822
28	18.66	1.71	0.28	13	18206	1604	802
29	27.29	1.97	0.38	18	36655	1576	788
30	24.44	2.06	0.33	16	17919	1568	784

Table 4
Computational results for Oslo Central Station. Each instance is 24 hours of traffic, with two rush-hour periods of 4 hours each and a total of 1032 trains. N_d is the number of trains that has been delayed to create the random instance, and each delayed train was delayed by a random number of minutes between 1 and 15.

N_d	t_{total}	t_{last}	t_{sep}	MIP	Nodes	Vars	Constr
0	43.45	3.02	1.64	19	19721	3064	1532
1	47.47	2.99	1.81	22	34519	3016	1508
2	89.21	3.48	2.69	32	51796	3202	1601
3	63.15	2.34	2.08	25	31459	3166	1583
4	53.14	2.91	1.69	21	53566	3208	1604
5	54.38	3.16	1.80	21	32423	3200	1600
6	36.19	2.88	1.31	17	34178	3068	1534
7	57.66	3.71	1.89	24	28443	3176	1588
8	23.80	2.52	0.95	12	22428	3010	1505
9	36.60	2.73	1.32	16	23108	3100	1550
10	54.25	2.71	1.70	21	39828	3108	1554
11	91.07	2.48	2.75	34	54384	3246	1623
12	53.08	2.64	1.77	22	59062	3070	1535
13	35.81	3.29	1.36	17	26755	3086	1543
14	36.29	3.22	1.27	16	20789	3044	1522
15	55.57	3.87	1.84	23	32907	3114	1557
16	76.93	3.95	2.04	25	34058	3202	1601
17	50.44	2.75	1.62	20	61812	3066	1533
18	31.90	2.71	1.30	16	30493	3004	1502
19	67.57	3.44	2.09	26	35096	3108	1554
20	26.92	2.54	1.13	14	26348	2924	1462
21	57.52	3.48	1.75	21	26297	3224	1612
22	65.72	3.19	1.95	25	46235	3130	1565
23	61.08	3.70	2.08	26	54550	3182	1591
24	56.80	3.05	1.74	22	18673	3166	1583
25	33.78	2.33	1.30	17	18268	3100	1550
26	71.83	3.03	2.35	30	40423	3078	1539
27	47.63	2.66	1.73	21	46208	3024	1512
28	37.13	3.35	1.38	17	32929	3130	1565
29	84.11	1.86	2.46	32	49202	3132	1566
30	51.30	3.97	1.80	21	29310	3134	1567

tive. When we replace (23.vi), we get even slower running times. In this case, strengthening the inequality removes the explicit connection between the path variables and the delay variables, which could explain the increased number of MIP nodes processed. The highlighted row of Table 2 shows the formulation with the best performance in our comparison experiments. In this formulation we have supplemented (23.iii) and (23.iv) with their strengthened version, and kept (23.vi). This results in the number of MIP solves and the number of MIP nodes processed being comparatively low. We use this formulation in the rest of our experiments.

Table 3 shows performance results from an experiment very similar to the comparison experiment, but with six consecutive hours of traffic. The new column (N_d) is the number of trains delayed to create the instance. Each instance has 330 trains moving through the station. Most trains have options for which path to take to their designated platform, and some trains have options for which platform to take. In each instance, the total number of path selection variables is 798. All the instances are solved in under 1 minute. This is fast enough that dispatchers can use the suggested solutions; the traffic pattern does not change too much in 1 minute. Furthermore, once a train is approaching the station, the signals will have been set, and changing the train's schedule will be very work-intensive. Therefore, there is no great need for a faster refresh rate than every minute; dispatchers need time to implement the planned schedule.

Table 4 shows the performance results from an experiment where each instance is made up of 24 hours of traffic: two 4-hour periods of rush-hour traffic and three periods of reduced traffic. Each instance has 1032 trains moving through the station, and the total number of path selection variables is 2520. All instances are solved within 100 seconds. Similar to the above situation, this is fast enough to be useful in real-time applications when dispatchers and schedulers plan this far ahead.

We note that for the experiments reported in Table 4, the number of potential conflicts $|\mathcal{K}|$ were on the order of 10^6 or higher, yet the number of generated selection constraints were only about 1500 in each instance, as shown in Figure 12.

In Table, it appears that the difficulty of an instance is not strongly correlated to the number of delays. There are two reasons main for this. First, the timetable does not provide schedule information other than at the platform, so we have extrapolated to the full station area. As a result, the initial schedule is not necessarily conflict free. Second, delays could occasionally shift trains away from a congested times.

7. Conclusions and Future Work

The computational results reported in the previous section show that our approach can solve the ODP in a large passenger train station up to a 24-hour planning horizon in a reasonable amount of time. As a part of the innovation project GOTO with the Norwegian infrastructure manager Bane NOR, a prototype based on our approach is currently undergoing a field-test campaign. The dispatchers at Oslo Central Station control center will be involved in the testing. As part of this test campaign, we are planning to ask dispatchers to solve small but complex instances of the problems presented here, so that we can compare our algorithm-based solutions with their expert solutions.

Even though the experiments on real-life data show that the approach will work in practice, the field campaign may present new challenges. Also, further research may be needed to tackle even larger stations with more congested traffic, which exist in other European railways.

In the GOTO project, we also developed a prototype, already under testing at Oslo Central Station, to dispatch trains on the lines incident to the station. The final objective of the GOTO project is to develop a decomposition approach to combine the two prototypes to control the traffic over the entire Greater Oslo region. The final prototype will be tested at Oslo Central Station in June 2022.

Based on discussions with dispatchers at Oslo Central Station, we have decided to rely on predetermined paths for the routing within the station. On the one hand, this approach aligns well with how dispatchers route trains in the station, making it easy for dispatchers to implement suggested solutions. On the other, using predetermined paths restricts the search space. In order to further improve our approach, we want to explore more options for path generation. First, we want to let dispatchers control which paths are available for each train, including drawing new paths. Second, we want to design an algorithm that generates paths on-demand based on the current traffic in the station.

CRediT authorship contribution statement

Carlo Mannino: Conceptualization, Methodology, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Andreas Nakkerud:** Conceptualization, Methodology, Software, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization.

Data availability

Data will be made available on request.

Appendix A. Detailed Conflict Modelling

In order to model Oslo Central Station (or any other large passenger station) with sufficient accuracy, it is necessary to understand the low-level business rules of operating the rail infrastructure that makes up the station. In this appendix, we explain the details behind our conflict detection approach. These are the details that allow us to achieve feasible schedules without adding safety margins.

A1. Track Infrastructure

Up to this point, we have considered the track infrastructure to be divided into interlocking routes with potential conflicts between interlocking routes that share some physical resources. While the interlocking routes are the finest division from a scheduling point of view, these routes are further divided into *track circuits* for actual conflict detection. Figure A.14 shows an example of how tracks can be divided into track circuits.

Track circuits can be seen as atomic track elements since they partition the track infrastructure; no two track circuits may share any piece of track. Track circuits are equipped with detectors that can tell if a train occupies the track circuit. For the purpose of conflict detection, we regard an interlocking route as a sequence of track circuits $r = (c_1, \ldots, c_n)$. A pair of routes are in potential conflict when they share at least one track circuit.

In our discussion so far, if one train must wait for another, the preceding train must enter the following route in its path before the waiting train may proceed (Section 3.4). By looking at the underlying track circuits of an interlocking route, we may introduce alternative, less restrictive variants of the precedence constraints.

When a train enters an interlocking route, some of the track circuits following the upcoming signal are temporarily reserved as an added layer of protection against collisions. Thus, two interlocking routes can be in potential conflict even if they do not share a track circuit directly.

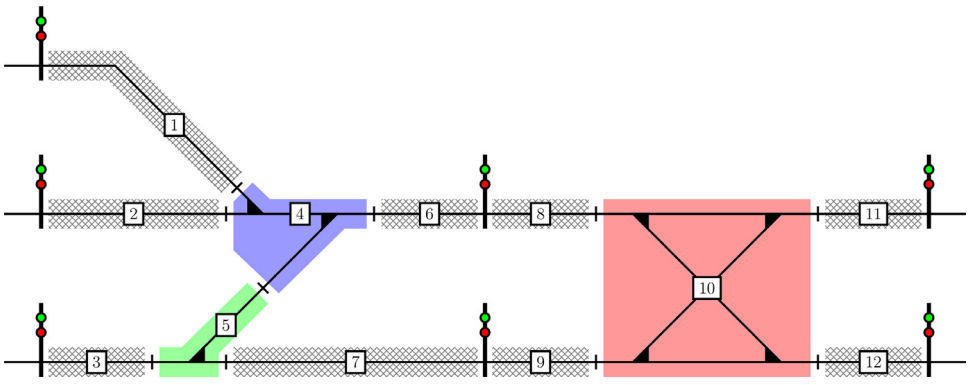


Fig. A.14. The figure shows an area of tracks divided into 12 track circuits, shown as shaded or crosshatched areas. The track circuits are delimited by the signals and the small hatches drawn on the lines. The large, shaded track circuit on the right prevents the two parallel tracks going through it from being used simultaneously.

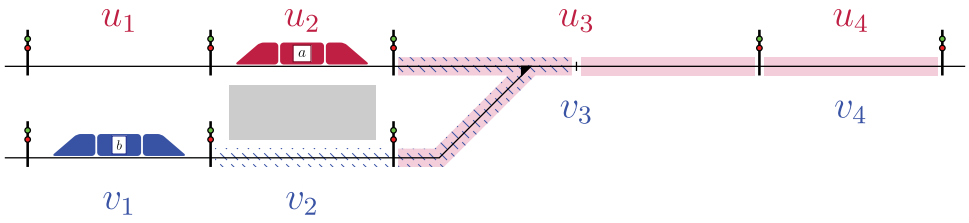


Fig. A.15. If the blue train wants to enter v_2, it must reserve the blue, hatched track circuits, including the first track circuit of v_3 as a safety zone. This would prevent the purple train from entering u_3 until the safety zone is released. Conversely, if the purple train wants to enter u_3, it must reserve the purple, shaded track circuits, including u_4 as a safety zone, this would prevent the blue train from entering v_2 until the first track circuit of u_3 is cleared by the purple train and released.

A2. Track Circuit Reservations

When a train enters an interlocking route, it must reserve all track circuits that are part of that route and possibly some track circuits forming a *safety zone* after the signal at the end of the route.

Any pair of trains reserving the same track circuit represent a potential scheduling conflict. In order to check if a candidate schedule realizes a potential conflict, we create an interval graph for each track circuit, where each interval represents a reservation of the track circuit. If any pair of intervals overlap, there is a realized scheduling conflict, and the candidate schedule is not feasible.

Track circuits reservations end and we say the track circuits are *released*, either after they have been passed by the train, when the train exits the containing interlocking route, or, in the case of safety zones, after a fixed amount of time.

In the algorithm laid out in Section 5, conflict detection is performed for each intermediate solution. In this case, the current solution is used to generate reservations.

A2.1. Safety Zones

Signals offer the primary protection of trains moving in interlocking routes. No collision is ever possible as long as no train ever passes a signal at danger (red light). *Safety zones* (or interlocking route *overlaps* [2]) offer additional protection against collision if a train should pass a signal at danger accidentally.

Figure A.15 shows an example of a safety zone. Train b (blue) is about to enter route v_2, and train a (purple) is about to enter u_3. In order to protect against collision, train b must reserve not only v_2, but also the first track circuit of v_3. If b were to pass the signal protecting v_3 (and u_3), an automatic emergency break could stop b in the safety zone. Were it not for this safety zone, train a could occupy the points in u_3 while b enters v_2. Should b pass the following signal, it could crash into a before an emergency break could bring b to a stop.

In order to prevent a collision, a safety zone must contain enough track circuits to cover the minimum braking distance of the train approaching the corresponding signal. Unlike the track circuits making up an interlocking route, the track circuits of a safety zone may be released before they are used by the reserving train. Consider the example in Figure A.15 again. When b enters v_2, it is going to stop at the adjacent platform. Then, a will leave the platform and enter u_3. The blue, crosshatched track circuits are reserved by b, and the purple-shaded track circuits are the ones a must reserve. A certain amount of time after b enters v_2 we can say for certain that b must have slowed. Otherwise, it would have reached the following signal. At this time, b can release the track circuits making up its safety zone in v_3, and a can reserve the track circuits on its path. Track circuits in a safety zone are reserved for a fixed amount of time.

A2.2. Route Release and Sectional Release

When a track circuit is part of an interlocking route, it is released either with the route or once the reserving train has passed through the track circuit itself.

We use Figure A.15 to illustrate the difference between route and sectional release. If we let train a take precedence over b, then b cannot enter v_2 before a releases the first track circuit of u_3, since this track circuit is the safety zone required by b when entering v_2. Under route release, a must fully enter u_4 before the contested track circuit is released, and b may proceed. Under sectional release, the track circuit can be released once a has passed fully into the last track circuit of u_3.

Under sectional release, train b may proceed earlier than under route release. Even if a must stop at the signal protecting u_4, b may proceed into v_2. In general, sectional release allows closer scheduling than route release.

A3. Conflict Types

We let $u = \langle a, r\rangle$ and $v = \langle b, s\rangle$ be route nodes. There is a potential scheduling conflict between u and v when they have a track circuit in common, either as part of the route itself or a part of a safety zone. The associated precedence constraints depend on the type of reservations made by the route nodes. A pair of routes may cause multiple potential scheduling conflicts. We add constraints to make sure none of the potential conflicts are realized. The form of these constraints depends on what the yielding train is waiting for. Table A.5 and Figure A.16 show the different types of potential scheduling conflicts.

Waiting for Route Release When one train is waiting for a preceding train to fully clear its route, the waiting train is waiting for a *route release*. This is, for example, the case when both trains are going to the same signal. In Figure A.15, if a precedes b under route release, then b cannot enter v_2 before a has fully left u_3 (and fully entered u_4). We get the constraint

$$t_{v_2} - t_{u_4} \geq \delta_{u_4} \qquad \text{if} \qquad y_{u_3 v_2} = 1 \tag{A.1}$$

where we note that the precedence decision is between u_3 and v_2, while it is u_4 that is used in the constraint. The arcs (u_4, v_3) and (v_4, u_3) in Figure A.16 both represent route release constraints. Unlike the other arc pairs, this pair is not anti-parallel. This is because the following train must wait for the leading train to exit the shared route and enter the next route, rather than wait for it to enter the shared route.

Waiting for Sectional Release If the route of the waiting train is crossing the route of the preceding train sufficiently far from the end of that route, then *sectional release* may apply. In sectional release, individual track circuits are released once they are passed. Since, when we reschedule, we may assume trains will only stop at signals, we can compute when a train passes a track circuit based on when it enters the route containing that track circuit. This will work as long as the track circuit is not part of the area where

Table A.5
When operating with safety zones, there are three types of conflicts between trains. First, two trains may want to use the same track circuit(s) as a safety zone. Second, a train may want to enter the safety zone of another train. And third, two trains may want to enter the same track circuit(s).

Purple	Blue	Variables	Description
Enter u_2	Enter v_2	$y_{u_2v_2}, y_{v_2u_2}$	Conflicting safety zones
Enter u_3	Enter v_2	$y_{u_3v_2}, y_{v_2u_3}$	Purple train enters safety zone of blue train
Enter u_2	Enter v_3	$y_{u_2v_3}, y_{v_3u_2}$	Blue train enters safety zone of purple train
Enter u_3	Enter v_3	$y_{u_3v_3}, y_{v_3u_3}$	Both trains enter the same track circuit(s)

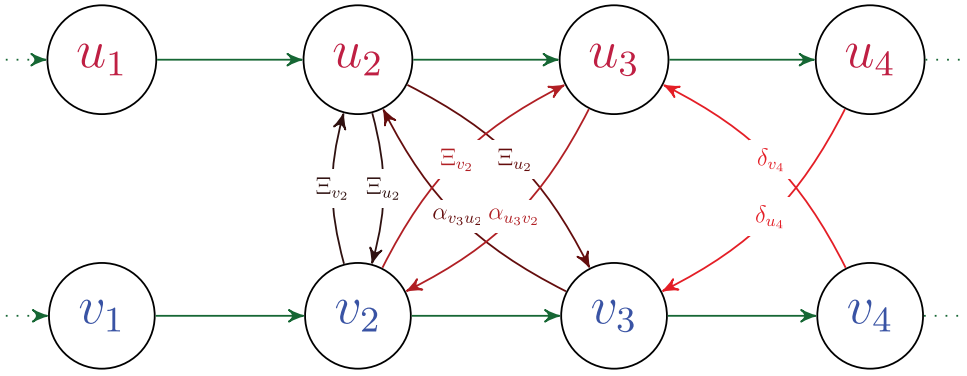


Fig. A.16. Four disjunctive precedence constraints of three different types. The physical situation is illustrated in Figure A.15, and the conflicts are described in Table A.5.

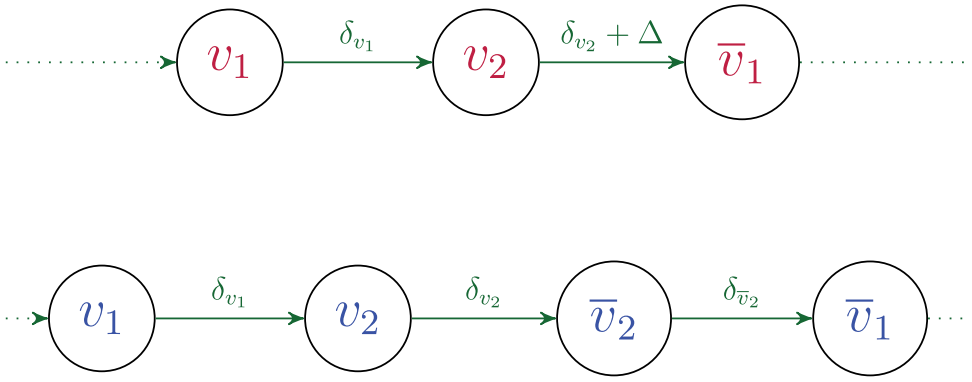


Fig. A.1. Options for modelling train reversal.

the train may stop before a signal. If train a precedes train b in Figure A.15, it is natural to apply sectional release.

We let α_{uv} be the time it takes for the preceding train (route node u) to get past the end of the last track circuit on the route of the waiting train (route node v). In the example in the figure we then get the constraint

$$t_{v_2} - t_{u_3} \geq \alpha_{u_3v_2} \quad \text{if} \quad y_{u_3v_2} = 1 \tag{A.2}$$

The arcs (u_3, v_2) and (v_3, u_2) in Figure A.16 both represent sectional release constraints.

The sectional release constraint (A.2) assumes that once the preceding train reaches the signal at the end of its current route, it has already released all the track circuits shared with the following train. If this is not the case, sectional release is not appropriate for this pair of trains in this order. This requirement for sectional release must either be checked for the model as a whole or for each individual conflict. In our case, we can ensure it holds generally, since we only schedule relatively short passenger trains.

Waiting for Safety Zone Release Waiting for a safety zone to be released is much like waiting for a sectional release. In Figure A.15, if v_2 precedes u_3, then train a only has to wait a fixed time after b enters v_2, instead of waiting for b to enter v_3 (which in this case would cause a different potential conflict with a). We let Ξ_u be the amount of time the safety zone corresponding to u must be reserved. Then,

$$t_{v_2} - t_{u_3} \geq \Xi_{u_3} \quad \text{if} \quad y_{u_3v_2} = 1 \tag{A.3}$$

The arcs (u_2, v_2), (v_2, u_2), (u_2, v_3), and (v_2, u_3) in Figure A.16 all represent waiting for a safety zone to be released.

Appendix B. Train Reversal and Shunting Operations

Our model deals with train reversals and shunting operations (coupling and decoupling) using only full route release for the routes where these operations occur. That way, we can ignore the precise movements of the train(s) within the route.

B1. Train Reversal

Consider the example shown in Figure A.15. In the figure, train b stops at the station in v_2. If we let $\overline{v}_1$ (resp. $\overline{v}_2$) represent the reverse of v_1 (resp v_2), we can let the train reverse by letting $\overline{v}_1$ follow v_2 after an appropriate delay added. Alternatively, we can let $\overline{v}_2$ follow v_2 with no delay added, and let $\overline{v}_1$ follow $\overline{v}_2$ normally. Figure A.17 shows these two modelling options.

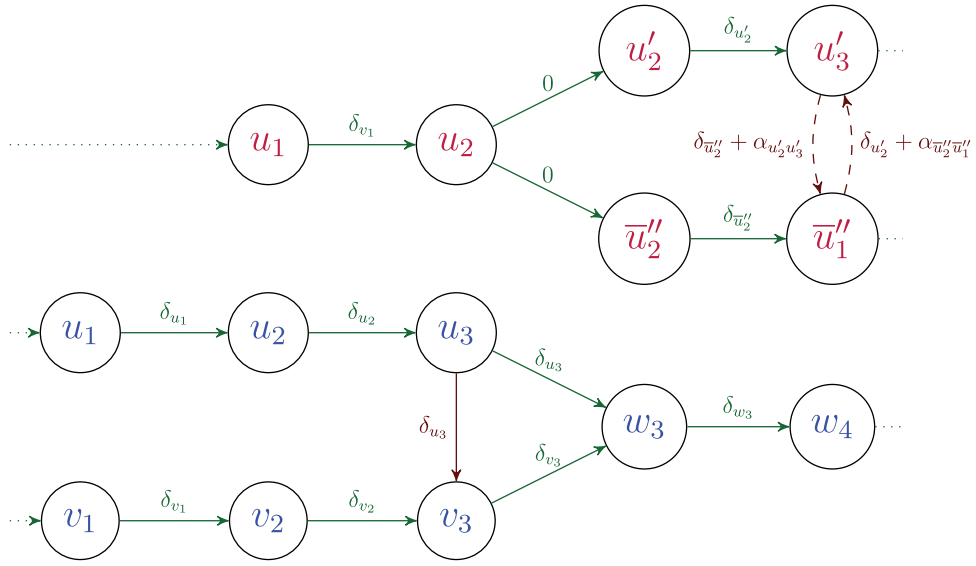


Fig. A.2. Modeling shunting operations: coupling and decoupling.

B2. Shunting Operations

When modelling shunting operations, specifically coupling and decoupling, we again use only full route release for the routes where the operations occur.

Figure A.18 show how we can model coupling and decoupling the trains in Figure A.15. u' and u'' represent the two parts of train a after it separates. In the figure, the two halves proceed in opposite directions. w represent the new trains resulting and train b connecting to train a to the right of the station. The δ values must be adjusted to account for reduced speed during shunting operations. The alternative arcs are added to ensure at most one train is moving in the interlocking route at any given time.

References

[1] SINTEF, GOTO Project, Accessed: 28 July 2021, (https://www.sintef.no/en/projects/2019/goto/).

[2] Railway Signalling and Interlocking. Theeg G, Vlasenko S, editors. 3rd. Leverkusen, Germany: PMC Media House; 2020.

[3] Cacchiani V, Huisman D, Kidd M, Kroon L, Toth P, Veelenturf L, Wagenaar J. An overview of recovery models and algorithms for real-time railway rescheduling. Transportation Research Part B: Methodological 2014;63:15–37. doi:10.1016/j.trb.2014.01.009.

[4] Corman F, Meng L. A review of online dynamic models and algorithms for railway traffic management. IEEE Trans. Intell. Transp. Syst. 2015;16(3):1274–84. doi:10.1109/TITS.2014.2358392.

[5] Fang W, Yang S, Yao X. A survey on problem models and solution approaches to rescheduling in railway networks. IEEE transactions on intelligent transportation systems 2015;16(6):2997–3016.

[6] Chen Z, Li S, DAriano A, Yang L. Real-time optimization for train regulation and stop-skipping adjustment strategy of urban rail transit lines. Omega 2022;110:102631. doi:10.1016/j.omega.2022.102631.

[7] Zhang H, Li S, Wang Y, Yang L, Gao Z. Collaborative real-time optimization strategy for train rescheduling and track emergency maintenance of high-speed railway: A Lagrangian relaxation-based decomposition algorithm. Omega 2021;102:102371. doi:10.1016/j.omega.2020.102371.

[8] Wolsey LA, Nemhauser GL. Integer and combinatorial optimization, volume 55. John Wiley & Sons; 1999.

[9] Queyranne M, Schulz AS. Polyhedral approaches to machine scheduling. TU, Fachbereich 3, Berlin; 1994.

[10] Harrod S. Modeling network transition constraints with hypergraphs. Transportation Science 2011;45(1):81–97.

[11] Ahujia RK, Magnanti TL, Orlin JB. Network flows: Theory, algorithms and applications. New Jersey: Rentice-Hall 1993.

[12] Desaulniers G, Desrosiers J, Solomon MM. Column generation, volume 5. Springer Science & Business Media; 2006.

[13] Reynolds E, Ehrgott M, Maher SJ, Patman A, Wang JY. A multicommodity flow model for rerouting and retiming trains in real-time to reduce reactionary delay in complex station areas. Optimization Online 2020. http://www.optimization-online.org/DB_HTML/2020/05/7816.html.

[14] Barnhart C, Hane CA, Vance PH. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. Operations Research 2000;48(2):318–26.

[15] Caimi G, Fuchsberger M, Laumanns M, Lüthi M. A model predictive control approach for discrete-time rescheduling in complex central railway station areas. Comput. Oper. Res. 2012;39(11):2578–93. doi:10.1016/j.cor.2012.01.003.

[16] Pellegrini P, Marlire G, Rodriguez J. Optimal train routing and scheduling for managing traffic perturbations in complex junctions. Transportation Research Part B: Methodological 2014;59(C):58–80. doi:10.1016/j.trb.2013.10.013.

[17] Zhu T, Mera JM, Suarez B, Maroto J. An agent-based support system for railway station dispatching. Expert Syst. Appl. 2016;61:39–52. doi:10.1016/j.eswa.2016.05.011.

[18] He B, Chen P, DAriano A, Chen L, Zhang H, Lu G. A microscopic agent-based simulation for real-time dispatching problem of a busy railway passenger station. In: 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC); 2020. p. 1–7. doi:10.1109/ITSC45102.2020.9294651.

[19] Foglietta S, Leo G, Mannino C, Perticaroli P, Piacentini M. An optimized, automatic tms in operations in roma tiburtina and monfalcone stations. WIT Transactions on the Built Environment 2014;135:635–47.

[20] Bentzrd SB. Bane Nor tr ikke love bedre punktlighet i 2020. Aftenposten 2019. https://www.aftenposten.no/meninger/nyhetsanalyse/i/K328MX/bane-nor-toer-ikke-love-bedre-punktlighet-i-2020.

[21] Lamorgese L, Mannino C. A Noncompact Formulation for Job-Shop Scheduling Problems in Traffic Management. Operations Research 2019;67(6):1586–609. doi:10.1287/opre.2018.1837.

[22] Lamorgese L, Mannino C, Natvig E. An exact micro macro approach to cyclic and non-cyclic train timetabling. Omega 2017;72:59–70. doi:10.1016/j.omega.2016.11.004.

[23] Mannino C, Nakkerud A, Sartor G. Air traffic flow management with layered workload constraints. Computers & Operations Research 2021;127:105159. doi:10.1016/j.cor.2020.105159.

[24] A. Nakkerud, Rail infrastructure data for oslo central station, Data in Brief (In Press).