A column-generation-based matheuristic for periodic train timetabling with integrated passenger routing

Bernardo Martin-Iradi, Stefan Ropke

Abstract
In this study, the periodic train timetabling problem is formulated using a time-space graph formulation. The problem is tested on the morning rush hour period of the Regional and InterCity train network of Zealand, Denmark. The implemented solution method is based on an iterative model that applies what we define as a dive-and-cut-and-price procedure. An LP relaxed version of the problem with a subset of constraints is solved using column generation where each column corresponds to the train paths of a line. Violated constraints are added by separation and a heuristic process is applied to help to find integer solutions. The passengers are routed based on each resulting timetable allowing the method to estimate the total passenger travel time. The solution approach shows robust performance in a variety of the scenarios being able to find good quality solutions in terms of travel time and path length relatively fast. In addition, the graph formulation covers different real-life constraints and has the potential to easily be extended to accommodate more constraints.

Keywords: Transportation, Periodic Train Timetabling, Matheuristics, Column Generation, Passenger Routing

1. Introduction
The planning process of railway companies is complex and is usually categorized into three main levels: strategic, tactical and operational (Bussieck et al. [1997]). These levels form a hierarchical process used as a decision-making tool where each of the levels includes different problems whose solution is used as an input for the problems at the subsequent level as depicted in Figure 1. The strategic level takes care of the long-term planning problems such as defining the infrastructure of the network (Network Planning Problem (NPP)) or defining the lines, their stopping patterns and their frequencies along the network (Line Planning Problem (LPP)). Next, the problems at the tactical level cover the medium-term planning processes such as generating the timetables for the lines (Train Timetabling Problem (TTP)), allocating the trains to the different tracks in the network (Train Routing Problem (TRP)), or specifying the assignment of the rolling stock and crew to the trains (Rolling Stock Scheduling Problem (RSSP) and Crew Scheduling Problem (CSP)). Finally, there is the operational level where the short-term planning processes are managed to cover real-time operations such as train re-scheduling or delay management.
In this study, the focus is mainly on the generation of timetables which is at the tactical level of the planning process. As mentioned before, the strategic level provides a network and the lines that are supposed to run on the network. Then, the problems at the tactical level assume that the infrastructure is fixed and try to allocate the available resources in the most efficient way.

1.1. Train timetabling

The process of generating a timetable for a given network of lines is formulated as the Train Timetabling Problem (TTP). Its main goal is to determine the arrival and departure times at the stations for each of the train lines. The departure and arrival times are subjected to multiple track capacity constraints and specific requirements from the railway operating company. An obvious example of track capacity constraints is that two trains cannot be in the same track segment at the same time whereas the requirements from the operating company can be very diverse (e.g. forcing specific lines to synchronize at specific stations or spreading lines with similar stopping pattern along the timetable period). In order to avoid having two trains at the same track segment at the same time, a headway is defined. The headway refers to the minimum time interval between two consecutive train movements. The headway is defined by the signaling system along the track. These signals define the so-called blocks and enforce that only one train can be in a block at a time. Likewise, a headway may be defined for both departures and arrivals of consecutive trains along the same track segment. Moreover, dwelling restrictions may be applied, requiring the train to stop a minimum time interval at stations. A minimum dwell time is necessary to allow passengers to get on and off the train as well as changing drivers at specific stations. In the same way, minimum running times between two stations may be enforced mainly due to the train speed, acceleration or breaking capabilities and track segment specifications. Moreover, an additional buffer time is usually considered to define the minimum running times called timetable margin.

Several objectives can be considered when creating a timetable. In general, these objectives are related to three main groups: customer satisfaction, robustness and cost-efficiency. These objectives may be conflicting in most cases. For instance, passengers would always prefer to have direct connections to their destinations at a high frequency, however, this would incur in an
enormous operational cost for the train operating company (TOC), assuming a feasible timetable exists. Therefore, a compromise between conflicting objectives should be found.

From the passenger’s point of view, minimizing the travel time or the transfer and waiting times at stations are good examples of objective functions. Also, the availability of seats and comfort at the train, or the ticket fares are factors that affect customer satisfaction.

On the other hand, train operators may be more interested in robust timetables that do not have a high operational cost. A good example of objectives for a robust timetable can be maximizing headway between consecutive trains. For cost-effectiveness, minimizing rolling stock circulation and crew scheduling can be seen as attractive objectives that are directly related to the timetable.

1.2. Paper structure

Section 2 lists several methods to solve the TTP through an extensive literature review. In Section 3 the model used and its characteristics are described. Section 4 describes the case study, which data was collected and used in the study as well as the scope covered. The solution method used to solve the problem is described in Section 5 where each of the steps in the algorithm and how they interact together are carefully explained. Section 6 summarizes the computational results obtained from different tests and conducts an analysis of them with further study proposals. The paper concludes in Section 7 with a generic overview of the whole study.

2. Literature review

The literature about train scheduling is extensive. The different publications apply a wide range of methods to different cases. Some of them consider just a corridor or a junction whereas others study a whole network. Moreover, the nature of the resulting timetable (i.e. periodic or non-periodic) also affects the algorithm proposed. Several extensive surveys have been published (see Cordeau et al. (1998), Caprara et al. (2007), Hansen (2009), Lusby et al. (2011), Cacchiani and Toth (2012) or Harrod (2012)).

Most of the studies that model a network assuming the periodicity of the timetable (periodic timetable) are based on the Periodic Event Scheduling Problem (PESP) first introduced by Ser-afini and Ukovich (1989). Odijk (1996) proposed a cutting plane algorithm to solve the PESP. Integer variables are used to ensure the travel intervals are respected and continuous variables to determine the arrival and departure times modulo the period. Later, Nachtigall (1998), Liebchen and M¨ohring (2002) and Peeters (2003) studied the Cycle Periodicity Formulation (CPF) that leads to a significant speed up in the solution times compared to earlier models. Given the effectiveness of the PESP, these models have been used to solve many network cases, whereas non-periodic approaches are used more often to model single-line corridors or congested networks where it may not be possible to schedule all trains in an efficient way.

Szpiegel (1973) presented one of the first Integer Linear Program (ILP) formulations for the non-periodic TTP. The formulation is regarded as a job-shop scheduling problem where jobs (trains) need to be assigned to machines (track segments). Szpiegel (1973) solved it using branch-and-bound applied to a Brazilian single-track line. Jovanovic and Harker (1991) proposed a Mixed Integer Linear program (MILP) formulation where the arrival/departure times are defined with continuous variables and the order of trains with binary variables and tries to find a reliable timetable. Carey and Lockwood (1995) proposed a mix of heuristic and branching procedure to solve a similar MILP as the one presented by Jovanovic and Harker (1991) in a one-way corridor,
and Carey (1994) extended it to a two-way corridor showing that no additional constraints are needed. In general, most of the models proposed for solving non-periodic timetables are used for scheduling multiple competing timetables from different operators.

Furthermore, Brännlund et al. (1998) introduced a pure ILP formulation where the time was discretized and therefore, the formulation could be represented as a graph where the nodes represent the arrival and departure time instants to each station. This new formulation is referred to as time-space graph formulation but cannot be directly applied to large instances due to the large number of binary variables. As a result, further studying the LP relaxation of the model becomes more attractive and different methods have been developed based on it. The ILP formulation proposed by Caprara et al. (2002) defines a variable for each arc in the graph and it is solved using Lagrangian relaxation combined with sub-gradient optimization. Cacchiani et al. (2008) proposed a formulation where the variables refer to whole paths instead, and solved it applying column generation together with separation techniques. Cacchiani et al. (2010b) extended the formulation presented by Caprara et al. (2002) to be applied in a network considering both passenger and freight trains and solved it using a similar procedure. Min et al. (2011) proposed a method for solving the train-conflict resolution problem with a column-generation based algorithm that takes advantage of the separability of the problem. Using a heuristic for the pricing problem (PP), the method is able to get near optimal conflict-free solutions in a few seconds. Cacchiani et al. (2013) applied dynamic programming to solve the clique constraints that arise in the graph formulations and developed an exact method whose performance is compared with various heuristics in Cacchiani et al. (2010a).

Last but not least, combining train timetabling and passenger routing has also been studied. Kinder (2008) extended the PESP model to a time-space graph and implemented an iterative approach where the timetable is re-planned after doing passenger routing. Gattermann et al. (2016) present an integrated model that finds timetables and passenger routes in which passengers are distributed temporally using time-slices. Borndörfer et al. (2017) also integrates timetabling and passenger routing in one model. The model tests and analyzes different passenger routing models on timetable optimization yielding significant improvements in travel time. Farina (2019) implements a two-phase heuristic method based on the large neighborhood search that integrates periodic train timetabling with passenger routing and applies it to the same network as studied in this paper.

2.1. Contribution and comparison to existing models

The modeling approach used in this paper is based on the time-space graph proposed in Caprara et al. (2002). As discussed in the literature review, this modeling approach has also been used in several later papers (e.g. Cacchiani et al. (2008) and Cacchiani et al. (2010b)). In Caprara et al. (2002), the integer programming model was solved using a Lagrangian relaxation heuristic. The Lagrangian subproblem solves a longest path problem through an acyclic network. In Cacchiani et al. (2008), the problem was solved using column generation where the pricing problem also searches for longest paths through an acyclic network. We also solve the problem using column generation but use a pricing problem that can determine 2 or 4 paths in one go. The pricing problem is solved as a standard shortest path problem (further details in Sections 3 and 5). This is possible due to tight frequency and symmetry constraints. There are several benefits of this approach: 1) The symmetry and frequency constraints are entirely handled in the pricing problem and fewer constraints are necessary in the master problem. 2) The LP relaxation produced by the master problem is potentially stronger compared to an approach that handles
symmetry and frequency constraints in the master problem. 3) Fewer pricing problems must be solved. We believe that this is a major contribution of our paper.

Caprara et al. (2002) already constructed a cyclic timetable. We use this as a basis to generate cyclic timetables with a one hour period, useful for modeling the passenger train timetabling problem that a train operator faces. Normally, this problem is solved using a PESP model and, to the best of our knowledge, it is the first time that the time-space graph approach is used for this application.

The solution approach presented in this study constructs the timetable while considering the routing of the passengers, focusing on the passenger travel time. As the literature review shows, this is an emerging topic in passenger train timetabling and we believe that the paper at hand proposes a simple but useful approach for integrating the passenger routing with the train timetabling problem.

Finally, it should be mentioned that the work in this paper was carried out in parallel with the work presented in Farina (2019) and shares similarities in terms of modeling and data with the aforementioned paper. However, the two papers are different in terms of solution methods where a meta-heuristic was employed in Farina (2019).

3. Problem formulation

The notation is based on the one from Cacchiani et al. (2010a). Let $S = \{1, \ldots, s\}$ denote the set of stations in the network. The network can be represented as a mixed multi-graph $N = (S, E \cup A)$ where each vertex $i \in S$ represents a station in the network and each edge $e = \{h, i\} \in E$ represents a single-track segment between two stations with no intermediate stations in between that is used by trains traveling in both directions (i.e. from $h$ to $i$ and from $i$ to $h$). Finally, each arc $a = (h, i) \in A$ represents a double-track segment between station $h$ and $i$ with no intermediate stations that can be used only by trains traveling in one direction (i.e. from $h$ to $i$). The graph can contain multiple arc/edges connecting the same two stations. For instance, in the network here studied there are segments with four tracks between two same stations (two in each direction). Therefore, the adjacent stations in between can be connected with four arcs (two in each direction) in the multi-graph. For convenience, for each station $i \in S$, let denote:

- $d(i, e)$: minimum time interval between consecutive departures of trains traveling in the same direction from $i$ on the track segment $e$. 
- $a(i, e)$: minimum time interval between consecutive arrivals of trains traveling in the same direction at $i$ on the track segment $e$. 

Moreover, in the case of single-tracks, additional time interval requirements need to be set for trains traveling in opposite directions. Therefore, for each edge $e \in E$ and station $i$ of $e$, let denote:

- $f(i, e)$: minimum time interval between an arrival at $i$ on $e$ and a departure from $i$ on $e$ of trains traveling in opposite directions.
• \(g(i, e)\): minimum time interval between a departure from \(i\) on \(e\) and an arrival to \(i\) on \(e\) of trains traveling in opposite directions.

• \(h(i)\): minimum time interval between an arrival to \(i\) and an arrival to \(i\) of trains traveling in opposite directions.

In this case study, due to safety requirements, a minimum value of \(d(i, e), a(i, e)\) and \(h(i)\) is defined, whereas there is no minimum time for \(f(i, e)\) and \(g(i, e)\). However, the value of \(g(i, e)\) is implicitly given by

minimum travel time from \(i\) to \(h\) on \(e\) + minimum travel time from \(h\) to \(i\) on \(e\),

where \(h\) is the other endpoint of \(e\).

### 3.1. Lines and timetables notation

The different lines link two major stations with a number of intermediate stations in between. Let \(L = \{1, ..., l\}\) denote the number of operating lines in the network space and \(D = \{1, 2\}\) the direction of the line, \(D = 1\) for direction out of Copenhagen and \(D = 2\) for direction towards Copenhagen. Let \(\Upsilon\) be the set of trains that cover the \(L\) lines and \(D\) directions. For each train \(j \in \Upsilon\) let denote \(f_j\) the starting station and \(e_j\) the ending station. Let \(S_j = \{f_j, ..., e_j\} \subseteq S\) be the ordered set of stations visited by train \(j\) (stopping or not). Some segments between stations are formed by quadruple-track segments, meaning that each train can choose between two tracks to travel along that track segment. In this study, the quadruple-track segments connects various consecutive stations and it has been assumed that the train runs along the same track and cannot switch to the other track during the whole quadruple segment (see Figure 2). Let \(N_j = (S_j, \mathcal{A}_j)\) be the auxiliary network for each train \(j \in \Upsilon\) where each arc in \(\mathcal{A}_j\) is either an arc in \(A\) or an edge in \(E\) with an orientation, corresponding to the unique travel direction of \(j\) along the single-track. A timetable for each train is given by the departure time at \(f_j\) and the arrival time at \(e_j\), and the arrival and departure times for the intermediate stations \(f_j + 1, ..., e_j - 1\). Let \(\phi_j(a)\) denote the running time along arc \(a \in \mathcal{A}_j\) of train \(j \in \Upsilon\). Let \(\omega_j{}^{\text{min}}(i)\) denote the minimum dwell time at station \(i\) for train \(j \in \Upsilon\) where \(i \in S_j \setminus \{f_j, e_j\}\). In the same way, there is an upper bound in the dwell time (i.e. \(\omega_j{}^{\text{max}}(i)\)) in the form of an additional percentage of the minimum dwell time \((\omega_j{}^{\text{max}}(i) \propto \omega_j{}^{\text{min}}(i))\). Note that, for a line containing \(N\) stations, there are \(N-1\) minimum running
times and N-2 minimum dwell times defined in one direction. The mentioned parameters above are defined for each train meaning that the running and dwell time sets are defined independently for trains in different directions for the same line, as they may differ. Finally, the time horizon is defined as $T = \{1, \ldots, t\}$ referring to a whole hour discretized into time instants of half a minute ($|T| = 120$ time instants) and each line has an associated running frequency $F^j$ indicating how many trains per hour cover each direction of that line.

3.2. A graph representation

![Graph representation](image)

The problem can be defined using graphs to represent the possible timetables (from now on referred to as train paths). A graph can be defined for each train $j \in \Upsilon$. Let $G^j = (V^j, R^j)$ be a directed and acyclic space-time graph (from now on referred to as Train graph) in which the nodes represent the arrivals or departures at a station at a given time instant. Figure 3 shows an example of a train path represented using a time-space graph.

The node set has the form

$$V^j = \{\sigma^j, \tau^j\} \cup \bigcup_{a \in [h, i] \in A^j} (U^a_t \cup W^a_h)$$

where $\sigma^j$ and $\tau^j$ are the artificial source node and artificial sink node respectively and the sets $W^a_h$ for $h \in S^j \setminus \{e\}$ and $U^a_i$ for $i \in S^j \setminus \{f\}$ represent the set of time instants where a train can depart from or arrive to station $h$ or $i$ on the track represented by arc $a \in A^j$ respectively (also called departure and arrival nodes). Let $u, w \in V^j$ be nodes of the node set and let $\theta(u)$ be the time instant associated with node $u$. Furthermore, let $\Delta(u, w) := \theta(w) - \theta(u)$ denote the time interval between nodes $u$ and $w$ if $\theta(w) \geq \theta(u)$ and $\Delta(u, w) := \theta(w) - \theta(u) + T$ otherwise. Due to the periodic nature of the time horizon $T$, it is said that node $u$ precedes or coincides with node $w$ (i.e. $u \leq w$) if $\Delta(w, u) \geq \Delta(u, w)$ as it is assumed that all the travel times used in this study case are far from the time horizon of one hour. Table 1 illustrates the time interval calculation with one example. For convenience, for each station $i \in S^j$, let denote $\delta^j_N(i) \subseteq A^j$ the set of edges
Table 1: Example of the time interval calculation between two nodes with a cycle time $|T| = 60$

|       |       |       |
|-------|-------|-------|
| $\theta(u)$ | $\theta(w)$ | $\Delta(u, w)$ |
| 10    | 15    | 5     |
| 15    | 10    | 55    |

incident to $i$ and arcs leaving $i$, and $\delta^-_i(i)$ the set of edges incident to $i$ and arcs entering $i$. The arc set $R^j$ for each graph can be defined by four main types of arcs.

**Starting arc set:** These arcs connect the artificial source node with the set of nodes for the departure of the first station in the line. These arcs have a null cost (free arcs).

**Segment arc set:** These arcs connect the nodes related to the departure time from one station to the nodes related to arrival time to the next station in the line. Furthermore, the arc needs to satisfy that $\Delta(w, u) = \phi_j(a)$ where $\phi_j(a)$ denote the travel time for arc $a \in A^j$. The cost of the arc corresponds to the travel time between the departure and arrival instants in the respective sets.

**Dwell arc set:** These arcs connect the nodes related to the arrival time to one station with the nodes related to departure time from the same station in the line. Furthermore, the arc needs to satisfy that $\Delta(u, w) \in [\omega_{\min}^j(i), ..., \omega_{\max}^j(i)]$ for $i \in S^j \setminus \{f_j, e_j\}$. The cost of the arc corresponds to the dwell time between the arrival and departure instants in the respective sets.

**Ending arc set:** These arcs connect the set of nodes of the arrival to the last station in the line with the artificial sink node. These arcs have a null cost (free arcs).

As a result, the timetable for train $j \in \Upsilon$ is defined by any path from the artificial source node $\sigma^j$ to the artificial sink node $\tau^j$.

3.2.1. Main assumptions

The final graph formulation presented in this study is based on the assumption that the travel time of each train along each track segment joining two stations is fixed. In other words, it is not possible to slow down the train along the track segment and, therefore, the departure time from one station uniquely determines the arrival time at the next station. Even if slowing down is something that has to be done at the operational level, this assumption is supported by the fact that, in practice, slowing down a train between two stations in most cases is equivalent to forcing the train to stop in an endpoint station of the track segment for a longer time and then to travel at the regular speed along the track. This statement is not true in general but it holds for realistic cases. In particular, experimental results performed by Caprara et al. (2006) show that the solution values found by heuristic procedures are marginally affected by this additional constraint, whereas the corresponding running time per iteration is widely reduced, since the graph $G$ turns out to be much smaller (for each train, the number of segment arcs between two stations is equal to the number of departure nodes). Furthermore, the above assumption simplifies the mathematical representation of the problem, yielding simpler and stronger overtaking and crossing constraints (see sections 3.4.3 and 3.4.4).

Another characteristic of the model assumed is the need for a symmetric timetable. When the train services are identical in both running directions it is easier to plan the timetable since the train path in one direction uniquely defines the path of the train in the opposite direction. Therefore, symmetric timetables are easier to plan and are more attractive to passengers as they provide equal transfer times in both directions (Liebchen, 2007). Nevertheless, this type of timetable reduces the degrees of freedom in the planning process and it is more suitable only when the passenger demands are similar in both directions.
As a result, these two main assumptions can lead to a new, more efficient, graph formulation. On one side, keeping the running times fixed reduces the number of nodes to half since the arrival of a train is directly defined by the previous departure. On the other side, assuming symmetric paths for each line requires just creating one train path for a line, as the remaining line train paths are automatically defined.

### 3.3. Symmetric Line graph

![Figure 4: Representation of a path in the Symmetric Line graph as the combination of two paths in the respective Train graphs. In this example the symmetry gap is set to $K = \pm 1$.](image)

The Symmetric Line graph formulation defines one graph for each line instead of one per train as stated initially, meaning that the number of graphs needed is equal to the number of lines considered. Ideally, each of the Symmetric Line graphs would include half of the nodes of one Train graph due to the fixed running times and symmetric paths. Nevertheless, in practice, the running times of trains running in opposite directions along the same track segment are sometimes slightly different, meaning that two exactly symmetrical paths cannot be achieved. Therefore, a maximum symmetry gap $K$ is considered, meaning that the departure time of the train in one direction and the arrival time of the train in the other direction are considered symmetrical if the sum of both time instants sum up to a value that is within the range of the cycle time and the gap considered (i.e., $|T| \pm K$). A path in the Symmetric Line graph corresponds to two symmetric train paths in opposite directions as it can be seen in Figure 4. In this figure, the exactly symmetrical times at a station are depicted by larger nodes in the Symmetric Line graph and the symmetric instants that are within the gap considered ($K$) are depicted with smaller nodes.

Each node in the graph represents the departure and arrival of two symmetrical train paths of the same line along a track segment. In other words, one node from the Symmetric Line
graph notation is equivalent to four nodes of the Train graph notation (see Figure 5). Note that, symmetry is only checked in one of the stations of the arc because it is assumed that the running times of both trains along the same track are very similar (if not identical) and, therefore, a slight deviation in the symmetry in the other endpoint station can be rescheduled adapting the dwell time of one of the trains in accordance.

Due to the symmetry gap allowed, each time instant in one direction has a range of symmetrical time instants in the opposite directions. As a result, the number of nodes increases proportionally to the range of symmetry allowed.

Additionally, the symmetry is checked at both end stations of the line, meaning that a final set of trivial nodes is added to the graph that measures the arrival of the train in $D_1$ to its last station and the equivalent departure from the first station of the train in $D_2$.

Regarding the arc set, the fact of assuming fixed running times allows to merge the segment and dwell arc in a single segment+dwell arc. The weight of these arcs is given by the sum of running time and dwell for both trains. At each single-track segment, all arcs that result in incompatible departures, arrivals or crossings are not included in the graph. This ensures that all graph paths correspond to feasible and compatible paths of the two trains in the line.

As seen in Figure 2, the choice of tracks in the quadruple-track segment is done by duplicating the set of nodes along the segment. Therefore, in this case it is enforced that both trains of the line choose the same track in both directions.

In the rush hour lines, trains run only in one direction and, therefore, only one train path is needed. As a result, the symmetry requirements are not necessary in this case and the nodes only denote the time instant of the single train path. The resulting graph is similar to the Train graph but with half of the nodes due to the assumed fixed running times.

In the case that the frequency of the line (i.e. $F^l$) is two trains per hour and direction, the outlook of the Symmetric Line graph does not get altered. As the train paths are exactly 30 min-

Figure 5: Representation of the Train graph nodes (black) associated with one node of the Symmetric Line graph formulation.
utes apart from each other, once a train path is defined in one direction, the rest of the frequency trains are uniquely defined. The only alteration affects the weight of the arcs. These do not sum only the time intervals of the two train paths but of the four of them.

3.4. ILP formulation

In this section, the model is formulated as an ILP. In order to illustrate the different parts of the formulation, the notation of the Train graph is used. As it is explained in Section 3.3, the set of nodes of the Symmetric Line graph are formed by combinations of node sets from the Train graph formulation.

3.4.1. Formulation without track capacity constraints

The problem can be formulated as a version of the Set Packing Problem (SPP) that aims to minimize the sum of total path lengths. The binary variable \( \lambda_q \in \{0, 1\}, q \in Q \) defines if the group of line paths \( q \) is included in the optimal solution where \( Q \) is the set of possible line group paths. The parameter \( c_q \) denotes the cost of choosing the group of line paths \( q \in Q \) that is the sum of path lengths. The formulation without the track capacity constraints is stated as follows:

\[
\begin{align*}
\min & \quad \sum_{q \in Q} c_q \cdot \lambda_q \\
\text{s.t.} & \quad \sum_{q \in Q} \lambda_q = 1 \quad \forall l \in L \\
& \quad \lambda_q \in \{0, 1\} \quad \forall q \in Q
\end{align*}
\]

The objective function minimizes the cost (path lengths) of the solution train paths. Constraints (2) ensure that train paths are chosen to cover each line where \( Q' \) is the set of possible line group paths for line \( l \in L \) and constraints (3) state the binary property of the decision variable.

3.4.2. Headway constraints

Headway constraints are one of the track capacity constraints and ensure the minimum headway times between consecutive arrivals and departures at stations in the network.

\[
\begin{align*}
\sum_{\nu \in U_i^a: \nu \leq u} & \sum_{q \in Q} \lambda_q \leq 1, i \in S, a \in \delta_N^{-}(i), u \in U_i^a, & (4) \\
\sum_{\nu \in W_i^w: \nu \leq w} & \sum_{q \in Q} \lambda_q \leq 1, i \in S, a \in \delta_N^{+}(i), w \in W_i^w, & (5) \\
\sum_{e \in \delta_N(i) \cap E} & \sum_{\nu, u \in U_i^t: \nu \leq u} \sum_{q \in Q} \lambda_q \leq 1, i \in \hat{S}^e, t \in T, & (6)
\end{align*}
\]
Let $Q_v$ be the set of line group paths that use node $v$. Constraints (3) and (5) enforce that the minimum headway distance between consecutive arrivals and departures at each station respectively, of trains in the same direction, is respected. Moreover, constraints (6) ensure that in the single-track segments the minimum headway between trains arriving in opposite directions is respected.

### 3.4.3. Overtaking constraints

![Figure 6: Illustration of an overtaking where $a(h,a) = 2$ and $d(i,a) = 2$. The left one is the simple version of the constraint while the right one is the stronger version implemented in this study.](image)

Let $Q_v$ be the set of line group paths that use node $v$. Constraints (3) and (5) enforce that the minimum headway distance between consecutive arrivals and departures at each station respectively, of trains in the same direction, is respected. Moreover, constraints (6) ensure that in the single-track segments the minimum headway between trains arriving in opposite directions is respected.

It is not allowed that two trains traveling in the same direction on the same track overtake each other.

A basic example of an overtaking is shown on the left side of Figure 6 where both train departures are incompatible. The basic overtaking constraint would enforce that, at most, one orange train will depart from $t = 0$ or one green train will depart from $t = 2$. In this study, a stronger version of this basic constraint is formulated based on the ones from Cacchiani et al. (2010b).

The following constraints (7) are defined for every pair of trains $j, k$ along an edge/arc $a = (i, h)$ that is an arc in both auxiliary networks $N^l$ and $N^k$. Moreover, $j$ is considered the "slow" train and $k$ is the train that can actually overtake it. Therefore, the travel time of train $j$ should be greater than the one from train $k$ (i.e. $\phi^j(a) > \phi^k(a)$). For a constraint, we define an earliest possible departure from $i$ for trains $j$ and $k$. These departure nodes are denoted $v_1$ and $v_2$ respectively. Node $v_1 \in W^l_i \cap V^j$ and node $v_2 \in W^k_i \cap V^k$ correspond to departure nodes that are incompatible with each other (i.e. if train $j$ departs at $\theta(v_1)$, then train $k$ cannot depart at $\theta(v_2)$ and vice versa). The two trains $j, k$ are considered incompatible when either $\min\{\Delta(v_1, v_2), \Delta(v_2, v_1)\} < d(i,e)$, meaning that their departures are too close in time or $\min\{\Delta(u_1, u_2), \Delta(u_2, u_1)\} < a(i,e)$ where $u_1, u_2$ are the respective arrival nodes for $j, k$ corresponding to $v_1, v_2$, meaning that their arrivals to the next station are too close in time or $v_1 < v_2 < u_2 < u_1$ meaning that train $k$ overtakes train $j$ along the track. Then, $v_3 \in W^l_i \cap V^j$ can be defined as the earliest possible departure of train $j$ that is compatible
with \( \theta(v_2) \) such that \( v_1 \prec v_3 \). Analogously, \( v_4 \in W_i^F \cap V^{k,l} \) can be defined as the earliest possible departure of train \( k \) that is compatible with \( \theta(v_1) \) such that \( v_2 \prec v_4 \). It can be seen that any departure of train \( j \) from \( [v_1, v_3) \) is incompatible with any departure of train \( k \) from \( [v_2, v_4) \).

This stronger version of the constraint is illustrated in the right side of Figure 6. Let \( Q_w^j \) be the set of line group paths that use node \( w \) and belong to train \( j \) of line \( I \). Nodes \( v_1 \) and \( v_3 \) are depicted as the first and second orange nodes in time respectively and nodes \( v_2 \) and \( v_4 \) are depicted as the first and second green nodes in time respectively. Note that in the illustration the minimum departure and arrival headway \( (a(i, e) \) and \( d(i, e)) \) are respected for the trains but they overtake each other along the track.

\[
\sum_{w \in W_i^F \cap V^I} \lambda_q + \sum_{w \in W_j^F \cap V^J} \lambda_q \leq 1, \forall j, k \in T, v_1, v_2 \in W_i^F,
\]

\[
(\text{where } l' \neq l, d' = d, i, h \in S^I \cap S^J, a = (i, h) \in (A^I \cap A^J)) \quad (7)
\]

### 3.4.4 Crossing constraints

It is not allowed that two trains traveling in opposite directions are on the same single-track segment at the same time.

A basic example of a crossing is shown on the left side of Figure 7 where both departures are incompatible. The basic constraint corresponding to this crossing would enforce that, at most, one orange or green train will depart from \( t = 0 \). In this study, a stronger version of this basic constraint is formulated based on the ones from Cacchiani et al. (2010b).

The following constraints (8) are defined in a similar way to constraints (7). They are defined for every pair of trains \( j, k \) traveling in opposite directions such that \( e = (i, h) \) and \( (h, i) \) are arcs
in the auxiliary networks $N^j$ and $N^k$ respectively and correspond to the set of edges $E$ in the network. For a constraint, we define an earliest possible departure from $i$ and $h$ for trains $j$ and $k$ respectively. These departure nodes are denoted $v_1$ and $v_2$ respectively. Node $v_1 \in W^e_i \cap V^j$ and node $v_2 \in W^e_k \cap V^k$ correspond to departure nodes that are incompatible with each other (e.g. if train $j$ departs at $\theta(v_1)$, then train $k$ cannot depart at $\theta(v_2)$ and vice versa). The two trains $j, k$ are considered incompatible when either $u_2 \leq v_1$ and $\Delta(u_2, v_1) < f(i, e)$ or $u_1 \leq v_2$ and $\Delta(u_1, v_2) < f(i, e)$, meaning that arrival to and departure from the same station are too close in time or $v_1 < u_2$ and $v_2 < u_1$ meaning that train $j$ and train $k$ cross each other along the track. Then, $v_3 \in W^e_i \cap V^j$ can be defined as the earliest possible departure of train $j$ that is compatible with $\theta(v_2)$ such that $v_1 < v_3$. Analogously, $v_4 \in W^e_k \cap V^k$ can be defined as the earliest possible departure of train $k$ that is compatible with $\theta(v_1)$ such that $v_2 < v_4$. It can be seen that any departure of train $j$ from $[v_1, v_3]$ is incompatible with any departure of train $k$ from $[v_2, v_4]$.

This stronger version of the constraint is illustrated in the right side of Figure 7. Nodes $v_1$ and $v_3$ are depicted as the first and second orange nodes in time respectively and nodes $v_2$ and $v_4$ are depicted as the first and second green nodes in time respectively. Note that even if the minimum arrival headway $(f(h, e))$ is respected by the trains departing, they cross each other along the track.

$$
\sum_{w \in W^e_i \cap V^j : v_1 \leq w \leq v_3} \lambda_q w + \sum_{q \in Q^e_i} \lambda_q + \sum_{w \in W^e_k \cap V^k : v_2 \leq w \leq v_4} \lambda_q \leq 1, \forall j, k \in \Upsilon, v_1 \in W^e_i, v_2 \in W^e_k \cap V^k
$$

(8)

3.4.5. Frequency constraints

There are specific pairs of lines that share identical or similar first and last stations but have slightly different stopping patterns. These pairs of lines (from now on referred to as Frequency lines) should be spread along the cycle time as much as possible. In order to do so, the frequency constraints behave in the same way as the departure headway constraints (5). Let $T_x$ denote the minimum time interval between consecutive departures of Frequency lines in one direction at each station. Finally let $\Xi := \{(m_1, n_1), ..., (m_k, n_k)\}$ denote the set of Frequency line pairs along the network where $m_k, n_k \in L$.

$$
\sum_{v \in W^e_i : v \leq w} \lambda_q \leq 1, \forall (l', \hat{l'}) \in \Xi, d \in D, w \in W^e_i
$$

(9)

Constraints (9) ensure that all the Frequency lines departures from any common station are spread at least a time interval of $T_x$ in each direction.
4. Case study of Danske Statsbaner (DSB)

The case study presented here aims at defining a passenger timetable for a future network scenario of Zealand, Denmark. More specifically, the scope covered corresponds to a period of one hour during morning rush hour. This period is considered the period with most transit of the day. Once a solution for this period is obtained, it can be repeated periodically removing or adding scheduled rush hour lines in order to roll out the full timetable plan for the day.

4.1. The network

This network scenario covers the Regional, Intercity and IntercityLyn (high-speed) lines running in Zealand as shown in Figure 8. The network is formed by 15 lines covering 43 passenger stations.

The number of tracks and the direction of trains running along them vary along each corridor. Three different types of track segments between stations are present in this network. A single-track segment, where trains can circulate in both directions but there can only be one train on the segment at a time. A double-track segment, where two tracks connect two stations allowing trains to travel in both directions (one track per direction) and a quadruple-track segment, formed by four tracks between two consecutive stations and trains can travel in both directions (two tracks per direction). The quadruple-track segments allow two trains going in the same direction to overtake each other along the segment. In the network considered, there are two main single-track segments: the segment between Holbæk and Kalundborg and the segment connecting Køge Nord and Næstved along the southern corridor. The rest of the network is connected by double-track segments with the exception of the segments between Høje Taastrup and Roskilde that are formed by quadruple-tracks.
4.2. Input data from DSB

Minimum running time: This parameter states the minimum required time for a train to travel between two specific stations. This time interval is usually depending on the rolling stock type and the speed limits on the track segment. A value is given for every track segment connecting two consecutive stations in each line and direction.

Minimum dwelling time: This parameter states the minimum required time for a train to dwell at a specific station. This time is usually the time required by the passengers to board and leave the train. A value is given for every station visited by each line and each direction.

Frequency lines: As mentioned in Section 3.4.3 there are specific pairs of lines that have similar or identical routes which are required by DSB to be as separated as possible in the time-line. There are three pairs of these lines considered in this case study. For example, the two lines reaching Kalundborg.

Minimum headway between trains: In this case study, three minimum headway values are given: 1) Minimum headway between two consecutive departing trains in the same track segment and direction, 2) minimum headway between two consecutive arriving trains in the same track and direction and 3) minimum headway between two consecutive trains arriving from single-tracks in opposite directions.

Origin-Destination matrix: This matrix defines the number of passengers traveling between each pair of stations.

Single-platform stations: Some stations along the single-track segments have only one platform meaning that the station can only host one train at a time and a crossing between two trains is not allowed. It is assumed that, for the rest of stations in the network, any train arriving from an adjacent track segment has an available arriving platform.

5. Solution method

The implemented solution method for the problem formulated is an iterative process that relies on, what we call, a dive-and-cut-and-price procedure. A Restricted Master Problem (RMP) is initialized with a subset of rows. Promising columns and violated cuts are added to it by column generation and separation procedure respectively in order to find an optimal LP solution. Then, branching is enforced through a dive heuristic in order to achieve integrality. Finally, the passengers are routed using the solution timetable and the travel time computed by solving a simple multi-commodity flow problem (MCFP).

Each of the steps in the process is explained in detail in the following sections.

5.1. Column generation procedure

Taking into account the cycle time, the size of the network and the symmetry gap allowed, the number of possible line train paths to be considered is extremely large. In order to handle that amount of variables efficiently, column generation techniques are necessary.

A reduced version of the Master Problem (MP) is initially considered known as the Restricted Master Problem (RMP) that includes only a subset of the variables. These initial variables can just be a set of ”dummy” artificial variables that satisfy the constraints of the RMP. For each line \( l \in L \) a pricing problem is created (i.e. \( PP^l \)) that is in charge of providing line paths objects \( q \in Q^l \) that can potentially improve the current solution.

The formulation of the RMP is identical to the one of the original problem (see constraints (1)-(9)) except for the relaxed version of the decision variable (constraint (10)).

\[
\lambda_q \geq 0 \quad \forall q \in Q
\]
5.1.1. Pricing Problem

The goal of the PP is to find new promising train paths for the RMP. There is one PP per line and their function is to create a group of line train paths (referred to as a column) with the potential to improve the objective function. For example, if line 2 has a frequency of 2 train per hour, the related pricing problem is in charge of creating 4 train paths (2 in each direction) that are feasible between them. Here is where the Symmetric Line graph formulation described in section 3.3 becomes relevant. The use of a single graph for all the train paths of a line reduces the PP to a single shortest path problem. From the fact that all the dual variables affecting the graph are non-positive and they are subtracted from the original weights of the edges, it can be concluded that the graph has always non-negative edge weights. Therefore, and knowing that the graph is directed acyclic (see Section 3.3), this problem can be solved using Dijkstra’s algorithm [Dijkstra, 1959] in polynomial time.

Every time the PP finds a column \( q \in Q \) with a negative reduced cost, it is added as a new variable to the RMP and it is included in all the constraints where it has a non-zero coefficient.

5.2. Separation procedure

It is decided to add Constraints (7)-(9) by separation as the total amount is too large and only a reduced amount of them may be binding. The headway constraints are considered from the beginning in order to provide guidance to the column generation process.

Once the column generation procedure stops providing columns with negative reduced cost the separation procedure is applied. The separation of constraints (7)-(9) is done by enumeration and are checked in the same iteration. Every constraint that is violated by the current solution is added to the RMP.

Once the violated constraints are added to the model, the column generation procedure should be restarted. Adding more constraints to the model modifies the solution space and new columns with negative reduced cost can be found. The overall procedure of column generation and separation is summarized in Algorithm 1.

5.3. Dive heuristic

The optimal solution for the MP can be fractional. In order to find an integer solution, a dive heuristic method is applied. The solution \( \lambda_q \) values are added to each of the graph nodes affected by that column. This measures the “usage” of each node and, if the solution is fractional, this means that some of the graph nodes are fractionally used (see Figure 9). The dive heuristic selects one of the fractionally used nodes and enforces to be part of the final solution, meaning that the final integer solution must contain that node. Apart from fixing the node, all the previously generated columns from the same graph that do not include the node need to be removed from the RMP. Once the heuristic step is concluded, the column generation should be started again.
Algorithm 1 Column generation and Separation pseudo-code

1: procedure colGenAndSep(fixedNodes)
2:  \( x = \{ \} \)  \( \triangleright \) start with empty solution
3:  \( PP \leftarrow fixedNodes \)  \( \triangleright \) fix nodes in graphs
4:  repeat
5:     repeat
6:         \( x \leftarrow solve(RMP) \)
7:     for all lines do
8:         \( \lambda \leftarrow solve(PP(line)) \)  \( \triangleright \) get new column
9:     if \( \hat{c}(\lambda) < 0 \) then
10:        \( RMP \leftarrow \lambda \)  \( \triangleright \) add columns with negative reduced cost
11:     end if
12: end for
13: until no more columns with negative reduced cost
14: \( RMP \leftarrow violatedConstraints(x) \)
15: until no more violated constraints
16: return \( x \)
17: end procedure

as new promising columns may be generated. One advantage of the dive heuristic is that it can lead faster to an integer feasible solution. A disadvantage of this method is that some branches of the tree are left unexplored and forcing the integrality of specific nodes that were fractional can lead to an infeasible final solution. In this study, as a rule of thumb, if the lambda values of the initial dummy columns are used more than 2% in the solution, the algorithm is restarted. Of course, most of the times, there are multiple fractionally used nodes in the solution and a criterion to choose one each time is needed. Constraints (2) and (10) dictate that nodes are used between 0 and 1. In this study, the concept of “most fractional” is used, meaning that a node with a usage fraction closer to 0.5 is prioritized. If there are multiple nodes with the same usage fraction corresponding to the “most fractional”, then a random one is selected. Note that, in this case, the strategy considers 0.4 and 0.6 as “equally fractional” but 0.6 is prioritized as we believe it has a lower risk of leading to an infeasible solution.

The procedure is summarized in Algorithm 2.

Algorithm 2 Dive heuristic pseudo-code

1: procedure diveHeuristic()
2:  \([fixedNodes]\) = \( \{ \} \)  \( \triangleright \) initialize empty list
3:  repeat
4:     \( x \leftarrow colGenAndSep(fixedNodes) \)  \( \triangleright \) generate LP solution
5:     if \( x \) is fractional then
6:         \([fixedNodes]\) \( \leftarrow \) new\( Node \)  \( \triangleright \) fix a new node
7:     end if
8:  until \( x \) is integer or infeasible
9:  return \( x \)
10: end procedure
5.4. Passenger routing

One of the main overall objectives of the model is to improve the passenger travel time (PTT). So far, the method minimizes the length of the train paths. This avoids extra additional dwelling of the trains at the stations and allows passengers traveling in the train to reach their destination fast. However, many passengers are required to transfer between trains to reach their destinations. Therefore, minimizing these transfer times becomes part of the overall objective of optimizing the passenger travel time.

The first step for calculating the passenger travel time in the network is defining the routes (i.e., train combinations) that each passenger can choose to travel from its origin station to its final station.

Once an integer feasible solution has been found, a graph is created representing all the train paths that form it. The same graph representation is used as the initial Train graph formulation described at the beginning of Section 3.2. Each train path’s arriving and departing times are connected with arcs only linking those stations where the train stops (i.e., where passengers can board or leave the train). The cost of those arcs is the time interval between the two nodes (i.e., $\Delta(w,u)$). There is an artificial source node for each station which is connected to the departure time nodes of the trains stopping in that station. Analogously, there is an artificial sink node for each station which is connected with the arrival time nodes of the trains stopping in that station. Therefore, it is assumed that all passengers arrive at their origin station at the exact time their train departs. For each station, transfer arcs are created connecting the arrival of a train with the departure of another with the related time interval as cost. It is assumed that from a train it is possible to transfer to any other train in the station. In this case, a minimum transfer time of 5 minutes is defined as a rule of thumb, meaning that if the time difference between the arrival of one train and the departure of another is lower, the transfer time corresponds to the time interval plus $|T|$. Once the graph is built, the route of the passenger can be computed as the shortest path from the artificial source node of the origin station to the artificial sink node of the destination station and the total travel time is directly given by the sum of costs of the arcs. This method is based on studies such as the ones proposed by Schöbel and Scholl (2006) and Rezanova (2015).

The total passenger travel time is finally estimated based on an OD matrix that indicates the number of passengers traveling between each pair of stations in the rush hour. The matrix only considers passengers traveling between stations in the network, not passengers from stations outside the network (i.e., people entering the network from Germany or cities in Jutland).

The main objective of the algorithm is to minimize the PTT. Therefore, every time a solution is computed, its PTT is compared with the best one found so far and updated if the new one is better. The process keeps iterating until the time limit is reached and the whole process is summarized in Algorithm 3.

Finally, in order to analyze the quality of the solution, this is compared with a lower bound solution. The lower bound (LB) value for the total path lengths is computed as the LP solution value at the root node. This is the equivalent to the shortest paths given by the graphs. In the case of the LB for the PTT, all transfers times between trains at the stations are equal to the minimum transfer time allowed.

6. Computational results

This section covers all the computational results obtained. First, the instances tested are defined. Then, all the results are described and the algorithm’s performance is discussed.
**Algorithm 3** Algorithm pseudo-code

1: procedure Algorithm()
2: \(x^b = \{}\) \(\triangleright\) initialize best solution
3: \(c(x^b) = \infty\)
4: repeat
5: \(x \leftarrow \text{diveHeuristic}()\) \(\triangleright\) generate a solution
6: if \(x\) is feasible then
7: if \(c(x) < c(x^b)\) then \(\triangleright\) compare passenger travel time
8: \(x^b = x\)
9: end if
10: end if
11: until time limit
12: return \(x^b\)
13: end procedure

### 6.1. Instances
A number of instances are created based on the basic case provided by the data from DSB. By changing the following three parameters, a total of 14 instances are obtained.

- \(HW_k\): Minimum headway between consecutive arrivals and departures at København H. This station is seen as one of the most congested stations in the network where all lines stop at and, therefore, the headway at this station becomes interesting to analyze individually. This parameter measures in minutes the minimum interval between consecutive arrivals or departures at København H in the same track segment.

- \(HW_n\): Minimum headway between consecutive arrivals and departures at any station in the network. This parameter measures in minutes the minimum interval between consecutive train arrivals or departures at each track segment and station in the network.

- \(HW_f\): Minimum headway between consecutive departures of Frequency trains in the same direction from common stations. The pair of Frequency lines may have slightly different stopping patterns or running and dwell times. This makes impossible to separate both train paths exactly half an hour during their entire trip. Therefore, a lower bound is needed that should be respected in any station. In this case, a minimum headway is defined for the consecutive departures from each station.

### 6.2. Model performance
Due to the large amount of parameter setting combinations, a base case is defined with the minimum values of each parameter. Then, each parameter is tested independently keeping the others fixed. The parameter values for the base case are shown in Table 2. All instances are tested with a maximum dwell time of 3 minutes at each station and a maximum symmetry gap of ±1.5 minutes.

| \(HW_k\) | \(HW_n\) | \(HW_f\) |
|---|---|---|
| 3 | 3 | 15 |

Table 2: Base case parameter setting
The algorithm is run 10 times for each scenario and the average values are calculated. The time limit for each algorithm run is set to 1 hour. The model has been entirely written in Julia language (Bezanson et al., 2017), modelled using JuMP (Lubin and Dunning, 2015) and using CPLEX v. 12.7 as the solver. It has been tested in a Intel Xeon Processor X5550 (quad-core, 2.66 GHz) using one thread.

Table 3: Average performance of the algorithm for different values of \( HW_k \), \( HW_n \) and \( HW_f \)

| HW_k (min) | Best PTT gap (%) | Avg PTT gap (%) | Avg path lengths (%) | AlgIters | Avg DiveIters | Avg SepIters | Avg CGIters | Avg columns | Avg added rows (%) | Avg RMP time (%) | Avg PP time (%) | Feasibility rate (%) |
|------------|------------------|-----------------|---------------------|----------|---------------|-------------|-------------|--------------|------------------|----------------|---------------|---------------------|
| 3          | 1.44             | 2.06            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 4          | 1.45             | 2.10            | 0.91                | 36.8     | 8.9           | 29.6        | 321.3       | 1027.2       | 5.5              | 24.7           | 65.0          | 29                  |
| 5          | 2.38             | 2.72            | 1.91                | 20.5     | 8.2           | 25.2        | 430.8       | 1719.7       | 5.2              | 72.8           | 24.2          | 31                  |
| 6          | 2.38             | 2.72            | 1.91                | 20.5     | 8.2           | 25.2        | 430.8       | 1719.7       | 5.2              | 72.8           | 24.2          | 31                  |

| HW_n (min) | Best PTT gap (%) | Avg PTT gap (%) | Avg path lengths (%) | AlgIters | Avg DiveIters | Avg SepIters | Avg CGIters | Avg columns | Avg added rows (%) | Avg RMP time (%) | Avg PP time (%) | Feasibility rate (%) |
|------------|------------------|-----------------|---------------------|----------|---------------|-------------|-------------|--------------|------------------|----------------|---------------|---------------------|
| 3          | 2.01             | 2.06            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 4          | 2.01             | 2.06            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 5          | 2.01             | 2.06            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 6          | 2.01             | 2.06            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |

| HW_f (min) | Best PTT gap (%) | Avg PTT gap (%) | Avg path lengths (%) | AlgIters | Avg DiveIters | Avg SepIters | Avg CGIters | Avg columns | Avg added rows (%) | Avg RMP time (%) | Avg PP time (%) | Feasibility rate (%) |
|------------|------------------|-----------------|---------------------|----------|---------------|-------------|-------------|--------------|------------------|----------------|---------------|---------------------|
| 15         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 16         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 17         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 18         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 19         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 20         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 21         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 22         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 23         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 24         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 25         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 26         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
| 27         | 1.34             | 2.41            | 0.41                | 63.6     | 6.2           | 24.3        | 105.9       | 827.5        | 5.6              | 14.1           | 66.1          | 40                  |
Figure 10: Average solution values for each scenario
Table 3 shows the results for each of the scenarios created by parameters $HW_k$, $HW_n$ and $HW_f$ respectively. The first column indicates the parameter value of the scenario. The second and third column displays the best and average solution values of PTT respectively found across the 10 runs which are compared to the lower bound defined at the end of Section 5.4. The fourth column indicates the average sum of path lengths related to the best solutions compared to the lower bound defined at the end of Section 5.4. The fifth column displays the number of algorithm iterations or equivalent repetitions of lines 5-10 in Algorithm 3 done per 1h run. The next three columns indicate the internal average iterations per algorithm iteration. First, the number of dive heuristic iterations which can be interpreted as the number of branches performed (i.e. nodes fixed). Next, the number of times the current LP solution is checked for violated constraints and, finally, the number of column generation iterations. The ninth and tenth columns shows the average number of columns and additional rows needed per algorithm iteration respectively. The eleventh and twelfth column indicate the proportional amount of time spent solving the RMP and PP respectively in relation to the total amount of time spent finding a solution. Last, the feasibility rate is stated that displays the proportion of algorithm iterations that result in a feasible integer solution. In order to have a better overview of the performance, the average solution values are displayed in Figure 10. Likewise, Figure A.14 in the Appendix displays the best solution values across the 10 runs.

The algorithm finds near optimal results both in PTT and path lengths in a reasonable amount of time for most of the scenarios. It can be noticed that when starting to increase the $HW_k$ and $HW_n$ values, the solution actually improves even though the solution space is reduced. This is counterintuitive and a perfect algorithm should not display this behaviour. There could be two main reasons for this behaviour. 1) the minimum transfer time is set to 5 minutes and this may cause certain headway values to perform better. Table A.4 in the Appendix shows that a lower minimum transfer time results in objective values that more closely follow an increasing pattern. 2) the chosen branching strategy is randomized but the set of variables that can be branched on is limited to a subset of all the potential variables (see Section 5.3). This may imply that there are solutions that cannot be reached. Table A.5 in the Appendix shows that when branching on random fractional nodes, the results again follow an increasing pattern more closely. However, on average, the solution quality decreases with this setting.

In terms of speed, it can be seen that the problem becomes harder to solve when increasing the parameter values. In particular, for high $HW_k$ and $HW_n$ values, the LP becomes very hard to solve. Nevertheless, the algorithm is able to find solutions for $HW_k = 6$ minutes which is the maximum possible as 10 trains arrive per hour in København H through the same corridor. Also, solutions are found for values up to $HW_n = 5$ minutes and higher values were not further tested as they do not seem realistic for the network studied. Moreover, the algorithm finds solutions for $HW_f = 27$ minutes which seems to be the maximum allowed due to the differences in running times of the pairs of frequency lines.

As future work, it would be interesting to study how to decrease the time for solving the LP relaxation of the master problem as this can become quite excessive for the more constrained instances. One may attempt to 1) leave the headway constraints out of the initial formulation and add the violated ones by separation or 2) attempt to stabilize the dual variables in the column generation algorithm (see e.g. Du Merle et al. (1999) or Oukil et al. (2007)).

When looking at the number of columns needed per iteration, it is also interesting to look from which lines the columns mainly come from. Figure 11 shows the average distribution of columns grouped per lines using or not the quadruple-track segment for the base case. Most of the columns belong to lines using the quadruple-track segment. Allowing two routes for the
trains doubles the number of possible columns that can be generated. It should also be noted that 32% of the total amount of columns belong to the two lines running until Kalundborg. This is related to the fact that at the single-track segment of this corridor, is the only place where a crossing between trains of different lines can occur. In order to cross, one of the trains needs to dwell for three minutes in one of the stations resulting in a poor path length. As the crossing constraints are added by separation, this results in a larger amount of columns generated.

The model is able to route the passengers realistically. This is analyzed using graphical tools such as the one shown in Figure A.13 which shows the passenger flow between trains at København H for an example solution. Nevertheless, a more realistic routing of the passengers in the most congested areas can help to have a complete perspective of the trips of the passengers and the occupancy of the trains. This can be further improved by taking train capacity into account (Rezanova, 2015) or achieving a more accurate estimation of the passenger demand. Although the fixed running times between stations simulate realistic cases to a large extent, considering variable running times at the track segments can increase significantly the solution space. However, the complexity of the model would increase accordingly. Also, considering different types of headway along the network allows a better utilization of the track capacity as more trains can be scheduled per corridor (Liu and Han, 2017).

Different graphical tools have been used to analyze the potential additional conflicts of a timetable such as the one in Figure A.12 which shows an example graphic timetable for the north-western corridor between København H and Kalundborg. Routing the trains at a more detailed level at some stations can allow having completely conflict-free solutions in the network. Currently, feasibility issues may arise from the model due to track-crossing conflicts at some stations where corridors join. This can be solved by adding additional graph nodes to model the track junctions. Likewise, turnaround times for trains at the end of stations can be enforced by removing the conflicting arcs in the graph. This can potentially lead to a better utilization of the rolling stock.

A variation of the solution method is proposed in Martin-Iradi (2018) where the best transfers of each solution are prioritized to be included in the next solution, transforming the algorithm into a Large Neighbourhood Search (LNS) meta-heuristic. However, the solution quality does not seem to improve. Nevertheless, it would be interesting to study alternative methods of rebuilding the solution efficiently.

7. Conclusion

In this study, the railway timetable generation process has been optimized from a passenger perspective. A model has been implemented to solve the network for Regional and InterCity trains in Zealand.
The model is based on a graph formulation that takes advantage of the symmetric timetabling strategy and the assumed fixed train running times between stations. As a result, all the required train paths for a line in a cycle time of one hour can be computed by a single shortest path. Furthermore, the algorithm relies mainly on both column generation and constraint separation techniques. This, combined with heuristic methods to achieve an integer solution, can be transformed in an iterative process that seeks to find multiple solutions.

The model has been shown to find good solutions to the network in a relatively fast time. It allows increasing the minimum headway easily along the network, achieving more robust timetables, without a significant detriment in time or solution quality. The model can potentially be improved and be implemented as a useful tool in the planning process of any train operating company.

Acknowledgements

The work of Stefan Ropke was funded by the Innovation Fund Denmark under the IPTOP project, this support is gratefully acknowledged. The authors are grateful to Federico Farina for his valuable comments and useful discussions. Moreover, thanks are due to Esben Linde from DSB for having provided relevant data for the instances presented in the paper.

References

Bezanson, J., Edelman, A., Karpinski, S., Shah, V. B., 2017. Julia: A fresh approach to numerical computing. SIAM review 59 (1), 65–98.
URL https://doi.org/10.1137/141000671

Borndörfer, R., Hoppmann, H., Karbstein, M., 2017. Passenger routing for periodic timetable optimization. Public Transport 9 (1-2), 115–135.

Brännlund, U., Lindberg, P., Nou, A., Nilsson, J.-E., 1998. Railway timetabling using Lagrangian relaxation. Transportation Science 32 (4), 358–369.

Bussieck, M. R., Winter, T., Zimmermann, U. T., Oct 1997. Discrete optimization in public rail transport. Mathematical Programming 79 (1), 415–444.
URL https://doi.org/10.1007/BF02614327

Cacchiani, V., Caprara, A., Toth, P., Jun 2008. A column generation approach to train timetabling on a corridor. 4OR 6 (2), 125–142.
URL https://doi.org/10.1007/s10288-007-0037-5

Cacchiani, V., Caprara, A., Toth, P., 2010a. Non-cyclic train timetabling and comparability graphs. Operations Research Letters 38 (3), 179 – 184.
URL http://www.sciencedirect.com/science/article/pii/S0167637710000088

Cacchiani, V., Caprara, A., Toth, P., 2010b. Scheduling extra freight trains on railway networks. Transportation Research Part B: Methodological 44 (2), 215 – 231.
URL http://www.sciencedirect.com/science/article/pii/S0191261509000812

Cacchiani, V., Caprara, A., Toth, P., 2013. Finding cliques of maximum weight on a generalization of permutation graphs. Optimization Letters 7 (2), 289–296.

Cacchiani, V., Toth, P., 2012. Nominal and robust train timetabling problems. European Journal of Operational Research 219 (3), 727–737.

Caprara, A., Fischetti, M., Toth, P., 2002. Modeling and solving the train timetabling problem. Operations Research 50 (5), 851–861.

Caprara, A., Kroon, L., Monaci, M., Peeters, M., Toth, P., 2007. Chapter 3 passenger railway optimization. In: Barnhart, C., Laporte, G. (Eds.), Transportation. Vol. 14 of Handbooks in Operations Research and Management Science. Elsevier, pp. 129 – 187.
URL http://www.sciencedirect.com/science/article/pii/S09270507066140037

Caprara, A., Monaci, M., Paolo Toth, P., Guida, F. L., 2006. A Lagrangian heuristic algorithm for a real-world train timetabling problem. Discrete Applied Mathematics 154 (5), 738 – 753. iV ALIO/EURO Workshop on Applied Combinatorial Optimization.
URL http://www.sciencedirect.com/science/article/pii/S0166218X05003045
Carey, M., 1994. Extending a train pathing model from one-way to two-way track. Transportation Research, Part B (methodological) 28B (5), 395–400.
URL https://doi.org/10.1057/jors.1995.136

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

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

Dijkstra, E. W., 1959. A note on two problems in connexion with graphs. Numerische Mathematik 1 (1), 269–271.

DSB, 2018. Langsigtet Planligning.

Du Merle, O., Villeneuve, D., Desrosiers, J., Hansen, P., 1999. Stabilized column generation. Discrete Mathematics 194 (1-3), 229–237.

Farina, F., 2019. A heuristic for periodic train timetabling with integrated passenger routing. submitted to European Journal of Operational Research.

Gattermann, P., Großmann, P., Nachtigall, K., Schöbel, A., 2016. Integrating passengers’ routes in periodic timetabling: A sat approach. In: 16th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2016). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Hansen, I. A., 2009. Railway network timetabling and dynamic traffic management. In: 2nd International Conference on Recent Advances in Railway Engineering, ICRAE-2009, Tehran, IR Iran, Sept. 27-28, 2009.

Harrod, S., 2012. A tutorial on fundamental model structures for railway timetable optimization. Surveys in Operations Research and Management Science 17 (2), 85–96.

Jovanovic, D., Harker, P. T., 1991. Tactical scheduling of rail operations - the SCAN-I system. Transportation Science 25 (1), 46–64.

Kinder, M., 2008. Models for periodic timetabling. Technische Universität, Berlin.

Liebchen, C., 2007. Periodic timetable optimization in public transport. In: Waldmann, K.-H., Stocker, U. M. (Eds.), Operations Research Proceedings 2006. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 29–36.

Liebchen, C., Möhring, R. H., 2002. A case study in periodic timetabling. Electronic Notes in Theoretical Computer Science 66 (6), 18–31.

Liu, P., Han, B., 2017. Optimizing the train timetable with consideration of different kinds of headway time. Journal of Algorithms & Computational Technology 11 (2), 148–162.
URL https://doi.org/10.1177/1748301816689685

Lucas, M., Dunning, I., 2015. Computing in operations research using julia. INFORMS Journal on Computing 27 (2), 238–248.

Lusby, R. M., Larsen, J., Ehrcott, M., Ryan, D., 2011. Railway track allocation: Models and methods. Or Spectrum - Quantitative Approaches in Management 33 (4), 843–883.

Martin-Iradi, B., 2018. Optimization in railway timetabling for regional and intercity trains in zealand.

Min, Y. H., Park, M. J., Hong, S. P., Hong, S. H., 2011. An appraisal of a column-generation-based algorithm for centralized train-conflict resolution on a metropolitan railway network. Transportation Research Part B: Methodological 45 (2), 409 – 429.
URL http://www.sciencedirect.com/science/article/pii/S0191261510001025

Nachtigall, K., 1998. Periodic network optimization and fixed interval timetables. Deutsches Zentrum für Luft-und Raumfahrt, Institut für Flugführung, Braunschweig.

Odijk, M. A., 1996. A constraint generation algorithm for the construction of periodic railway timetables. Transportation Research Part B-methodological 30 (6), 455–464.

Ouikl, A., Amor, H. B., Desrosiers, J., El Gueddari, H., 2007. Stabilized column generation for highly degenerate multiple-depot vehicle scheduling problems. Computers & Operations Research 34 (3), 817–834.

Peeters, L., 2003. Cyclic railway timetable optimization = optimalisatie van cyclische spoorwegdienstregelingen. Ph.D. thesis, ETH Zurich, diss. Univ. Rotterdam, 2003.

Rezanova, N. J., 2015. Line planning optimization at dsb. In: 13th Conference on advanced systems in public transport, Erasmus University.

Schöbel, A., Scholl, S., 2006. Line Planning with Minimal Traveling Time. In: Kroon, L. G., Möhring, R. H. (Eds.), 5th Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS’05). Vol. 2 of OpenAccess Series in Informatics (OASIcs). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany.
URL http://drops.dagstuhl.de/opus/volltexte/2006/660

Serafini, P., Ukovich, V., Nov. 1989. A mathematical model for periodic scheduling problems. SIAM J. Discret. Math. 2 (4), 550–581.
URL http://dx.doi.org/10.1137/0402049

Szpigel, B., 1973. Optimal train scheduling on a single track railway. Proceedings of the Ifors Conference Operational Research ’72, 343–52, 343–352.

26
Appendix A.

Figure A.12: Timetable of the lines running through the North-West corridor
Table A.4: Average performance of the algorithm for different values of $HW_k$, $HW_n$ and $HW_f$ with a minimum transfer time of 30 s.

| $HW_k$ (min) | Best PTT gap (%) | Avg PTT gap (%) | Avg path lengths gap (%) | Alg Iters | Avg Dive Iters | Avg Sep Iters | Avg CG Iters | Avg columns | Avg added rows (%) | Avg RMP time (%) | Avg PP time (%) | Feasibility rate (%) |
|-------------|-----------------|-----------------|--------------------------|---------|---------------|--------------|--------------|-------------|-------------------|----------------|--------------|----------------------|
| 3           | 6.21            | 6.45            | 0.17                     | 67.6    | 6.2           | 24.5         | 170.0        | 423.5       | 7.6               | 15.3           | 67.5          | 48                   |
| 4           | 6.05            | 6.42            | 0.39                     | 53.7    | 7.3           | 25.0         | 217.4        | 611.4       | 5.9               | 18.8           | 67.6          | 42                   |
| 5           | 6.02            | 7.05            | 0.77                     | 35.3    | 8.6           | 28.8         | 301.5        | 976.4       | 5.3               | 26.3           | 63.5          | 34                   |
| 6           | 6.61            | 7.11            | 0.86                     | 17.4    | 7.9           | 24.8         | 377.4        | 1571.0      | 5.2               | 60.5           | 35.0          | 28                   |

Table A.5: Average performance of the algorithm for different values of $HW_k$, $HW_n$ and $HW_f$ with a random branching strategy.

| $HW_k$ (min) | Best PTT gap (%) | Avg PTT gap (%) | Avg path lengths gap (%) | Alg Iters | Avg Dive Iters | Avg Sep Iters | Avg CG Iters | Avg columns | Avg added rows (%) | Avg RMP time (%) | Avg PP time (%) | Feasibility rate (%) |
|-------------|-----------------|-----------------|--------------------------|---------|---------------|--------------|--------------|-------------|-------------------|----------------|--------------|----------------------|
| 3           | 6.10            | 6.31            | 0.38                     | 68.6    | 6.2           | 24.4         | 170.0        | 424.0       | 7.6               | 15.3           | 67.5          | 48                   |
| 4           | 6.01            | 6.20            | 0.66                     | 36.2    | 8.5           | 32.9         | 263.7        | 807.9       | 7.0               | 42.6           | 47.5          | 44                   |
| 5           | 6.02            | 6.85            | 0.77                     | 17.4    | 7.9           | 24.8         | 377.4        | 1571.0      | 5.2               | 60.5           | 35.0          | 28                   |
| 6           | 7.90            | 6.34            | 1.30                     | 6.0     | 8.1           | 33.0         | 693.0        | 3146.8      | 16.4              | 94.8           | 4.2           | 7                    |

| $HW_n$ (min) | Best PTT gap (%) | Avg PTT gap (%) | Avg path lengths gap (%) | Alg Iters | Avg Dive Iters | Avg Sep Iters | Avg CG Iters | Avg columns | Avg added rows (%) | Avg RMP time (%) | Avg PP time (%) | Feasibility rate (%) |
|-------------|-----------------|-----------------|--------------------------|---------|---------------|--------------|--------------|-------------|-------------------|----------------|--------------|----------------------|
| 3           | 6.10            | 6.45            | 0.17                     | 71.7    | 6.2           | 24.3         | 169.2        | 422.8       | 7.6               | 14.3           | 68.4          | 49                   |
| 4           | 6.02            | 6.62            | 0.38                     | 70.9    | 6.1           | 25.3         | 179.3        | 537.7       | 8.0               | 18.8           | 67.4          | 45                   |
| 5           | 6.10            | 6.43            | 0.44                     | 74.0    | 7.2           | 24.6         | 168.2        | 466.5       | 8.7               | 13.1           | 68.1          | 46                   |
| 6           | 6.79            | 6.83            | 0.61                     | 61.2    | 7.2           | 24.7         | 194.6        | 552.2       | 9.8               | 14.6           | 67.3          | 46                   |
| 7           | 6.06            | 6.34            | 0.26                     | 81.2    | 6.0           | 20.6         | 139.5        | 410.9       | 9.0               | 15.4           | 63.8          | 65                   |
| 8           | 6.07            | 6.68            | 0.62                     | 73.2    | 6.3           | 20.8         | 152.7        | 456.5       | 10.2              | 15.0           | 64.2          | 45                   |
| 9           | 6.45            | 6.92            | 0.58                     | 98.9    | 3.9           | 14.4         | 108.7        | 312.5       | 10.5              | 13.0           | 62.5          | 9                    |

| $HW_f$ (min) | Best PTT gap (%) | Avg PTT gap (%) | Avg path lengths gap (%) | Alg Iters | Avg Dive Iters | Avg Sep Iters | Avg CG Iters | Avg columns | Avg added rows (%) | Avg RMP time (%) | Avg PP time (%) | Feasibility rate (%) |
|-------------|-----------------|-----------------|--------------------------|---------|---------------|--------------|--------------|-------------|-------------------|----------------|--------------|----------------------|
| 15          | 1.83            | 2.13            | 0.42                     | 63.6    | 6.9           | 25.2         | 197.0        | 527.3       | 7.0               | 19.7           | 68.7          | 54                   |
| 3.5         | 1.80            | 2.15            | 0.64                     | 38.5    | 7.0           | 28.0         | 262.6        | 791.4       | 7.8               | 35.6           | 53.7          | 43                   |
| 4           | 1.89            | 2.34            | 0.54                     | 42.5    | 7.3           | 24.4         | 263.9        | 910.6       | 5.6               | 22.7           | 66.3          | 34                   |
| 6           | 2.51            | 2.86            | 1.12                     | 22.5    | 6.4           | 19.4         | 340.2        | 1445.9      | 4.6               | 55.1           | 39.9          | 32                   |

Table A.6: Average performance of the algorithm for different values of $HW_k$, $HW_n$ and $HW_f$ with a random branching strategy.
Figure A.13: Amount of passengers transferring between trains at København H
Figure A.14: Best solution values for each scenario