A Branch-and-Price Framework for Railway Rolling Stock Rescheduling During Disruptions

Haahr, Jørgen Thorlund; Lusby, Richard Martin ; Larsen, Jesper; Pisinger, David

Publication date:
2014

A Branch-and-Price Framework for Railway Rolling Stock Rescheduling During Disruptions

Jørgen T. Haahr (jhaa@dtu.dk)
Richard M. Lusby (rmlu@dtu.dk)
Jesper Larsen (jesla@dtu.dk)
David Pisinger (dapi@dtu.dk)

Technical University of Denmark
Department of Engineering Management
Produktionstorvet
Building 426
2800 Kgs. Lyngby
Denmark

Abstract

Rescheduling rolling stock during a disruption is a passenger railway optimization problem. In current practice this is typically optimized manually despite the high complexity and high runtime requirements of the task. In this paper we propose a path-based mathematical formulation that is solved using column generation in a complete Branch-and-Price framework. In contrast to flow-based approaches our formulation is more easily extended to handle certain families of constraints, such as train unit maintenance restrictions. We benchmark the framework against real-life instances provided by the suburban railway operator in Copenhagen (DSB S-tog). In combination with a lower bound method we show that near-optimal solutions can be found within a few seconds during a disruption. In addition we show that framework is also able to find solution within a few minutes for non-disturbed timetables.

Acknowledgements: The Danish Council for Strategic Research and DSB S-tog

1 Introduction

Rolling stock scheduling is a non-trivial task which must be addressed by railway operators on a regular basis. It is often hard just to find a feasible solution, i.e., an assignment of physical train units such that all train services are covered, depot capacities are respected, end-of-day depot balances are correct and such that several other business rules are satisfied. More than one feasible schedule might exist and it is important to find a good solution as rolling stock is one of the major expenses for a railway company. Rolling stock schedules are usually made months in advance by planners. These are then given to dispatchers who may make some changes or add details to the schedules a few days prior to operation. Rolling stock scheduling or circulation has been subject to much research over the last two decades which has advanced this research field. An approach by Schrijver [27] considers a model for determining the minimum number of units needed for a single train line. Fioole et al. [16] propose a richer model that includes train compositions, demand shortage, shunting movements and train splitting. More recently Nielsen et al. [22] adopt a version of this model for scheduling in real-time during disruptions. The majority of models and

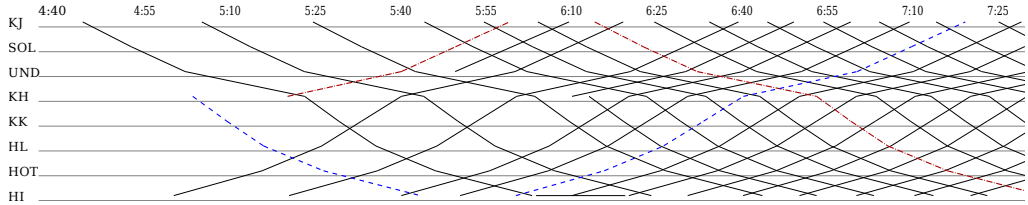

Figure 1: An example of a timetable for a single line with trips going between two end-stations. Two trip-sequences have been highlighted for illustrative purposes.

solution methods are focused on offline planning and are not capable of the real-time rescheduling needed for handling large disruptions.

Inevitably unplanned events occur on a daily basis that disturb the pre-planned operation. Depending on the severity a disrupted situation may occur rendering the rolling stock schedule infeasible. Examples of large disruptions include: passenger induced delays, rolling stock breakdown, signalling problems, and infrastructure damage or failure. Large disruptions occur multiple times per year and the situation requires a new schedule to be found immediately. The impact of a disruption may be hard to comprehend for rolling stock dispatchers and high quality recovery solutions may be very hard to find. If not handled promptly and thoroughly, the effects of the disruption can easily cascade and result in even more problems. Real-time rolling stock rescheduling is an emerging area of research that has received relatively limited attention in the literature [22, 18]. In recent years there has been a growing interest in this field but the topic still remains a challenge.

We present and benchmark a Branch-and-Price (B&P) framework for solving the Rolling Stock Rescheduling Problem (RSRP). Column generation is an interesting technique that has proven to work well with many problems. It is already the state of art for railway crew scheduling [24, 25] but applications in rolling stock rescheduling are limited [7, 23]. We present a unit-based rolling stock model that can be solved using delayed column generation. To the best of our knowledge such a model and approach have not been published before. A unit-based model presents a few advantages such as the ability to model maintenance constraints. The proposed framework is designed to work in a rolling time horizon scheme. Given a planned schedule and a point in time the framework can identify units in depots and in transit, and reschedule the foreseeable future. During disruption a new schedule must be found every time the timetable changes, e.g., due to new or updated information about the disruption.

The rest of the paper is structured as follows. A description of the problem is given in Section 2 followed by a literature review. In Section 3 the optimization model is presented and described. Afterwards in Section 4 the B&P framework is presented. Section 5 describes the test instances and shows the experimental results. Finally, the paper is concluded in section 6

2 Problem Description

2.1 Terminology

To the public a timetable is a chart stating departure and arrival times of all trains in the system. The timetable consists of a set of *lines*, where each line is a set of *train trips* which connect the same origin and destination pair and contain the same subset of intermediate stations. A limited number of trips on the same line can start or end at some intermediate station that is different to the origin and destination. Every train trip is an individual journey from one origin station to a destination station, and determines the departure times at every station from the origin to the target. A trip does *not*, however, include or determine which physical train unit should perform the given journey. Figure 1 shows a time-space diagram of a single line containing multiple trips. Multiple trip *legs* constitute a trip, where each trip leg refers to two consecutive

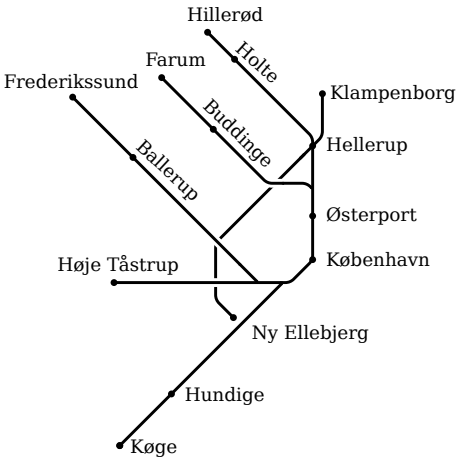

Figure 2: The S-tog railway infrastructure network. The depot- and end stations are highlighted.

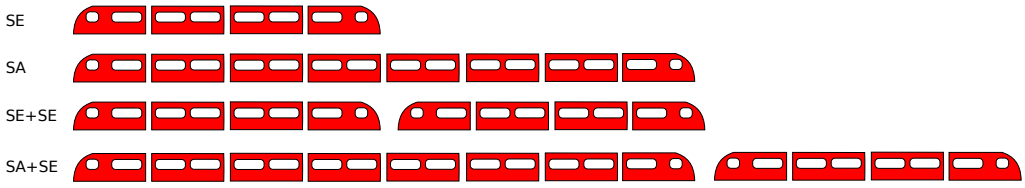

Figure 3: Examples of train compositions.

stations on a trip where the train stops. In other words, a trip consists of a sequence of connected legs. A *trip sequence* denotes a continuous series of trips that can be performed by the same physical train composition. That is, the same composition can be assigned to the whole set of trips in the trip sequence without going into or out of the depot or require any shunting activity. Trip sequences typically span a whole day and stay within a single line. Each line is serviced by a set of trip sequences and every trip, except *deadheading* trips, is assigned to a single trip sequence. Deadheading refers to an empty, non-revenue gathering train movement, which is used to re-position rolling stock.

In reality we need to assign train units to every leg in a timetable; however, such a fine level of detail is not needed as units are typically assigned to whole trips. In our approach this assignment is too coarse and we therefore introduce *subtrips* that cover the gap between trips and legs. A trip is partitioned into subtrips where each subtrip represents a sequence of connected legs. Furthermore, we partition the trips such that every origin and target station of a subtrip is a key station (depot- or end-station). Figure 2 illustrates the key stations in the S-tog network. Such a partition enables our mathematical model to make relevant decisions such as coupling, decoupling and turning. During disruptions a few additional key stations can appear since an infrastructure blockage may require trains to turn around at other stations.

A train unit is the smallest inseparable vehicle we consider, i.e. a set of wagons or cars which is not practical to separate on a day-to-day basis. In our study we only have one family of units that consists of two different types; this means that every unit can be coupled together with any other unit available, resulting in a longer train. Additionally, all train drivers can drive any type of train unit, and all trains are self-propelled, i.e., do not need a locomotive for pulling. The important types characteristics are the number of seats, the physical train length and the operational cost.

Units can be coupled together to form train compositions, where a composition is a set of units in a specific order. Figure 3 shows examples of a train compositions. In this work the compositions are implicitly determined by the individual assignment of train units to subtrips. We argue that the composition plays a smaller role in our study for two reasons. Firstly, compositions on trips

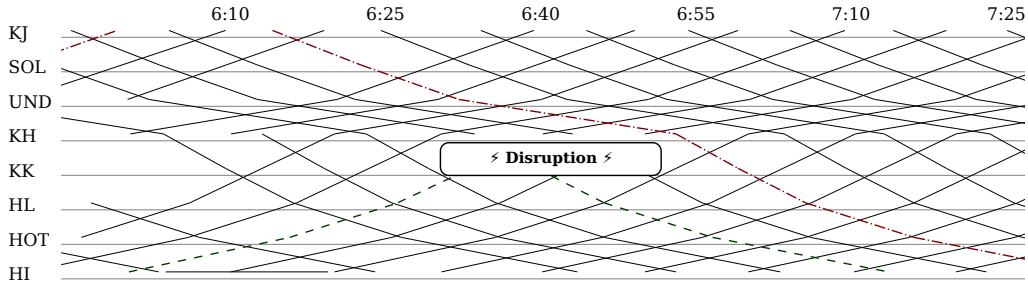

Figure 4: A disrupted timetable due to infrastructure blockage between station A and B. The updated timetable resolves the situation by turning the train services at station A and B.

typically range from one to two units in size at the Copenhagen Suburban Railway Operator (DSB S-tog), which means that we do not have any ordering conflicts when coupling or decoupling units. One uncommon composition with three units can occur, which respects the maximum length of the platforms on specific lines. However, in this composition all three units must be of the same smallest type. Secondly, we are primarily concerned with disruptions, which means that the ordering of individual units (in a composition of a train) for passenger convenience is not a governing priority. Given a planning period with a pre-specified start and end time, the rolling stock (re)scheduling problem entails assigning train units (i.e. rolling stock units) to every timetabled trip-leg for the given planning period. The assignment is subject to a number of natural constraints such as fleet capacity, train unit location and depot capacity. The problem is similar and comparable to the multi-commodity flow problem.

The infrastructure owner and railway operator might define a rolling stock schedule as the actual train-unit assignment of every departure. Note that we will also adopt an equivalent definition which we find more natural from a train-unit perspective: A rolling stock schedule is a chronological list of tasks performed by each train unit available. The possible tasks performed by a train unit are subtrips, decouplings, depot movement and couplings.

2.2 Disruption Management

Trains do unfortunately not always arrive and depart on time. This is due to unplanned events such as accidents, human mistakes, infrastructure failures, rolling stock breakdowns, and weather conditions. In this paper we consider events that cannot be absorbed by network buffers and will therefore lead to a disrupted situation on the network. In such cases, network operators must make changes to the timetable in order to avoid a total breakdown. As a consequence of timetable changes the rolling stock schedules and other planned activities may be rendered infeasible. Figure 4 shows an example of a disrupted line due to a blockage between two stations in both directions. Once an updated timetable has been decided upon by the infrastructure owner a new, candidate rolling stock schedule must be presented by the rolling stock dispatchers within short time. In reality the timetable and rolling stock schedule are not determined completely independently. The infrastructure owner cannot expect to carry out a timetable that is infeasible with respect to rolling stock, and they are also not interested in cancelling more train departures than necessary. Thus, when rescheduling rolling stock it makes sense to integrate timetable decisions to some degree.

During a disruption the situation can be quite chaotic; it must be resolved as quickly as possible consequently giving all actors little time to reschedule plans. In our case study (and probably in many other cases) the infrastructure owners have exclusive control over all train activities while the train operators have a supporting role. The infrastructure owners respond to a disruption quickly by making train service cancellations or alterations. Predefined emergency plans exist and are used as guides for handling the disruption. Their immediate goal is to avoid delay propagation and further escalation. Depending on the time of the day and the location of the disruption, the

available time before a decision must be made ranges from 3 to 20 minutes. Groth et al. [17] present an in-depth description of the disruption management process. A temporary timetable is formed and the operators must try to reassign train units according to the new disrupted timetable. This can become a non-trivial task and the new plans may be far from optimal since the issues are resolved sequentially by degree of severity and priority using rules of thumb while only considering the immediate future. Train units of cancelled train services are e.g. usually put into the nearest depot without considering the long-term consequences. After the disruption is resolved the original timetable is restored by the infrastructure owners, and (in our case study) the railway operator has, by contract, a 60 minutes time limit to recover the rolling stock (and all other) schedules. It may however be impossible to cover all trains at this point in time due to the effects of the disruption. The network operator must thus be able to find new plans quickly both during and after the disruption. This highly motivates the use of a computer-aided decision support system.

Computer-aided decision support systems are already used by DSB S-tog during the different planning phases for determining the rolling stock schedules. However, the system tends to have a higher level of abstraction and requires more runtime than is available during a disruption. In the planning phases, anonymous units, whose initial positions can be freely chosen, are routed. For detailed real-time planning we need to adhere to the current positions of the units and we also need to consider individual unit restrictions. Significant unit properties such as time or mileage limits – due to maintenance requirements – should be respected. Other less critical unit-specific properties are even interesting to consider in a planning context, e.g., paths for trains equipped to carry bikes or trains covered by commercials.

Increasing the desired level of detail typically hinders the runtime performance. Sacrifices must be made in order to achieve desirable runtimes. During a disruption the rolling stock dispatchers allow themselves to relax some of the *desirable* and non-critical business rules, e.g., they might be willing to do a few odd or unplanned shunting operations if this resolves the problem. Furthermore, the exact position of a train unit in a train composition is not so critical due to the fact that units are bi-directed, self-propelled and because the compositions are not very long. Ignoring the exact compositions of trains may pose more of a problem for some other railway operators, see Fioole et al. [16].

2.3 Objectives

The primary objective is to find any feasible schedule; however, if several exist, then some schedules will clearly be more attractive than others. Therefore, in this work the following secondary objectives are included: the number of trip cancellations, robustness notions, seat shortages, end-of-day depot balances and operational costs.

Trip cancellations are the most expensive component as these contribute to the reliability performance of the railway operator. The price-function is a bit complex since a fine is issued if the reliability percentage falls below a certain contracted threshold, and secondly the size of the fine depends on the deviation of the reliability. There is a similar contract for punctuality, but since we do not consider the timetable planning, this detail is not pursued further.

The concept of robustness is often ambiguous and can include several things. In this paper we do not decide on arrival or departure times – the robustness can only be affected by the choice of assigned units paths. Couplings and decouplings are performed many times daily but, although necessary, these operations reduce overall robustness. A coupling or decoupling can easily take more time than expected due to manual interaction and unforeseen technical difficulties. In addition, it also requires coordination with a second train or a depot driver. Keeping the number of couplings to a minimum and performing the couplings outside critical time periods is important. Reserving enough time for (de)couplings can also be considered as a robustness indicator, see Cardaso et al. [10] for more consideration on robustness.

The railway operator is obliged to satisfy the actual demand on a trip by supplying enough seats to cover some percentage of the passengers. Therefore, it is not only necessary to cover a trip but also to assign units, i.e. a composition, that collectively provide enough seats. Interestingly,

seat shortages can also result in a lower robustness. Consider the case where a train arrives at a platform crowded with people. If the train is small, i.e. the number of total seats is too low, then passengers need extra time to get on and off the train, perhaps causing an unforeseen delay.

The end-of-day depot balance is important for continued operation the next day since a certain number of trains must be available at each depot the following morning. If the balance is off, then unplanned deadheading trips will be necessary to execute the plan the next day of operation. Deadheading train units is undesirable due to the expense of running rolling stock units and hiring a driver to move the train.

The operational cost of driving a train mainly consists of power consumption, wear and tear, infrastructure allocation, cleaning, and driver related expenses. The first two are linear with respect to the distance travelled and the other costs are more or less fixed depending on whether the trip has been cancelled or not. The infrastructure is already reserved and it has already been decided how many drivers are working on the day of operation. Cleaning costs are constant as the unit utilization is close to 100% every day.

The number of cancellations is usually (in literature and at DSB S-tog) the most important objective and is therefore given a cost that is significantly higher than the other objectives. The seat and operational objectives are conflicting and a non-trivial trade-off is sought. The network operator is obliged to meet the passenger demand, but not at all costs, e.g., the operator would not couple a rolling stock unit to a train service only to provide few additional passenger seats. In the benchmarks we will investigate this trade-off and try to find a sensible balance. Note, that this effectively undermines the importance of finding optimal solutions in contrast to some good solutions. Finally, the end-of-day balance is relevant when handling disruptions, as this may obscure the fleet position at the end of operations. A sufficient number of train units must be present at specific depots for the following day. A mismatch is corrected by performing deadheading trips, e.g., during the night.

For the purpose of testing our solution method we will not introduce a more complicated objective for disruptions, see [22, 25] for a adopting a separate objective for disruptions that minimizes deviation from the original plan. In contrast to crew scheduling, identical rolling stock units can be interchanged without substantial complications w.r.t. the trip coverage, seat shortage and operational expense. We penalize the end-of-day deviations, but we will not consider penalizing missed or unplanned shunting operations (in addition to the normal cost) during disruptions. Changing plans are understandably a nuisance that requires communication and coordination. However, with the recent advancements in mobile technology we believe that this is a diminishing issue.

2.4 Literature

Most literature related to passenger railway optimization (from an operators perspective) can be categorized by the problem they address, e.g., timetable planning, rolling stock scheduling, crew scheduling, or shunting. In this section we limit our discussion to the somewhat more scarce literature related to rolling stock scheduling with an emphasis on real-time disruption management. We refer to Caprara et al. [13] for an overview of different railway optimization problems, and Kroon and Huisman [20] or Jespersen-Groth [19] for an overview in a disruption management context. In this paper we consider self propelled trains units that do not require a powered locomotive for pulling. Several papers exist that consider the locomotive and carriages problem, see e.g. [14, 21, 1].

The rolling stock problem has been studied in a variety of different settings using different mathematical models and targeted time horizons. A good reference for the rolling stock scheduling problem is the one used by the Dutch railway company Nederlandse Spoorwegen (NS), described by Fioole et al. [16]. The authors present the problem of assigning train unit compositions to every departure and arrival. The main goal is to cover every train trip while respecting several constraints such as fleet capacity, composition change restrictions and allowing trains to be combined and split.

The strategic or tactical rolling stock problem has been studied in various forms over the past two decades. The goal is to determine the fleet size or to find a feasible rolling stock schedule (or circulation).

An early approach was studied by Schrijver [27], where the aim is to minimize the number of required trains to satisfy passenger demand. This approach does not consider the train compositions, composition changes, depot parking nor maintenance. A small data-instance from NS is presented and can be solved within seconds.

Alfieri et al. [3] consider a richer rolling stock model that considers the compositions of train arrivals and departures. The model assumes a single line and one family of compatible train units. They are able to solve a generic workday from NS for a line within 1-2 hours.

Peeters and Kroon [23] study a similar setting also spanning multiple lines using one family of train unit types. They present a B&P method which outperforms a commercial Mixed Integer Program (MIP) solver. Computation times are less than a minute for the larger data instance consisting of three lines. A few what-if scenarios are considered that quantify the effect of changing the fleet composition (and platform lengths) and allowing shunting at a new station.

Fioole et al. [16] study an extension of the same model that allows combining and splitting trains. They consider a subset of lines from NS and solve the problem as a MIP using a commercial solver. In addition a heuristic based on the Linear Program (LP) relaxation is used. Good quality solutions are found within hours. The solution model was used as a supportive tool for generating the Dutch timetable for 2005.

Cadarso et al [11] consider an approach similar to Fioole et al. [16] which additionally tries to optimize the robustness of the rolling stock schedule while penalizing high densities of standing passengers. Deadheading train units is allowed in order to increase capacity. Robustness is modelled by penalizing the total number of coupling operations, and by giving shunting operations during rush hours a special penalty. They consider two data instances, a single day of a single line from RENFE with 4 depot stations and 320 train services, and two lines with 9 depot stations and 400 train services. Optimal solutions are generated within minutes using a commercial MIP solver.

A two-stage optimization model is presented by Cacchiani et al. [9] for solving the rolling stock scheduling problem with a set of generated failure scenarios. The model is similar to one presented by Fioole et al. [16] but without minimum shunting time, combining and splitting of trains. A full MIP is presented and solved heuristically using a Benders Decomposition method. The data instance is a one day of single line from NS with up to 400 trips and 28 failure scenarios. The solution method is able to find good solutions within 2 minutes.

Several studies consider a short-term rolling stock problem aiming at rescheduling a few days before the day of operation. Compared to a long-term model such approaches have to deal with a greater level of detail and cannot assume a clean-slate approach as the availability of the different resources is fixed.

Budai et al. [8] consider a rebalancing problem where the goal is to minimize inventory end-of-day deviations from the original plan. They present two very fast heuristics methods which aim to improve existing schedules. They prove that the rebalancing problem is NP-hard.

Other related research for short-term scheduling with a slightly different problem definition also exists. Lingaya et al. [21] study the problem of rescheduling locomotives and carriages at VIA Rail Canada. Ben-Khedher et al. [6] present a schedule planning system and a tactical capacity-adjustment system for increasing revenue at SNCF.

The online (real-time) rolling stock problem is a new area of research which has received limited attention recently. The online approaches aim to accommodate railway disruption management where the offline approaches are not eligible partly because of the strict runtime requirements, but also because of different goals, restrictions or level of detail.

A paper by Nielsen et al. [22] proposes a setting and framework that is similar to the work of this paper. A rolling stock scheduling framework is presented that uses time horizons to overcome disruption uncertainty and runtime difficulty. The optimization model is similar to the model presented by Fioole et al. [16], but without the possibility of splitting and combining trains. The main goal is to minimize the number of cancelled trains, end-of-day off-balances and changes to the original shunting plan. The end-of-day train unit inventory balances are heuristically determined through the horizons. Every horizon is solved using a commercial MIP solver.

Cadarso et al. [12] present an approach which integrates rolling stock and dynamic passenger

demand. The model is allowed to cancel train services on predetermined lines, insert emergency train services and determine the direction of one-way tracks in case of one-way blocked segment. Passenger demand is updated iteratively after solving the rolling stock model until the demand stabilizes. The solution method is benchmarked on a 2 hour disruption case study provided by RENFE consisting of 5 lines, 10 depot stations and 760 train services. Solutions are found within few minutes. The case study does not reveal significant changes in passenger demand after the first iteration.

Jespersen-Groth [17] studied a variety of DSB S-tog related problems in her PhD thesis. One of the problems focused on the reinsertion of train lines once a disruption is over. The model considers the distribution of available trains and the time required for drivers to reach them. The model is solved as a MIP which minimizes the time of the last inserted train. The test instances are solved within 30 seconds.

3 Model

We now define the RSRP and notation more formally before presenting the proposed mathematical model. To distinguish between parameters and decision variables, the latter will always appear in bold type. Let $\mathcal{S}$ denote the set of key stations, which is an appropriate aggregation of all stations in the railway infrastructure. The set $\mathcal{D} \subseteq \mathcal{S}$ defines the set of stations with an associated depot. Furthermore, we let $\mathcal{U}$ be the set of all train unit types. We assume that all unit types are compatible and self-propelled, i.e., all units can be coupled together and no locomotive is required to pull the train. Finally, we let $\mathcal{T}$ be the set of all subtrips that have to be serviced as indicated by the timetable. A subtrip $t \in \mathcal{T}$ departs at a certain time from a source station and arrives at a certain time at a target station. For convenience, we define $\mathcal{A}$ to be the set of all subtrips that arrive at a depot-station $d \in \mathcal{D}$, where $\mathcal{A} \subseteq \mathcal{T}$. We define $depot(a)$ as the shorthand for the depot-station of subtrip $a \in \mathcal{A}$.

The RSRP is the problem of finding an assignment of all available units to subtrips. The assignment of each unit must respect the inherent temporal and spatial constraints, i.e., a unit can only be at one place at the same point in time and can only be assigned to a subtrip if it is located at the departure station of the subtrip prior to the subtrip's departure time. A unit entering or leaving a trip sequence must have reserved enough time to perform the (de)coupling before being assigned or reassigned to any other activity. Finally, the depot parking space is limited which means that only a certain total length of train units can stay parked at depot-stations any point in time. Consequently, a unit cannot perform a subtrip and then park if the destination depot is full.

We consider the following objectives in the mathematical model: train trip coverage, seat shortage, end-of-day balance, (de)couplings and operational cost. Trip coverage is directly related to trip cancellations, and it is our primary concern to minimize the number of cancellations. If no unit is covering some subtrip in $\mathcal{T}$ then the trip is considered to be cancelled. Matching the demand is also important, thus the seat shortage is minimized. The number of available seats depends on how many units service a subtrip, and the difference between available seats and the demand defines the shortage. However, if there is a surplus of seats then the shortage is zero. Couplings and decouplings affect the robustness of the solution, and hence we want to minimize the total number of these. We approximate the number of (de)couplings by counting the number of times a unit enters and leaves a trip-sequence. This is not an accurate number of required shunting operations; however, it measures how many times we change the composition of a trip-sequence which is directly related to robustness. We do not consider schedule deviations in the objective function in our approach; the model can, however, be extended to do so with the addition of more constraints and variables.

Instead of presenting a compact flow-based formulation of the rolling stock rescheduling problem we present a path-based model with an exponential number of path variables. Let $\mathcal{P}_d^u$ denote the set of all paths for unit type $u \in \mathcal{U}$ starting in depot-station $d \in \mathcal{D}$. A path represents a chronological list of subtrips that can be preformed by a single unit, i.e., a unit's schedule for the

$\mathcal{S}$	Set of stations
$\mathcal{D}$	Set of stations that have an associated depot
$\mathcal{T}$	Set of subtrips
$\mathcal{U}$	Set of train unit types
$\mathcal{A}$	Set of all arrival events across all stations in $\mathcal{S}$
$\mathcal{P}_d^u$	Set of all possible paths for unit type u starting at station d

Table 1: List of sets

$\boldsymbol{\lambda_p}$	Binary variable deciding whether path p is used
$\boldsymbol{y_t}$	Binary slack variable deciding whether subtrip t is covered
$\boldsymbol{z_t}$	Integer slack variable determining the seat shortage on train t
$\boldsymbol{w_d^u}$	Integer slack variable determining end-of-day balance shortage for unit type t

Table 2: List of Variables

c_1^t	Cost of cancelling a subtrip t
c_2	Cost of one seat shortage per kilometre
c_3	Cost of one end-of-day balance shortage
c_4	Cost of one coupling or decoupling
c_5^u	Operational cost per kilometre for unit type u
s_u	Number of seats on train unit u
$demand_t$	Seat demand for subtrip t
$length_t$	Maximum train composition length allowed for subtrip t
$inventory_d^u$	Number of units of type u starting at depot d
eod_d^u	Target end-of-day balance for train unit u at depot d
$track_d$	Combined length of tracks at depot d

Table 3: List of parameters

κ_t	Number of kilometres on subtrip t
ξ_p	Number of couplings and decouplings imposed by path p
κ_p	Number of travel kilometres accumulated on path p
α_p^t	Binary coefficient which takes the value 1 iff path p visits subtrip t
β_p^d	Binary coefficient which takes the value 1 iff path p terminates at depot d
$\gamma_p^{d,a}$	Binary coefficient which takes the value 1 iff path p is staying at depot d on or before the arrival of subtrip $a \in \mathcal{A}$

Table 4: List of Coefficients

considered planning period. Shunting operations are implicit as a unit has to be parked whenever the pause between two subtrips is too long The rescheduling problem thus consists of assigning exactly one path to each vehicle.

The mathematical model contains four sets of decision variables, two of which are binary and two of which are integer. First, $\boldsymbol{\lambda_p} \in \{0,1\}$ governs whether path $p \in \mathcal{P}$, where $\mathcal{P} := \bigcup_{u\in\mathcal{U},d\in\mathcal{D}} \mathcal{P}_d^u$, is selected in the final solution or not. Second, $\boldsymbol{y_t} \in \{0,1\}$ is a slack variable that determines whether a subtrip $t \in \mathcal{T}$ is cancelled. Third, $\boldsymbol{z_t} \in \mathbb{Z}_0^+$ is a slack variable that counts the total seat shortage on subtrip $t \in \mathcal{T}$. Finally, $\boldsymbol{w_d^u} \in \mathbb{Z}_0^+$ denotes the number of units of type $u \in \mathcal{U}$ that are missing in depot $d \in \mathcal{D}$ from the scheduled end-of-day balance. The model is presented below.

$$\text{Minimize: } \sum_{t\in\mathcal{T}} \left(c_1^t \boldsymbol{y_t} + c_2^t \kappa_t \boldsymbol{z_t}\right) \tag{1}$$

$$+ \sum_{u\in\mathcal{U}} \sum_{d\in\mathcal{D}} c_3^{d,u} \boldsymbol{w_d^u} \tag{2}$$

$$+ \sum_{u\in\mathcal{U}} \sum_{d\in\mathcal{D}} \sum_{p\in\mathcal{P}_d^u} (\xi_p c_4 + \kappa_p c_5^u) \boldsymbol{\lambda_p} \tag{3}$$

The notation is summarised in Tables 1, 2, 3 and 4. The objective function is a weighted average of five components. In (1) the cost of cancellations and seat-shortage are added. In (2) the end-of-day balance shortage cost is added. Finally, in (3) the (de)coupling and mileage cost is added.

$$\sum_{p\in\mathcal{P}_d^u} \boldsymbol{\lambda_p} = inventory_d^u \qquad \forall u \in \mathcal{U}, d \in \mathcal{D} \tag{4}$$

$$\sum_{p\in\mathcal{P}} \alpha_p^t \boldsymbol{\lambda_p} \geq 1 - \boldsymbol{y_t} \qquad \forall t \in \mathcal{T} \tag{5}$$

$$\sum_{u\in\mathcal{U}} s_u \sum_{d\in\mathcal{D}} \sum_{p\in\mathcal{P}_d^u} \alpha_p^t \boldsymbol{\lambda_p} \geq demand_t - \boldsymbol{z_t} \qquad \forall t \in \mathcal{T} \tag{6}$$

$$\sum_{p\in\mathcal{P}} \beta_p^d \boldsymbol{\lambda_p} \geq eod_d^u - \boldsymbol{w_d^u} \qquad \forall u \in \mathcal{U}, d \in \mathcal{D} \tag{7}$$

$$\sum_{u\in\mathcal{U}} l_u \sum_{d\in\mathcal{D}} \sum_{p\in\mathcal{P}_d^u} \alpha_p^t \boldsymbol{\lambda_p} \leq length_t \qquad \forall t \in \mathcal{T} \tag{8}$$

$$\sum_{u\in\mathcal{U}} l_u \cdot inventory_d^u + \sum_{u\in\mathcal{U}} l_u \sum_{d\in\mathcal{D}} \sum_{p\in\mathcal{P}_d^u} \gamma_p^{d,a} \boldsymbol{\lambda_p} \leq track_{depot(a)} \qquad \forall a \in \mathcal{A} \tag{9}$$

$$\boldsymbol{\lambda_p} \in \{0,1\} \quad \boldsymbol{y_t} \in [0;1] \quad \boldsymbol{z_t}, \boldsymbol{w_d^u} \in \mathbb{Z}_0^+ \tag{10}$$

Constraints (4) ensure that the number of paths for a specific unit type leaving a depot corresponds to the number of units of that type available at the depot. The second set of constraints (5) ensures that $\boldsymbol{y_t}$ is set to 1 if a subtrip t is not covered by any path chosen. Constraints (6) ensure that the lack of seats is captured by $\boldsymbol{z_t}$, while Constraints (7) make sure that the end-of-day balance shortage is captured in $\boldsymbol{w_d^u}$ for every depot d and unit type u. The maximum train composition length is restricted by constraint (8) according to the platform lengths at the subtrip's endpoints. Constraints (9) enforce the requirement that the capacity at any depot d must be respected by every arrival event. If the track capacity at every arrival is satisfied up until the last arrival, then the schedule respects the depot capacities throughout the day. Finally, Constraints (10) restrict the domains of the variables. Note that the slack variables are naturally integral in the presented model if the path variables are integral.

Naturally we do not propose to solve the full model, instead we intend to solve the model using delayed column generation. Such path based models have proven to improve runtime drastically for a number of routing problems using a delayed column generation approach [15]. We do not

achieve a tighter LP relaxation for the simple rolling stock rescheduling problem. A tighter LP relaxation is however obtained when we extend the problem with maintenance restrictions, and thereby require the subproblem to find resource constrained shortest paths in contrast to simple shortest paths.

3.1 Rescheduling Extension

Considering a single horizon in the middle of a whole day presents some challenges which requires some minor changes to the model. Essentially these changes also allow the framework to work using a rolling horizon, c.f. [22].

Trains are running continuously and some decisions made a few minutes ago cannot be undone, e.g., if a train unit was assigned to a departure happening before the horizon begins, then the train will go through with this trip until it arrives at the next station. Likewise, any train unit assigned to a departure before the horizon ends will have to go through with the trip into the following horizon.

In the proposed model every train unit starts at a depot and is available from the beginning of the planning period. However, inside a horizon some trains will only become available some time after the start of the horizon, due to being en route when the horizon begins. This implies that some units will be unable to cover the first subtrips. To model this, we partition the sets of unit types available at each station $s \in \mathcal{S}$ further, based on a set of n_s discrete event times $\tau_s = \{\theta_1, \theta_2, \ldots, \theta_{n_s}\}$. Associated with each time $\theta \in \tau_s$ is a known number of available units of each type $u \in \mathcal{U}$. For the sake of completeness, we assume this number is $inventory_s^{u,\theta}$, and note that, for all $u \in \mathcal{U}$, $inventory_s^u = \sum_{i=1}^{n_s} inventory_s^{u,\theta_i}$. Constraints (4) hence become:

$$\sum_{p \in \mathcal{P}_s^{u,\theta}} \boldsymbol{\lambda_p} = inventory_s^{u,\theta} \qquad \forall u \in \mathcal{U}, s \in \mathcal{S}, \theta \in \tau_s, \tag{11}$$

where $\mathcal{P}_s^{u,\theta}$ contains all paths for unit type $u \in \mathcal{U}$ originating from station $s \in \mathcal{S}$ after time $\theta \in \tau_s$. The other constrains must now sum over all these paths and not just for all depots and unit types alone. We note that this modification does not affect the structure of the subproblem and does not increase the number of subproblems to solve.

Another complicating issue that arises is how to determine the number of couplings. Units arriving on a subtrip from the previous horizon may have been routed to a platform (in order to continue on some trip sequence) or into the depot for parking. Since the arrival times of these units occur after the beginning of the horizon, the previous decisions can be altered. Thus, such arriving units are denoted *undecided*, and the model decides if it is to be parked in the depot or if it should go to the platform and continue on some immediate subtrip. As a consequence, the price of decoupling must be paid in the current horizon as it was undecided whether the subtrip assignment would lead to a decoupling once it arrived. This detail is complicated even further if we impose a minimum (de)coupling time. In this case the model might have decided to park a unit arriving in the last horizon, and this unit will not be available in the current horizon until a certain time has elapsed. In contrast to before the unit is now available from the depot only, and can not take a subtrip from the platform immediately. This issue can be modelled the same way as the undecided units by adding a new similar set of constraints and limiting the possible paths to emerge from the depot.

4 Branch-and-Price Framework

In this section we propose a B&P framework for solving the RSRP. This well known technique for solving large-scale integer programs combines column generation with Branch-and-Bound (B&B) [5]. Column generation is typically preferred when it is computationally intractable to build a mathematical model which explicitly contains an enumerated set of all possible variables in the problem. To implement column generation, one relaxes all integrality restrictions and

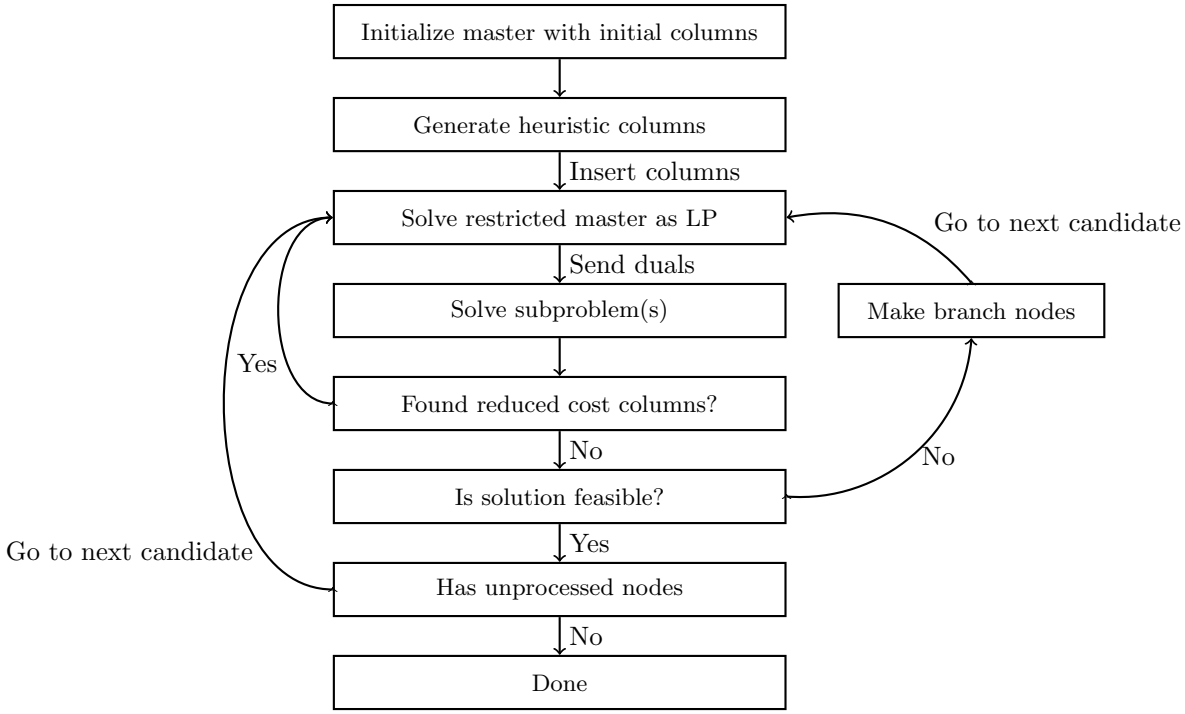

Figure 5: Flow diagram for the proposed Branch-And-Price framework

decomposes the problem into a *master problem* and one or more independent *subproblems*. The master problem contains only a subset of the variables for the full problem, and is typically referred to as the Restricted Relaxed Master Problem (RRMP). The subproblem(s) is/are an optimization problem responsible for generating variables (i.e. columns) that are not included in the RRMP, but which have the potential to decrease the RRMP's objective function value. More specifically, each subproblem utilizes the dual information from an optimal solution to the RRMP, and attempts to identify negative reduced cost columns that can be added to the RRMP. Column generation refers to the iterative procedure between master and subproblems that must be performed. Incorporating this into a B&B setting in which columns are generated (or "priced") at each node of the B&B tree gives rise to the B&P terminology. This section focuses on specific details concerning our B&P approach. Figure 5 illustrates the steps in the framework. First, a master model is generated with initial *feasibility* columns, i.e., a set of variables are initially inserted such that a feasible solution exists for the LP relaxation. The RRMP is solved and the dual values are used to find variables with reduced cost by the subproblem. Whenever new variables are found the RRMP must be resolved, otherwise the RRMP is optimal. A B&B node is finished when an optimal solution to the RRMP is found. If the solution is feasible the upper bound is updated otherwise new branching nodes are created. The framework terminate once all nodes are processed.

4.1 Master Problem

The RRMP consists of a set of constraints as in the model presented in section 3, where constraints (10) are replaced with

$$\boldsymbol{\lambda_p} \in [0,1] \quad \boldsymbol{y_t} \in [0,1], \quad \boldsymbol{z_t}, \boldsymbol{w_d^u} \in \mathbb{R}, \tag{12}$$

and restricting the set of path variables for each depot and unit type pair (d,u) to a subset $\mathbb{P}_u^d \subset \mathcal{P}_d^u$. For this relaxed version of the problem we define dual variables $\pi_d^{u,\theta}$, μ_t, δ_t, ω_u^d, ϕ_t,

and ν_a; these are associated with Constraints (5), (6), (7), (8), (9), and (11) respectively. Each column defines a legal path through the rail network for a unit of type u originating from depot d.

4.2 Subproblem

The role of a subproblem is to identify one or more negative reduced cost columns. As indicated above, this can be viewed as the pricing step in the conventional simplex algorithm, with the exception that all variables are not stored and priced explicitly. Variables with negative reduced cost for the RRMP are returned by the subproblem(s). The variables correspond to unit paths for particular unit types. When generating a path the following information must be determined: the depot where the unit type begins the path, the terminating depot, the sequence of subtrips the unit will perform, and any intermediate coupling activity the unit type must perform.

We formulate the problem of finding such paths as a Shortest Path Problem on an acyclic time-space network. We group together units that have the same characteristics (i.e unit type) and define a subproblem for each such group. The subproblem for each group routes one physical unit, where any specific unit from the group can perform the path. Note that the underlying network structure of each subproblem is identical since we assume any unit can perform any subtrip; however, some parameters of each subproblem must be modified to reflect the individual characteristics of the respective unit groups. In the following we present a more detailed description of the time-space network. To assist the reader in understanding all details of the network involved, it is introduced in stages. Not investigated in this paper is the possibility of extending the subproblem into a Shortest Path Problem with Resource Constraints (SPPRC) [15]. Unit-specific properties such as maintenance constraints can be modelled using SPPRC.

4.2.1 Underlying Network

To generate a unit path one must identify a *feasible* sequence of subtrips starting and ending at particular depots. Modelling all such possibilities for a given planning period can be achieved via a time-space network. In such a network, every node corresponds to a particular *event*. Here an event can refer to a departure, a passthrough, or an arrival. A departure event refers to the start of a trip, while an arrival event is associated with the end of a subtrip. A passthrough event represents a midway stop between two trips - one incoming and one outgoing subtrip is associated to each passthrough event. Any subtrip indicates the movement of a train between two particular stations at a certain time and thus can be introduced as an edge in the time-space network.

Associated with each subtrip edge is a passenger demand, a distance κ_t (between the delimiting stations), and a subtrip operating cost, $c_5^u \kappa_t$. Note that the operating cost depends on the unit type under consideration. Furthermore, each subtrip is associated with a constraint from (5), a constraint from (6), and a constraint from (8). Thus, the reduced cost contribution, and hence the weight, of each such edge is $c_5^u \kappa_t - \mu_t - s_u \delta_t - l_u \phi_t$. Additionally, edges which indicate an arrival subtrip at the depot are adjusted by $-l_u \nu_a$ consumption of depot track capacity. By adding a super source as well as a super sink node, connecting the former to all departure events and the latter to all arrival events, we obtain an acyclic, time-space network, in which all trips within the planning period are implicitly represented by paths (consisting of subtrip edges). Note that the edge weight of edges emanating the origin is $c_4 - \pi_u^d$, while edges to the sink have weight $c_4 - \omega_u^d$. Here, the coupling/decoupling cost c_4 is added to reflect the fact that the unit enters and exits a trip sequence.

Figure 6 below illustrates such a time-space network for a simplified example with three stations, two of which are depot stations (denoted d_1 and d_2). The example contains two trip sequences (each having three trips), and there are 12 unique subtrips. All nodes corresponding to station events are ordered in time, and appear on the same horizontal level. The network is labelled assuming we are pricing a unit of type u. An example path from depot-station d_1 to depot-station d_2 is given; the weight of any subtrip edge is $c_5^u \kappa_t - \mu_t - s_u \delta_t - \phi_t$, where t denotes the associated subtrip. Note that this network does not allow decoupling at intermediate stations. Here a and a' denote the two arrival subtrips at the depot.

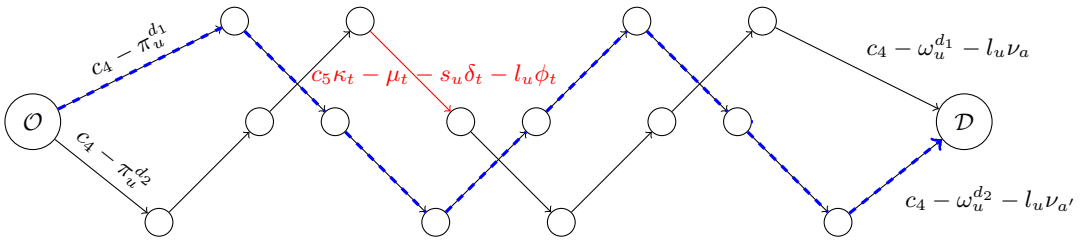

Figure 6: Simplified example of the acyclic time-space network. This figure illustrates the subtrips of two trip sequences. The blue path is an example of a possible unit path. The current network does not capture depot details.

While this time-space network includes all subtrips in the planning horizon, it is inflexible in the sense that it does not permit a unit to be decoupled from a trip sequence; since each node in the graph has degree at most two, each path must consist of a sequence of consecutive subtrips. In reality this is not ideal, it may be preferable to allow units to follow only part of a trip-sequence. To provide such flexibility, we duplicate all nodes for station events occurring at all depot-stations, creating a *platform event* node as well as a *depot event* node. The purpose being to provide finer details on the possibilities available for a unit, given its location at the time of the respective station event. A platform event stipulates that the unit type is at the platform and must perform the outgoing subtrip, while a depot event merely indicates that the unit type is positioned in the depot and cannot perform the outgoing subtrip.

To allow a unit type to remain parked at a depot-station we introduce edges connecting consecutive depot events at the same station. Similarly, we introduce so-called *coupling edges* connecting depot events with platform events indicating that the unit type will be "coupled" with other units to perform the outgoing subtrip. Coupling edges must respect a certain minimum coupling time, meaning a depot event cannot connect to the station platform event before the coupling time has elapsed. Similarly, *decoupling edges* are introduced, each of which connects a platform event with a depot event. In contrast to coupling edges, the edges are not connecting platform and depot events at the same station, thus eliminating the possibility of coupling a unit without taking a subtrip. Again, we assume a minimum decoupling time on decoupling edges, which is modelled by connecting these edges to the target station depot after the decoupling is completed. Note that the introduction of decoupling edges implicitly duplicates the number of subtrip edges that terminate at depot-stations; for each such subtrip there are two edges in the time-space network, one indicating whether the unit ends at a platform (a *platform subtrip*) and one indicating if it ends in the depot (a *depot subtrip*). All coupling and decoupling edges are assigned an additional cost c_4 to reflect the price one must pay to perform a coupling. The cost an any depot subtrip edge is $c_4 + c_5^u\kappa_t - s_u\mu_t - \delta_t - \phi_t$, where $t \in \mathcal{T}$ is the subtrip in question. By counting the number of coupling edges used in any path from the source to the sink, one can determine how many times the unit is coupled and/or decoupled from a trip sequence. Finally, all coupling and decoupling edges must be adjusted by the dual contribution from Constraints (9) since any such edge indicates the consumption (or release) of depot track capacity. This dual contribution on any decoupling edge is given by $-l_u\check{\nu}_a$, with subtrip $a \in \mathcal{A}$ and $\check{\nu}_a = \sum_{a' \in \mathcal{A}_a^+} \nu_{a'}$. Here $\mathcal{A}_a^+$ gives the set of all subtrips in $\mathcal{A}$ arriving at depot $d = depot(a)$ whose arrival time is at least as large as that of subtrip a. Similarly, the depot track dual contribution on any coupling edge is $l_u\hat{\nu}_a$, where $\hat{\nu}_a = \sum_{a' \in \mathcal{A}_a^-} \nu_{a'}$. Here $\mathcal{A}_a^-$ is the set of all subtrips departing from d whose departure time is at least the arrival time of a. Informally, in the former case the dual represents the cost of allocating depot space when decoupling; in the latter case the dual represents the gain of releasing depot space.

Constructing the network in such a fashion allows units to perform a set of non-consecutive subtrips, include the coupling costs, and respect the minimum coupling durations. The last feature allows us to monitor when a unit type will be available to perform certain subtrips if coupling

and/or decoupling decisions have been made. The shortest path in such this network identifies the column with minimum reduced cost.

Figure 7 shows how the time-space network from Figure 6 can be extended to include depot subtrips as well as coupling and decoupling edges. Stations with depot tracks have depot event nodes and are coloured light gray in the figure. Note that all coupling and decoupling edges have not been included; however, an example of each can be seen in two of the three example paths provided. The blue path depicts a unit which does not take the first outgoing subtrip from depot-station d_2, but is instead coupled to the second outgoing subtrip. The unit then performs four consecutive subtrips. The red path also illustrates a unit that starts from depot-station d_2 and performs four consecutive subtrips; however, this unit performs the first outgoing subtrip and is decoupled upon completing its fourth subtrip. In contrast, the green path gives an example path for a unit that does not move from depot-station d_1 during the planning period.

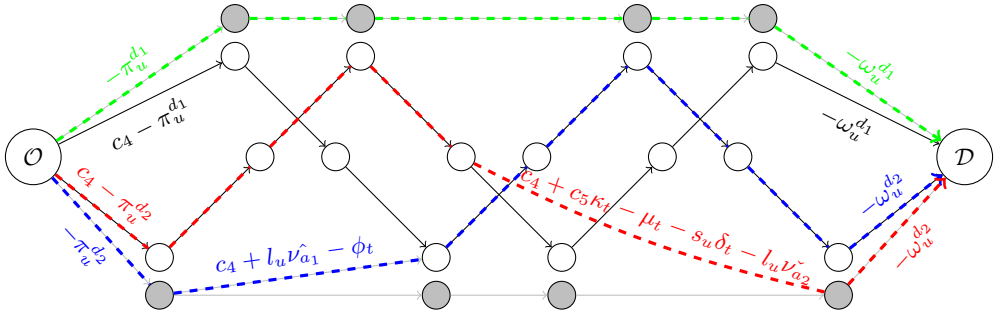

Figure 7: A simplified example of the acyclic time-space network showing some of the possible coupling arcs. Only two of the stations have an associated depot. Three possible unit paths have been highlighted.

4.2.2 Modifications Necessary for Rescheduling Extension

While Figure 7 accurately reflects a unit's possible movements over the entire planning period, some additional features are needed to ensure it can be embedded in a rescheduling environment. As mentioned in Section 3.1, in such a situation it is necessary to distinguish between unit types that are definitely positioned in depots at the start of the planning horizon, and those which are unavailable to be assigned immediately due to being en route when the planning horizon commences. To incorporate such functionality, a *station source* node, a *depot inventory* node, and a number of *undecided* nodes are introduced for each station. The super source connects to each of the station sources, which in turn each connect to a station's inventory and undecided nodes. The inventory node grants access to the full network by being connected to the station's first platform and depot event node. In contrast, the undecided node grants access to a limited part of the network via several *availability edges*. Each availability edge is associated with a specific station $s \in \mathcal{S}$ and a particular time $\theta \in \tau_s$ corresponding to the time of arrival. Such edges ensure that the associated set of undecided units $inventory_s^{u,\theta}$ cannot be assigned paths until they are available, i.e. after a certain time has elapsed. For each pair (s, θ) there are at most two edges giving the respective set of undecided units the possibility of continuing on a consecutive subtrip, or entering the depot.

Continuing with our examples from Figures 6 and 7, Figure 8 illustrates a complete subproblem network that is applicable in a rescheduling environment. The illustrated horizon starts just after the departure of the first subtrip from depot-station d_1 in Figures 6 and 7. All coupling and decoupling edges have been included and are given as dashed lines. Station sources, depot inventory nodes, and undecided nodes are denoted s, i, and u respectively. In Figure 7, two subtrips are in progress upon termination of the first horizon, and as a result the units assigned to each in the first horizon will only be available after a certain time has elapsed. The edges

emanating u_3 describe this. Figure 8 gives the two example unit paths from Figure 7. The blue path illustrates a path which begins at depot-station d_2. The unit is coupled to the first outgoing subtrip, performs four consecutive subtrips, and terminates at the platform of depot-station d_2. The red path is an example of a path that continues from the first horizon. The units performing this path are only available after a certain time θ_1 at station d_3. The red path depicts a path that performs three consecutive subtrips, after which it is decoupled at depot-station d_2. Edges from the super source to the station sources have cost zero. The same is true of the edges emanating the station sources.

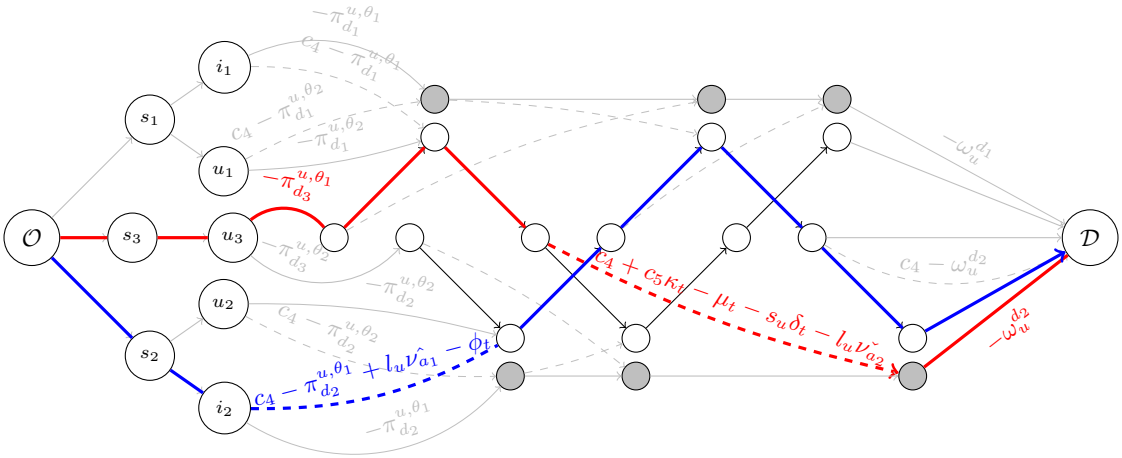

Figure 8: A simplified example of the acyclic time-space network showing the extensions necessary for it to be used in a rescheduling environment

By including restrictions on unit mileage and coupling count we can consider maintenance restrictions, which may require a unit to have a path that does not exceed a certain maximum length, and control how many couplings – at most – we wish to permit any unit to perform. The nature of the time-space network is such that we can also satisfy requests which ask specific units to end at specific depot-stations. Note, that adding potential deadheading trips corresponds to adding edges between station depots, and does not require additional variables or constraints in the master problem. As a side comment, the network will contain negative edge weights; however, negative cycles cannot not exist due to its acyclic nature. The shortest path can therefore be found using a directed acyclic graph shortest path algorithm with linear time complexity.

4.3 Branching

The column generation process terminates with an optimal solution to the RRMP. If the variables satisfy the integral constraints a feasible solution to the RSRP has been found, and we can update the upper bound and prune the current branch. This is, however, not always the case and we need to define branching rules in order to enforce integrality.

Branching on (master) variables in the RRMP is not preferable in general, nor in our case. Forcing a fractional path variable to take the value 0 or 1 in separate branch nodes complicates the subproblem. In the 0 branch the subproblem will (without alterations) likely return the forbidden path as a new variable. Thus, in order to disallow specific paths we need to solve the *k-shortest (resource constrained) path problem*, just to make sure that we have the shortest path that is not forbidden by branching rules. In addition, the efficiency of branching on paths is uncertain as the total number of paths is extremely large. We propose branching on certain characteristics of the paths instead, which is easier to handle in the subproblems.

An interesting and effective branching rule has been proposed by Ryan and Foster [26] for the similar crew scheduling problem. Instead of branching on original or *subproblem* variables, which often is imbalanced or ineffective, they propose a more balanced branching rule which can be handled entirely by the subproblem. In crew scheduling a split flow is found and a branching rule is set such that a column either covers two consecutive crew tasks or does not cover them both. Despite the similarities of the considered problem we cannot adopt this approach since our problem does not exhibit the required structure for this type of branching. In our case, we can enforce branches based on subtrip variables, but the cover of a sub-trip is allowed to be greater than exactly one. Furthermore, applying the branching scheme on specific units introduces at least two disadvantages. First, such a rule will not have a significant effect since we have many similar train units. Secondly, we will need to solve variations of the subproblem for every unit depending on its individual branching constraints.

Our first family of branching rules enforces integrality on the number of paths originating and terminating at a depot. The number of paths leaving a depot (as the first action) can never be fractional. The same holds for the number of paths entering (as the last action) a depot. This rule can be applied on two levels, either for a specific unit type or aggregated over all types. Our B&P framework imposes this rule on an aggregated level where possible, otherwise on a specific unit type.

Even if the paths respect the integral requirements above, the solution may still be fractional. The second family of branching rules enforces integrality on subtrip coverage. The subtrip cover is equal to the number of (possible fractional) paths that service a subtrip. We therefore ensure that the total subtrip cover $f \in \mathbf{R}$ is integer by imposing that the cover is $\leq \lfloor f \rfloor$ in one branch and $\geq \lceil f \rceil$ in the other. This branching rule can also be applied on two different levels. Where possible we aggregate over unit types, otherwise we create a branch based on a specific unit type.

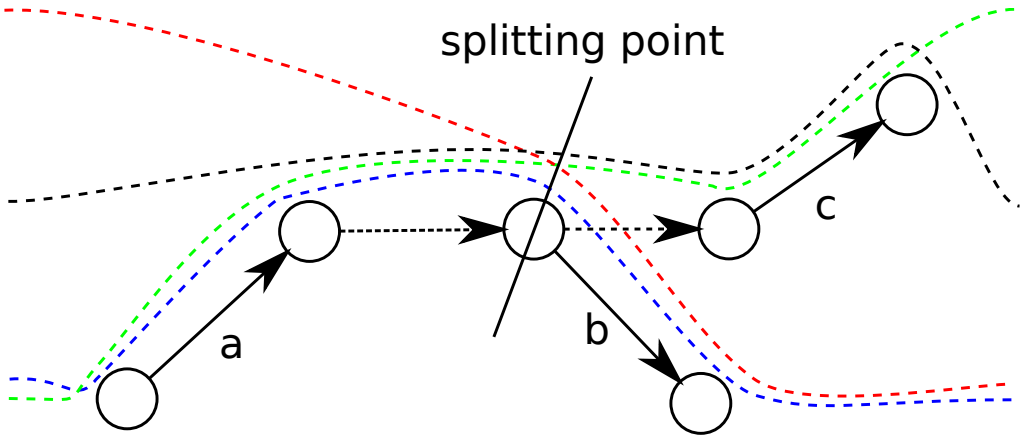

Figure 9: A general setting of two fractional paths where they begin to split from each other. The example include two subtrips edges and one waiting at station edge.

Proposition 1. *The proposed branching rules form a complete branching strategy given that the decoupling and coupling costs are strictly positive.*

Proof. Assume we have a solution that respects the constraints in RRMP and the branching rules. If a variable is fractional then there must exist other fractional paths in order to maintain integral trip cover and integral start/end inventory. The situation is illustrated in Figure 9. Note, that the illustrated paths may consist of multiple paths where the sum is fractional. The figure illustrates a general case where two connected fractional paths split. Flow can only enter and leave stations with a sum that is integral. Consequently, before the paths are able to split (black and red) there must exists two paths (blue and green) that arrived earlier through a common subtrip. The case

where both paths (blue and green) follow each other after the split is not interesting as this can be substituted with a single path, thus requiring another fractional path to follow the black path.

We now argue that the general case with fractional paths can be transformed such that the paths (blue and green) end up following each other to a greater extent. Thus, if a solution contains fractional paths then the same argument can be repeated until all splits have been resolved. Note that when we swap path trajectories at the splitting point only the coupling and decoupling signature changes. Other operational costs stay constant as we service the same subtrips and do not change end-of-day balances. The argument is divided into three exhaustive cases based on which path is following a trip sequence.

1. If the blue path is following a trip sequence (i.e. subtrips a and b belong to the same sequence) then we know that the green path has to decouple at the splitting point and the red path has to do at least one coupling. Thus, if we instead swap the red and green trajectories after the splitting point, then we save the green decoupling. The red path saves one coupling, but it is uncertain whether it has to use a coupling to service subtrip c. The new transformed solution is at least one coupling cheaper and we conclude that this scenario cannot appear since it contradicts that the flows are part of an optimal solution.

2. If the green path is following a trip sequence (i.e. subtrips a and c belong to the same sequence) then the blue path has to decouple at the splitting point while the black path must at least do one coupling. Thus, swapping the blue and black paths after the splitting point will definitely save the blue decoupling. Again, this scenario cannot appear as it leads to a contradiction.

3. The above argument is symmetric if either the red or black follow a trip sequence. If however none of the paths follows a trip sequence, then swapping the red and green paths after the splitting point would not change the objective value. In such cases we can find an integral solution of equal value.

Note that if the coupling and decoupling costs are zero, then the whole argument can be repeated, only in every case we can transform the fractional paths to integral paths while maintaining the same objective value.

□

5 Computational Results

In this section we present our computational results using the presented B&P framework. The solution method is benchmarked against data provided by DSB S-tog. Our primary focus is on disruption handling which is why we will investigate the potential of the solution method as a recovery tool during large disruptions using disrupted timetables. However, the only essential differences between a normal and a disturbed case are only the timetable and the objective weights. Generally trip cover and seat demands are hard constraints in planning but arguable from a planners perspective soft constraints allow more flexible solutions, e.g. covering 99.9% of seat demand. From the perspective of the B&P algorithm only the input data is changed. Therefore, aside from the general assumptions in this paper the framework can be used to solve normal day-to-day schedules. We will therefore initially also present some results of normal (non-disrupted) timetables.

5.1 Data

The test instances consist of real-life data provided by DSB S-tog operation in spring of 2014. We have collected realized rolling stock schedules as well as the tactical and short term plans for specific dates. DSB S-tog adopts a weekly periodic timetable that amounts to four distinct timetables. All weekdays follow the same timetables while Saturday and Sunday have a different

Name	Stops	Subtrips	Subtrips'	Weekday	Lines
Fri	28 719	3 259	1 086	Friday	A,B,Bx,C,E,F&H
Sat	20 474	1 795	706	Saturday	A,B,C&F
Sun	19 919	1 753	690	Sunday	A,B,C&F
Mon	28 017	3 196	1 074	Monday	A,B,Bx,C,E,F&H

Table 5: Four timetable and demand datasets provided by DSB S-tog. The columns respectively show the instance names, total number of stops, total number of natural subtrips, number of subtrips after preprocessing, Weekday and finally the lines that are running.

Scenario #1 - L Segment	
Contingency Plan	A.3.10 (Valby - Frederikssund)
Description	One track is blocked on one of the fingers
Cancelled lines	H
Turned lines	C1
Unchanged	A1, A2, B1, B2, Bx, C2, E1, E2

Table 6: Details of the *leaf* type disruption case. The disruption blocks one track between Valby and Frederikssund stations.

weekend timetable. However, on Fridays and Saturdays there are additional night-trains. In total we have four different cases: Monday-Thursday, Friday, Saturday and Sunday. Table 5 summarises key metrics of the selected instances. Preprocessing has been performed to combine subtrips that arrive or depart from stations without shunting options. Such subtrips can be combined with the succeeding or preceding subtrip since the composition cannot be changed. Considering non-preprocessed data-set can be misleading as the complexity of the instance is related to the number of *real* decisions that can be made.

The number of possible disruption scenarios is very high as there can be multiple points of failure, disruption durations, start times and levels of severity. For the point of benchmarking our solution method we find it sufficient to benchmark two different types of disruptions of different severity. The topology of the infrastructure is star-shaped which means that two interesting cases appear: A disruption in the center (known as *the central section*), or a disruption along one of the leaves (known as *the fingers*). The disruption timetables are modified as stipulated by the existing contingency plans. We will test different start times and durations of both disruption types. Figure 6 and 7 describe the two selected cases.

Scenario #1 - Central Segment	
Contingency Plan	A.0.16 (København H - Østerport)
Description	One track is blocked in the central segment
Cancelled lines	A2, B2, Bx, C1, E2, H
Turned lines	A1, B1, C2, A1, C1
Unchanged	E1

Table 7: Details of the *central* type disruption case. The disruption blocks one track between København H and Østerport stations

5.2 Lower Bound

The gap between the optimal solution and the root LP relaxation is in some cases relatively high - even above 10%. This is not surprising as the problem has many similarities with the Integer Multicommodity Flow Problem (MCFP), which shows a similar property, see [4].

We find another bound using a MIP formulation of the problem. With the addition of depot-related constraints, the model is essentially the formulation presented by Fioole et al. [16]. The correctness of this bound is apparent as the flow model solves the same problem, only not accounting for the structure of the generated paths. A valid solution to the path formulation is feasible in the flow model.

The lower bound found by the flow model proves to be stronger than the LP relaxation of the master problem. The model is solved in a few seconds or up to one minute. We therefore adopt this bound for future results.

5.3 Tuning

Every B&P framework has many tuning-options. Without going into details we briefly mention some initial tuning experiments which have been performed.

The columns of the initial master problem are chosen carefully to ensure feasibility. We tried inserting different heuristic columns that cover sequences of subtrips. Inserting such heuristic columns did however not seem to improve convergence nor turn out to be beneficial in any way.

An experiment was performed inserting the lower bound as a constraint on the objective in the master problem. This experiment did however not show any consistent improvement on data sets.

Preliminary tests revealed that the branching rules are not very effective - the number of nodes in the B&B tree is high. We have four different types of branching rules and the number of possible branches in every fractional node is large. We implemented and tested *Strong Branching* which proved to efficiently reduce the B&B tree in general. We adopt a variant of *Full Strong Branching* where all possible branching candidates are generated and ordered by fractional distance. The candidates are evaluated in order until no improvement has been made for a certain number of iterations [2].

Generating multiple columns in every iteration in the B&P framework improves convergence, compared to only inserting one path with the lowest reduced cost. However, generating too many columns proved to have a negative impact on runtime. For every depot we find and use the lowest reduced cost path per unit type.

Depending on the considered objective the instance root relaxation is, more often than not, far from integrality. In order to get an fast initial solution we tried add a step which solves the root node relaxation as a MIP. In many cases this proved to find very good initial solutions in a few seconds. We therefore adopt this approach.

A B&P framework is ideal for parallel execution. Our results did however show that the performance can sometimes decrease, especially if small instances are considered. Our conclusion is that parallel execution is most beneficial when the instances are large, and when the B&B tree becomes large. For the sake of clarity, the following benchmarks will be run using only a single thread.

A comparison between using CPLEX and CLP (COIN-OR simplex solver) for solving the master relaxation showed that the latter is both faster and more robust. We observed a faster and more steady convergence using CLP. Note however that CPLEX, due to a different execution path, was in a few cases able to find other better incumbent solutions faster. In our experiments we use CLP for solving the master relaxations.

5.4 Planning Benchmark

The rolling stock rescheduling problem has multiple objectives. In our paper we consider a single objective function which in turn is a weighted sum of all objectives. Finding the correct or

Instance	Horizon	Subtrips	Columns	Time	Runtime		
					Strong	Column	LP
Fri	4:40-27:09	1 086	2 127	13	0.3%	5.7%	46.8%
Sat	2:00-27:09	706	682	3	2.4%	20.5%	0.0%
Sun	2:00-25:37	690	450	2	0.0%	4.3%	8.9%
Mon	4:37-25:37	1 074	1 941	11	0.4%	6.1%	36.0%

Table 8: Optimal solutions obtained only considering cancellation costs and a small cost for couplings. The elapsed time is measured in seconds. Strong, Column and LP show the relative time spent in strong branching, column generation and LP solving.

appropriate weights for each objective is far from trivial. The most important objective is the total number of trip cancellations; all other objectives are of small importance in comparison.

The only other factor related to economic cost is the total mileage of every unit. Even though we want to minimize cost, this is often less important than meeting the expected demand and performing shunting movements. The operational expenses are even less important during a disruption. During disruptions an unwanted end-of-day balance incurs economical cost since this implies that trains have to be deadheaded.

The planners are obliged to meet the passenger demand and cannot cancel any trains when they construct a rolling stock schedule in the planning phase. However, they are reluctant to add more units to a train if only a few passengers are expected to miss a seat. The exact number of acceptable seat shortages is left to the subjective judgement of the planners. During disruptions the problem is further complicated since the demand is no longer known due to a change of passenger behaviour. A study of passenger behaviour is out of scope for this paper which is why we will assume that demand stays unchanged for all trains services during disruptions. Arguably, this assumption may be unrealistic, nevertheless this information is better than none.

As argued, it can be difficult to find appropriate settings for objective coefficients which is why a few iterations may be necessary before making a decision. Different settings will also have an effect on the total runtime as it changes the difficulty of finding the optimal solution. The ideal settings thus depend on the quality of the solution and the runtime requirements. In the benchmark we solve the problem to optimality whenever practicable otherwise we will terminate the framework on a proven bound.

In the first experiment we benchmark the performance of the framework, only considering the most important objective, namely the number of cancellations. We assign zero cost to all other objectives but the coupling cost. A small cost to the number of couplings since preliminary experiments show that the solution method performs worse without. Without a coupling cost the number of alternative optimal paths in the subproblem increases drastically. The number of different path (with the same cost) has a bad effect on integrality in the master problem. The results for the first benchmark are shown in Table 8. The results show that we are able to solve the planning instances to optimality within a few seconds. In all cases we had a gap of 0% between the relaxed root node and optimal value. This suggests that our formulation has some good properties using the considered objectives. All instances were solved to optimality before branching could occur.

A solution with minimal cancellations is not necessarily a good solution - a minimal seat-shortage is also desirable. Experiments performed using a small coupling cost and different shortage costs in shown in Table 9. The results show that the algorithm has difficulty solving the problem to optimality. Seat cover is at 100% which means seat shortage contributes with a zero cost to the objective. A 5% termination gap was used since the framework has a hard time improving this gap, even given a timelimit of 1 hour. The gap is therefore solely caused by the number of couplings performed. The relatively large root gap is not unexpected since fractional unit paths are able to cover seat demand more cost-efficiently than integral paths which have to pay full coupling prices. In a planning context, the number of couplings is not a primary concern

Instance	Subtrips	Time	Columns	Gap	Root Gap	Cover
Fri	1 086	47	3 798	4.3%	13.9%	100.0%
Sun	706	2	714	0.0%	0.0%	100.0%
Sat	690	2	718	0.0%	2.8%	100.0%
Mon	1 074	66	4 268	1.9%	9.9%	100.0%

Table 9: Solutions obtained only considering cancellation costs, a small shunting cost and a seat shortage cost. A termination gap of at most 5% was set. The elapsed time is measured in seconds. *Gap* and The *Root Gap* show respectively the relative optimality bound and the relative distance to the root node relaxation. *Cover* shows the percentage of covered seats in the found solution.

Instance/Penalty	0.0	0.1	1.0	10.0
Fri	85.6%	99.0%	100.0%	100.0%
Sat	95.5%	99.6%	100.0%	100.0%
Sun	97.6%	99.5%	100.0%	100.0%
Mon	84.7%	98.9%	100.0%	100.0%

Table 10: Seat demand coverage obtrained using increasing seat shortage penalties.

compared to scheduling new couplings during the day of operation.

In planning the goal is to meet the expected demand. A seat-coverage of 100% is ideal, however if the cost of covering 99% is significantly less, then this is also an acceptable solution. Table 10 shows the seat cover using varying seat shortage costs - as expected, the seat cover increases as the cost increases.

Minimizing cancellations, shunting and seat shortages can lead to expensive schedules since train services can be executed with too much capacity. Minimizing the mileage is therefore also essential for the planning phase. Results using varying mileage costs are shown in Table 11. Note, that the given mileage is relative to the travel-distance of assigning one unit to all subtrips. The results confirm that the total mileage decreases as the mileage cost increases. The mileage decreases noticeably when assigning a small non-zero cost compared to ignoring it all together. The results show that an increase in mileage cost has an negative effect on seat coverage. Observing that the runtime quickly escalates while seat shortage starts to drop we adopt a value between 0.1 and 1 further on. Although the total runtime ranges up to 2 minutes we see that the best solution is found within a few seconds. The time spent after finding a good solution is merely closing the optimality gap. The low mileage cover suggests that the two smaller instances are easy instances since we do not need more than one unit to cover every trip.

5.5 Disruption Benchmark

Adopting the parameter settings found in the planning benchmarks we continue to study a number of disruption cases. In this setting we will only consider short runtimes as this is a crucial precondition during a disruption resolution process.

We will benchmark two types of disruptions (and corresponding contingency plans) illustrated in Figures 6 and 7. We consider disruptions of length 1 and 4 hours with three different starting points: 9:00, 11:00 and 15:00. The selected periods overlap with the peak periods in different ways. The weekend instances, *Sat* and *Sun*, are omitted since they are both smaller and since the effect of activating a contingency plan is smaller.

A disruption occurs unexpectedly during the day and from this point in time a new modified timetable (c.f. the contingency plans) is followed. A forecast of the disruption duration is made which determines when the normal timetable can be used again. The expected time required to resolve the cause of the problem is essentially unknown and should be handled by a complete framework, however in the this paper (and others before) we will assume that the duration is

Instance	Penalty	Time	Best	Tree	Cols	Gap	Root	Cover	Mileage
Fri	0.00	163	11	72	1 942	0.8%	5.7%	99.0%	135.5%
Fri	0.01	148	14	62	2 744	0.7%	5.4%	99.1%	128.5%
Fri	0.10	71	19	41	2 354	0.7%	4.7%	99.0%	125.9%
Fri	1.00	182	14	69	2 815	1.0%	5.6%	98.5%	123.0%
Sat	0.00	2	2	1	469	0.0%	0.0%	99.6%	109.7%
Sat	0.01	3	2	0	668	0.0%	0.0%	99.6%	100.1%
Sat	0.10	3	3	0	727	0.0%	0.0%	99.6%	100.1%
Sat	1.00	3	3	0	647	0.0%	0.0%	99.6%	100.1%
Sun	0.00	2	2	1	368	0.0%	0.0%	99.5%	106.6%
Sun	0.01	2	2	0	575	0.0%	0.0%	99.5%	100.1%
Sun	0.10	3	2	0	656	0.0%	0.0%	99.5%	100.1%
Sun	1.00	2	2	0	633	0.0%	0.0%	99.5%	100.1%
Mon	0.00	75	11	40	1 918	0.4%	4.3%	98.9%	136.0%
Mon	0.01	100	17	46	2 606	0.9%	4.7%	98.8%	125.8%
Mon	0.10	99	16	46	2 955	1.0%	4.2%	98.7%	125.0%
Mon	1.00	86	18	39	2 833	1.0%	4.3%	98.6%	124.6%

Table 11: Shows the results obtained for the instances using increasing mileage cost penalties. A termination gap of 1% was set. The column show the instance, mileage penalty, runtime, when the best solution was found, B&B tree size, no of columns generated, optimality gap, root lp relaxation optimality gap, seat coverage and mileage cover.

known. Note, that if (or when) the forecast changes we assume that our framework is used to resolve the updated timetable. Therefore for simplicity and without loss of generality we solve and benchmark single disruptions where the disruption durations are given in advance. Up until the start of the disruption we assume that a optimized plan has been followed, which means that we know the location of all train units. In this paper the optimized plan has been obtained by solving the whole planning period (without disruptions) as shown in the previous section.

The results of the benchmarks are shown in Tables 12 and 13. The framework was set to terminate and report the best found solution within 5 minutes or when a optimality gap of 1% is proven. Note, that since we solve the remainder of the day this means that larger durations and later starting points result in smaller instances.

In the first Table 12 we see the results of activating the contingency plan *A311* with multiple duration and starting points. Within a few seconds we obtain very good solutions. Due to a decrease in the number of subtrips a low runtime is not unexpected. Again, the lower bound shows a noticeable improvement compared to the relaxation of the root node. In all cases the final solutions are found after solving the root node as a MIP. A full cover is obtained in all cases, which means that all trips in the disrupted timetable are possible to cover. The cover metric does not include the trips cancelled due to switching to the contingency plan. The mileage use and seat coverage is close to what we observed during planning.

In the second Table 13 we see the results of activating the contingency plan *A016* with multiple duration and starting points. Again, good solutions are found within a few seconds. Now the strength of the flow bound is more apparent than before, in the worst case the improvement is up to approximately 15% compared to the relaxation of the root node. In all cases the final solutions are found after solving the root node as a MIP. This change is probably correlated to reduced trip cover. We observe that some trips are now cancelled in the disrupted timetable. A fractional solution found in the root node relaxation is able to cover more trips, seats and use less couplings and mileage. The mileage usage is still comparable to our previous results. The slightly lower the seat cover is partly due to the necessity of cancelling trans but also due to a more turbulent fleet position.

					Quality				
	Disruption	#	Time	Cols	Gap	Root	Cover	Seat	Mileage
Fri	9:00-10:00	823	5	2 563	0.9%	3.4%	100.0%	99.0%	122.9%
Fri	9:00-13:00	814	6	2 608	0.0%	3.2%	100.0%	99.4%	124.8%
Fri	11:00-12:00	703	5	2 234	0.3%	3.8%	100.0%	99.1%	121.3%
Fri	11:00-15:00	692	5	2 483	0.3%	2.1%	100.0%	99.0%	122.7%
Fri	15:00-16:00	456	3	1 456	0.0%	0.8%	100.0%	98.9%	118.3%
Fri	15:00-19:00	442	3	1 362	0.0%	1.2%	100.0%	99.2%	118.8%
Mon	9:00-10:00	807	5	2 642	0.1%	2.6%	100.0%	98.9%	126.1%
Mon	9:00-13:00	798	6	3 391	0.6%	2.8%	100.0%	99.1%	126.3%
Mon	11:00-12:00	687	6	2 460	0.9%	4.0%	100.0%	98.8%	124.5%
Mon	11:00-15:00	676	5	2 854	0.5%	2.9%	100.0%	98.9%	121.6%
Mon	15:00-16:00	440	5	1 626	0.0%	2.4%	100.0%	98.2%	119.0%
Mon	15:00-19:00	426	3	1 416	0.7%	2.1%	100.0%	98.2%	121.1%

Table 12: Results of solving the first type of disruption case. The first columns show the instance, disruption duration, no of subtrips and runtime. *Cols* shows the no of generated columns. The next columns show the optimality gap w.r.t. the lower bound and root node relaxation. Finally, the last column measure the quality of the solution w.r.t. covered subtrips, covered seats and train unit mileage.

					Quality				
	Disruption	#	Time	Cols	Gap	Root	Cover	Seat	Mileage
Fri	9:00-10:00	836	11	4 096	0.3%	15.1%	99.8%	98.9%	122.4%
Fri	9:00-13:00	782	18	6 068	0.5%	14.7%	99.8%	98.7%	119.4%
Fri	11:00-12:00	713	8	3 051	0.6%	16.7%	99.8%	98.9%	122.1%
Fri	11:00-15:00	657	10	3 939	0.2%	14.8%	99.7%	98.4%	117.5%
Fri	15:00-16:00	468	6	2 379	0.3%	2.1%	99.6%	97.2%	113.8%
Fri	15:00-19:00	402	5	1 727	0.0%	1.9%	99.8%	98.4%	115.3%
Mon	9:00-10:00	820	13	4 129	0.4%	2.7%	99.8%	98.4%	121.6%
Mon	9:00-13:00	766	14	5 827	0.1%	2.5%	99.8%	98.8%	118.3%
Mon	11:00-12:00	697	10	3 665	0.5%	16.5%	99.8%	98.8%	122.4%
Mon	11:00-15:00	641	8	4 066	0.2%	14.3%	99.7%	98.4%	118.6%
Mon	15:00-16:00	451	6	2 239	0.0%	1.5%	99.6%	97.5%	119.1%
Mon	15:00-19:00	385	6	2 053	0.2%	1.9%	99.8%	97.2%	118.2%

Table 13: Results of solving the second type of disruption case. The first columns show the instance, disruption duration, no of subtrips and runtime. *Cols* shows the no of generated columns. The next columns show the optimality gap w.r.t. the lower bound and root node relaxation. Finally, the last column measure the quality of the solution w.r.t. covered subtrips, covered seats and train unit mileage.

6 Conclusions

In this paper we presented and investigated a Branch-and-Price framework for optimizing the Rolling Stock Rescheduling Problem (RSRP), and benchmarked both planned and disrupted real-life DSB S-tog instances.

The RSRP is essentially a multi-criteria optimization problem that is usually solved using a linear weighted objective. However, we observed that the objective must be carefully tuned in order to achieve the desired balance between operational cost, shunting and seat shortages. In this paper we found a balance that achieves a demand cover very close to 100%, while minimizing optional costs.

The B&P framework is able to solve the instances to optimality, but this proves to be time-consuming in certain settings. However as discussed, proven optimality is not essential since finding the correct objective weight balance is non-trivial. The benchmark revealed that we can find solution with a 1% proven optimality gap within 3 minutes for the hardest planning instances, and within 20 seconds for the disruption instances. In general, the test shows that the framework is able to find good solutions very fast, but on the hand not able to close the optimality gap efficiently. A MIP based lower bound efficiently improved the optimality gap.

In future research we will focus on several things. One obvious extension is to impose train composition and shunting restrictions. This would allow the train operators to more explicitly control how already running train compositions can be upgraded or reduced in order to improve robustness or feasibility. Another natural extension is to extend the subproblem to handle resource constraints in order to be able to model new types of constraints such as train unit maintenance restrictions. Finally, it would be interesting to extend the current models in order to model already used maneuvers at DSB S-tog during disruption. Examples include allowing trains to turn earlier than planned, and allowing two trains to swap services when meeting on the same station.

References

[1] E. Abbink, B. van den Berg, L. Kroon, and M. Salomon. Allocation of railway rolling stock for passenger trains. *Transportation Science*, 38:33–41, 2004.

[2] T. Achterberg, T. Koch, and A. Martin. Branching rules revisited. *OPERATIONS RESEARCH LETTERS*, 33(1):42–54, 2005.

[3] Arianna Alfieri, Rutger Groot, Leo Kroon, and Alexander Schrijver. Efficient circulation of railway rolling stock. *Transportation Science*, 40(3):378–391, 2006.

[4] Barnhart, Hane, and Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Oper. Res. (USA)*, 48(2):318–326, 2000.

[5] Cynthia Barnhart, Ellis L Johnson, George L Nemhauser, Martin WP Savelsbergh, and Pamela H Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998.

[6] Nejib Ben-Khedher, Josephine Kintanar, Cecile Queille, and William Stripling. Schedule optimization at sncf: from conception to day of departure. *Interfaces*, 28(1):6–23, 1998.

[7] Ralf Borndörfer, Markus Reuther, Thomas Schlechte, and Steffen Weider. A Hypergraph Model for Railway Vehicle Rotation Planning. In Alberto Caprara and Spyros Kontogiannis, editors, *11th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems*, volume 20 of *OpenAccess Series in Informatics (OASIcs)*, pages 146–155, Dagstuhl, Germany, 2011. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[8] Gabriella Budai, Gábor Maróti, Rommert Dekker, Dennis Huisman, and Leo Kroon. Rescheduling in passenger railways: the rolling stock rebalancing problem. *Journal of Scheduling*, 13:281–297, 2010. 10.1007/s10951-009-0133-9.

[9] Valentina Cacchiani, Alberto Caprara, Laura Galli, Paolo Toth, Leo Kroon, Gábor Maróti, and Leo Kroon. Railway rolling stock planning: Robustness against large disruptions. *Transportation Science*, 46(2):217–232, 2012.

[10] Luis Cadarso and Ángel Marín. Robust routing of rapid transit rolling stock. *Public Transport*, 2(1-2):51–68, 2010.

[11] Luis Cadarso and Ángel Marín. Robust rolling stock in rapid transit networks. *Computers & Operations Research*, 38(8):1131–1142, 2011.

[12] Luis Cadarso, Ángel Marín, and Gábor Maróti. Recovery of disruptions in rapid transit networks. *Transportation research. Part E, Logistics and transportation review*, 53:15–33, 2013.

[13] A. Caprara, L.G. Kroon, M. Monaci, M. Peeters, and P. Toth. Passenger railway optimization. In C. Barnhart and G. Laporte, editors, *Handbooks in Operations Research and Management Science*, volume 14, chapter 3, pages 129–187. Elsevier, 2007.

[14] Jean-François Cordeau, François Soumis, and Jacques Desrosiers. A benders decomposition approach for the locomotive and car assignment problem. *Transportation Science*, 34(2):133–149, May 2000.

[15] Jacques Desrosiers, Yvan Dumas, Marius M. Solomon, and François Soumis. Chapter 2 time constrained routing and scheduling. In C.L. Monma M.O. Ball, T.L. Magnanti and G.L. Nemhauser, editors, *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, pages 35 – 139. Elsevier, 1995.

[16] Pieter-Jan Fioole, Leo Kroon, Gábor Maróti, and Alexander Schrijver. A rolling stock circulation model for combining and splitting of passenger trains. *European Journal of Operational Research*, 174(2):1281–1297, 2006.

[17] Julie Groth, Daniel Potthoff, Jens Clausen, Dennis Huisman, Leo Kroon, Gábor Maróti, and Morten Nielsen. Disruption management in passenger railway transportation. In Ravindra Ahuja, Rolf Möhring, and Christos Zaroliagis, editors, *Robust and Online Large-Scale Optimization*, volume 5868 of *Lecture Notes in Computer Science*, pages 399–421. Springer Berlin / Heidelberg, 2009. 10.1007/978-3-642-05465-5_18.

[18] Julie Jespersen Groth and Jens Clausen. *Optimal Reinsertion of Cancelled Train Lines*. 2006.

[19] Julie Jespersen Groth, Daniel Potthoff, Jens Clausen, Dennis Huisman, Leo Kroon, Gábor Maróti, and Morten Nyhave Nielsen. Disruption management in passenger railway transportation. *Robust and Online Large-Scale Optimization*, pages 399–421, 2009.

[20] Leo Kroon and Dennis Huisman. Algorithmic support for railway disruption management. In Jo A.E.E. Nunen, Paul Huijbregts, and Piet Rietveld, editors, *Transitions Towards Sustainable Mobility*, pages 193–210. Springer Berlin Heidelberg, 2011.

[21] N. Lingaya, JF Cordeau, G. Desaulniers, J. Desrosiers, and F. Soumis. Operational car assignment at via rail canada. *Transportation Research Part B: Methodological*, 36(9):755 – 778, 2002.

[22] Lars Kjær Nielsen, Leo Kroon, and Gábor Maróti. A rolling horizon approach for disruption management of railway rolling stock. *European Journal of Operational Research*, 220(2):496–509, 2012.

[23] Marc Peeters and Leo Kroon. Circulation of railway rolling stock: a branch-and-price approach. *Computers & operations research*, 35(2):538–556, 2008.

[24] Daniel Potthoff, Dennis Huisman, Daniel Potthoff, Dennis Huisman, and Guy Desaulniers. Column generation with dynamic duty selection for railway crew rescheduling. *Transp Sci*, 44(4):493–505, 2010.

[25] Natalia J. Rezanova and David M. Ryan. The train driver recovery problem—a set partitioning based model and solution method. *Computers & Operations Research*, 37(5):845 – 856, 2010.

[26] Ryan, Foster, and Wren. An integer programming approach to scheduling. 1981.

[27] Alexander Schrijver. Minimum circulation of railway stock. *CWI QUARTERLY*, 6:205–217, 1993.