Optimizing the Cargo Express Service of Swiss Federal Railways*

Alberto Ceselli[1] **, Michael Gatto[2], Marco E. Lübbecke[3], Marc Nunkesser[2], and Heiko Schilling[4]

[1] Dipartimento di Tecnologie dell'Informazione, Università degli Studi di Milano, Italy,
ceselli@dti.unimi.it
[2] Department of Computer Science, ETH Zürich, Switzerland,
{michael.gatto|marc.nunkesser}@inf.ethz.ch
[3] Technische Universität Berlin, Institut für Mathematik, MA 6-1, Straße des 17. Juni 136, 10623 Berlin, Germany,
m.luebbecke@math.tu-berlin.de
[4] TomTom

Abstract. The Cargo Express service of Swiss Federal Railways (SBB Cargo) offers fast overnight transportation of goods between selected train stations in Switzerland and is operated as a hub-and-spoke system with two hubs. We present three different models for planning the operation of this service as a whole. All models capture the underlying optimization problem with a high level of detail: Traffic routing, train routing, make-up, scheduling, and locomotive assignment are all addressed. At the same time we respect hard constraints like tight service time windows and train capacities, and we avoid hub overloading. We describe our approaches for obtaining provably good quality solutions. Our algorithmic techniques involve branch-and-cut, branch-and-price as well as problem specific exact and heuristic acceleration methods. We conclude our study with computational results on realistic data.

Revision: Friday 21$^{\text{st}}$ September, 2007 at 23:44

1 Introduction

The planning of freight train operations comprises various difficult intertwined decisions on the strategic (long-term), tactical (mid-term), and operational (short-term) level. Fig. 1 depicts the tactical level and the operational level according to the survey article by Cordeau, Toth, and Vigo [13]. Considerable research has been devoted both to each single aspect of this process and towards integrating as many of these planning steps as possible. In general, the particular freight train system at hand determines how difficult and how important each single aspect is. For example, there are significant differences between the American and the European systems and even within a given country different systems with different focus are operated.

The Cargo Express service of Swiss Federal Railways (SBB Cargo) [42] considered in this paper serves as an example of a freight system: Cargo Express offers fast overnight transportation of goods between selected train stations in Switzerland and is operated as a hub-and-spoke system (as airlines do) with two hubs.

* This work was partially supported by the Future and Emerging Technologies Unit of EC (IST priority - 6th FP), under contract no. FP6-021235-2 (project ARRIVAL).

** Partially supported by a Marie Curie scholarship while visiting TU Berlin.

strategic	- - - -► "planning direction" - - - -►		tactical			operational
Freight Routing	**Blocking**	**Train Routing**	**Makeup**	**Schedule**	**Locomotive Assignment**	
Route cars through network Distribute classification work among yards	shipment → blocks (at each yard)	Route trains through network	block to train blockless: car to train	*coarse*: frequency of trains *fine*: arrival and departure of trains	engines → trains	Empty cars Dispatching, Classification schedules, Fitting of trains into timetable

Fig. 1. Mid and short term planning tasks in freight railroading

In parallel, SBB Cargo offers a “classical” freight train system, Cargo Rail, with looser time constraints and a larger service network.

We briefly describe the Cargo Express service. Its prominent features are the focus on fast transport, guaranteed pickup and delivery times for customers, a dense railroad network spanning a comparatively small area. While the blocking mode, in which cars are grouped and routed as a single *block* through the network (and thus not reclassified at intermediate classification yards), is popular in the U.S., SBB Cargo Express operates in non-blocking mode, as many European companies do. In fact, cars are reclassified in two central hump yards, the hubs. Each train can be composed by a limited number of cars. Finally, hub overloading must be avoided by limiting the number of cars in the yard at the same time.

These features give a distinguishing flavor to the planning steps and set the Cargo Express service off from most of the systems that have been considered in the literature and the results that have been obtained for these [3, 13, 15]. A classical paper is by Crainic, Ferland, and Rousseau [14], who study the interactions between blocking, routing, and makeup for a Canadian freight system. Ahuja, Jha, and Liu [2] give a particularly detailed model for the blocking problem arising in the U.S. Campetella et al. [8] consider an Italian freight service of size comparable to SBB Cargo Express, for which they do traffic routing, and planning of service frequency and empty cars, ignoring train load limits and yard capacities. None of these models is intended to produce a detailed schedule. Therefore, such models are inappropriate for planning the SBB Cargo Express service. Kwon et al. [29] describe how an existing solution consisting of blocking, routing and make-up plans and a target schedule can be adapted to meet the train load constraints. Gorman [23] extends a model of Keaton [28] to incorporate time constraints on a coarse time scale. He addresses the blocking, traffic routing, makeup and scheduling problem for a given set of candidate routes. He proposes a tabu search approach which he tests on a U.S. instance. On this instance he finds cost savings with respect to a solution used in practice. However, the method produces operating plans that can violate constraints on the load of the trains and on time windows. In contrast, in the SBB Cargo Express setting both time windows and load constraints are hard, and the time windows are tighter.

In this paper, we present three different ways to exactly model the SBB Cargo Express service, all of which capture the optimization problem at hand with a high level of detail. Our models span the whole tactical planning process: Traffic routing, train routing, make-up, scheduling, and the basic engine (locomotive) assignment are all addressed. On the other hand we do not model the operational level, i.e., our models do not capture empty cars movements, the precise shunting operations in the classification yards, minimum buffer times between arriving trains at the yard, or the fitting of trains into the timetable. As we model an overnight service, the latter aspect is not as crucial as for other freight systems.

In the first approach, we provide a compact integer linear programming (ILP) formulation of the problem, and apply a state-of-the-art general purpose solver to it. In the second approach we hierarchically decompose the problem as suggested by Fig. 1 and provide separate models for distributing the classification work, for the combination of routing, makeup, locomotive assignment, and finally for scheduling in our setting. We develop a branch-and-cut approach for its hardest subproblem. Our third approach is to formulate the entire planning problem as a side-constrained set partitioning problem, and solve it using column generation. The integration of the above planning steps leads to a complex master problem. To the best of our knowledge, this model is the first to integrate all planning steps of the tactical level. Although we tailored the solution approach to our specific problem, the model itself is applicable with minor modifications to any non-blocking freight system of larger scale.

Our solution methods span different algorithmic techniques generally used for ILPs. Moreover, the three approaches have an increasing level of modeling detail, and require algorithmic techniques of increasing complexity. We carry out a comparison of the three methods in order to identify the best trade-off between modeling detail, computational tractability, and implementation effort.

All experiments are performed on data derived from a typical day of operation of the SBB Cargo Express service, provided by SBB Cargo.

The paper is organized as follows. First, we give a detailed problem description. Then, we introduce our three models, and next describe the corresponding solution methods. Finally, we report on computational experiments, comparing the running time of the algorithms and the quality of the solutions found.

Problem Description The SBB Cargo Express system is designed for customers who regularly need overnight transportation of freight by train: a typical customer orders a regular transport of shipments from an origin to a destination station. The customer announces her fixed daily demand for the lifetime of the schedule in advance and negotiates an earliest departure time and a latest delivery time for the shipments with SBB Cargo. Once a year SBB Cargo designs a new operating plan and a schedule that accommodates all customer demands.

The transport itself works as follows. In the evening, the customers deposit their cars at the departure stations before the earliest departure times. During the night, a fleet of trains collects all deposited cars. In general, each train of the fleet collects several shipments at different stations. The cars of the same shipment are always transported together, different shipments at the same station may be transported by different trains. In addition to the shunting time needed at the hubs, a substantial amount of time is spent in brake testing at each stop which involves pickup or delivery.

The SBB Cargo Express network is operated as a hub-and-spoke system: the fleet transports all the goods to central classification yards in Däniken and Zürich-Mülligen where inbound trains are reclassified to form outbound trains. Outbound trains either go directly to the other hub nearby without en route pickup or delivery, or they deliver their cars at the respective delievery stations. Finally, direct trains between origin and destination stations are also possible.

The service is operated such as to guarantee the negotiated earliest pickup and latest delivery times.

Since trains are composed by few cars, each engine of the fleet operates in the same way: That is, the planners consider the fleet to be homogeneous. Each engine of the fleet can perform only one of the tasks described above in one night, i.e., either going to and from a hub once including potentially a few rides between the nearby hubs, or transporting a shipment directly to its destination. An important constraint of this system is the yard capacity: Only a limited number of cars can stay in each hub concurrently and, most of all, hub overloading affects reclassification time.

The main costs of an operating plan consist in operating the engines, that is employing the drivers and servicing the equipment. In addition to this fixed cost component, there is a cost per kilometer. The overall goal is to minimize the total fixed and variable cost of an operating plan.

A schedule is determined and operated with minor daily changes for a whole year. Currently, the planners do not use any decision support system: they construct the schedule by hand and slightly adapt it in a trial period after its implementation. If demands change over a year or new customers want to be served, the SBB Cargo team generally succeeds in adapting (again by hand) the existing schedule to the new situation.

2 Models

Capturing all the characteristics of the SBB Cargo Express system in a single model is a challenging task. In this section we first lay the ground for a precise mathematical description of the problem by introducing the necessary common notation. Then we present a suite of three models for our application. The first one, Model 0, is an ILP formulation for the problem, neglecting hub overloading issues and forcing exactly one reclassification step for each shipment. In Model 1 we decompose the problem into three consecutive planning steps which we treat separately. Model 2 is a set partitioning integer linear program that exploits all the optimization degrees of freedom by considering all the decision levels at once.

2.1 Common Notation

We are given a *(railroad) network* $N = (V, E, \ell)$. The node set V represents stations, hubs, and junctions, the edge set E represents the tracks connecting those; $\ell : E \to \mathbb{R}^+$ is a *length function* on the edges. In our models we consider the problem in which an arbitrary number of nodes of the network represent hubs, and we denote the set of hubs by $H \subseteq V$. In the following we give a list of parameters and features that we consider.

– The set S of *shipments* contains an element for each transportation request. Each shipment $s \in S$ has the following properties:

  - orig(s): the *origin station*,
  - dest(s): the *destination station*,
  - depart(s): the *earliest possible pickup time* at station orig(s),
  - arrive(s): the *latest possible delivery time* at station dest(s),
  - vol(s): the number of cars (volume) composing s.
- The *maximum train load* $L_{\max}$ bounds the total number of cars that any engine can pull.
- The *shunting time at a hub* T^h_{shunt} is the additional shunting time an outbound train has to wait before departing from hub $h \in H$ after its last shipment has arrived. This time is assumed to be independent of the number and volume of the shipments. Similarly, an engine needs T^h_{shunt} time units to be decoupled from an inbound train and coupled to an outbound train.
- The *couple time at the stations* T^v_{couple} is the additional time incurred by picking up or delivering any set of shipments at a station $v \in V$. It is independent of the number and volume of the shipments.
- The *hub capacity* cap$_h$ specifies, for each hub $h \in H$, the maximum number of cars that can concurrently stay at h without overloading the hub.
- The *engine cost* C_{engine} represents the operating cost of one engine (fixed cost).
- The *average speed* $\bar{v}$ is used to calculate the traveling times of trains on the tracks.
- The *kilometer cost* $\bar{c}$ represents the travel cost of an engine per kilometer (variable cost).

The fundamental part of a solution is a *route* of a train through the network. By route we mean a graph theoretic walk, which can in particular contain repeated edges and nodes, and a set of shipments served at the visited nodes. We distinguish between pickup and delivery routes, respectively bringing shipments from the stations to the hubs or from the hubs to the stations, and hub-connecting routes, which move cars between two hubs. Given $v \in V$, we abuse notation slightly and write $v \in r$ to indicate that route r visits node v; similarly, given $s \in S$ we write $s \in r$ to indicate that shipment s is served by route r. The length l_r of a route r is the sum of the lengths of the (possibly repeated) edges used in r, and the cost c_r of the route is computed as $\bar{c} \cdot l_r$. The volume vol(r) of a route r is the sum of volumes of the shipments served by r. We call r *volume-admissible* if it satisfies the load constraint: $\sum_{s \in r} \text{vol}(s) = \text{vol}(r) \leq L_{\max}$.

A complete solution to our problem consists first of sets of pickup routes R^x, delivery routes R^y, and hub-connecting routes R^h. Second, it specifies arrival and departure times arrive(r, v), depart(r, v) at each node $v \in r$ for these routes. We call a route together with such timing information a *scheduled route*. If a scheduled route respects all time windows and the couple time T^v_{couple} we call it *time-consistent*. A scheduled pickup route r^x to hub h is *compatible* with a scheduled delivery route r^y from h if either r^x and r^y have no common shipment: $\nexists s \in S : s \in r^x \wedge s \in r^y$ or they refer to the same hub and respect the precedence including shunting time at the hub: $\text{depart}(r^y, h) \geq \text{arrive}(r^x, h) + T^h_{\text{shunt}}$. The same notion of compatibility analogously applies to connections involving hub-connecting routes. Formally, our railroad problem is defined as follows:

Definition 1 (Multi Hub and Spoke Optimization Problem (MHSOP)). *Given a railroad network $N = (V, E, \ell)$, a set of hubs $H \subseteq V$, a set of shipments S and the parameters $L_{\max}$, T^h_{shunt}, T^v_{couple}, cap$_h$, C_{engine}, $\bar{v}$ and $\bar{c}$ as defined above, find a feasible solution of minimum cost. A feasible solution consists of the size k of the necessary engine fleet and a set of pairwise compatible, scheduled, time-consistent, volume admissible routes, such that all shipments are transported from their respective origin to their respective destination, the hub capacity limits are respected, and the routes can be driven by k engines. The cost of a solution is given by the sum of costs of the selected routes plus the cost of operating k identical engines.*

MHSOP is strongly NP-hard as it contains problems like the traveling salesman, bin-packing, and diverse scheduling problems [21]. This justifies computationally expensive approaches like mixed integer programming.

2.2 Model 0: Edge-Based Classical Vehicle Routing Model

ILP formulations have proven to be a powerful and versatile tool for modeling the whole host of NP-hard problems. In addition, as long as standard solvers can be applied, they represent an attractive choice from an implementation cost point of view. Therefore, we first evaluated the potential of such approaches.Toth and Vigo discuss various formulations for simpler but related vehicle routing problems (VRP), like the capacitated VRP and the VRP with time windows [43]. In the following, we adapt and extend one classical formulation therein to model MHSOP.

Let $\bar{K}$ be an upper bound on the number of engines used in an optimal solution. Clearly, $\bar{K} \leq |S|$. We build a three-index vehicle flow formulation [43] for MHSOP, which uses $O(|E| \cdot \bar{K})$ integer variables u_{et}, each counting the number of times track $e \in E$ is traversed by a train with engine t, and $O(|S| \cdot \bar{K} \cdot |H|)$ binary variables v^h_{st}, each taking value 1 if shipment s is served over hub h by a train with engine t. Moreover, in order to handle time windows, we introduce $O(\bar{K} \cdot |V|)$ non-negative variables w_{tv}, each representing the arrival time of engine t at station v. These sets of variables allow the formulation with a polynomial number of constraints.

However, the use of multiple hubs introduces several complicating issues. First, pickup and delivery routes require different sets of variables; the hub-connecting routes have to be modeled, needing $O(|S| \cdot \bar{K} \cdot |H|^2)$ binary variables $h_{s,t,\text{start}(t),\text{end}(t)}$. Furthermore, delivery routes and pickup routes must be compatible which makes further $O(\bar{K} \cdot |S| \cdot |H|)$ binary variables necessary to model the dependency between in- and outbound trains. Direct connections further complicate the model. Finally, in this formulation we do not model hub overloading issues. The complete model is available in [9]

The resulting formulation is “compact” in the sense that it involves only a polynomial number of variables and constraints. The approach proved to be very useful to obtain feedback from our partners quickly, gain solid understanding of the problem and to evaluate the optimization potential. However, the approach is completely impractical already on very modestly sized instances, as we will see in Section 4.1.

2.3 Model 1: Hierarchical Decomposition: Cluster-First, Route-Second, Schedule-Third

A natural approach is to follow the planning process depicted in Fig. 1 and translate it to the particular problem at hand. For SBB Cargo Express, this means: First, *partition* the shipments and the engines among the hubs, second *route* trains separately for the hubs, third *schedule* the routes. In this section, we give one model for each step. As with most such hierarchical approaches it is important to robustly design the objectives and constraints of the models for the early steps such that they anticipate feasibility problems that might occur in later steps.

Partition The partitioning is performed with three objectives: keep the average travel time of each shipment short, reclassify shipments from or to the same station in the same hub, and balance the classification work among the hubs (as in the classical blocking problem).

Since too much traffic at a hub may yield infeasible scheduling problems, we treat load balancing as a hard constraint. Then, we trade off the two remaining objectives by computing a kilometer equivalent cost for connecting each station to a hub. The resulting problem is a variation of a facility location problem having a load balancing constraint.

Routing The partitioning step creates $|H|$ single-hub subproblems which we route separately, i.e., we do not plan dedicated hub-connecting routes. As solely the routing and not the schedule is decided in this step, one cannot fully foresee the compatibility of generated routes and the feasibility for the size of the engine fleet. Still, it is necessary to take these aspects into account. To this end, we a priori fix the number of engines to a value K to guarantee that the same number of engines enter and leave the hub. Second, we impose a maximum distance constraint on both pickup and delivery routes to avoid that long routes are constructed. The reason for this is that too long routes cannot be scheduled in a compatible way.

An advantage of this approach is that the routing now further decomposes into routing of pickup and of delivery trains. The two routing problems are symmetric; hence we only give a definition of the pickup problem.

Definition 2 (Pickup Train Routing Problem with fixed train fleet (TRP)). *Given a network $N = (V, E, \ell)$, a specified hub node $h \in H \subseteq V$, a set of shipments S_h, a maximum train load $L_{\max}$, a maximum* *trip distance $D_{\max}$, the average speed $\bar{v}$, the couple time T^v_{couple}, and the fleet size K. A feasible solution to the TRP problem consists of a set of K volume admissible* routes *R^x each ending in h such that the following properties hold:*

1. *For each shipment $s \in S_h$, there is a route $r \in R^x$ serving s: $\forall s \in S_h \; \exists r \in R^x : s \in r$.*
2. *No route is longer than $D_{\max}$. The* length *of a pickup route $r \in R^x$ is defined as the length of the route plus the following term accounting for the coupling time: $|\{v \in r \mid \exists s \in r : orig(s) = v\}| \cdot T^v_{couple} \cdot \bar{v}$.*

The cost of a solution is the sum of the lengths of the routes. The TRP asks for a solution of minimum cost.

In Section 3.1 we discuss a branch and cut approach for this TRP.

Scheduling A complete schedule for given solutions R^x for the pickup TRP and R^y for the delivery TRP specifies the departure and arrival times of each route at each station. However, it is not necessary to specify a schedule in such detail: The arrival time and departure time windows are one-sided, in the sense that there is a priori no latest pickup time or an earliest delivery time for the shipments. For this reason, it is never convenient for a train to slow down on the tracks or to wait outside a station until it is possible to enter it. Therefore, we can completely specify a schedule by giving the arrival and departure times of the routes at the hub and assume w.l.o.g. that the trains travel in the fastest possible way according to $\bar{v}$ and T^s_{couple}.

The objective of the scheduling step is to minimize the maximum hub load:

Definition 3 (Train Shunting and Scheduling Problem (TSSP)). *Let the solutions R^x and R^y to the corresponding pickup and delivery TRPs for a set S of shipments via a hub h be given. A feasible solution to TSSP defines an arrival time arrive(r^x, h) for each $r^x \in R^x$ and a departure time depart(r^y, h) for each $r^y \in R^y$ such that:*

1. *the (inferred) arrival and departure times of each route at the stations are time consistent, i.e., they respect the time windows of the shipments.*
2. *All routes are compatible w.r.t. the chosen arrival and departure times at the hub.*
3. *There are always enough outbound engines available:*

$$\left|\{r^x \in R^x \mid arrive(r^x) \le t\}\right| \ge \left|\{r^y \in R^y \mid depart(r^y) \le t + T^h_{shunt}\}\right| \quad \forall t \in \mathbb{R}_+. \tag{1}$$

The cost of a solution equals the maximum number of cars that are in the hub at the same time:

$$\max_{t \in \mathbb{R}_+} \left\{ \sum_{r^x : arrive(r^x) \le t} vol(r^x) - \sum_{r^y : depart(r^y) \le t} vol(r^y) \right\} \tag{2}$$

An optimal solution to TSSP is one for which this cost is minimum.

We remark that the number of relevant constraints in (1) is finite and that the maximum in (2) can be computed over a finite set of *relevant* points in time, for example the arrival and departure times of the trains at the hub plus the times derived by adding or subtracting the shunting time.

In this model we treat the capacity limit as a *soft* constraint by declaring it as the objective. In fact, as sketched in Section 1 it can be argued that the capacity of a classification yard is not determined by a single number of cars which if exceeded renders the yard inoperable, but rather by a range where it increasingly gets more and more difficult to fulfill the requested classification.

Model 1 has some limitations: First, as in all such approaches, optimization potential is lost in the hierarchical planning process. Second, on instances with tight time windows and capacity constraints, the separation between routing and scheduling may lead to feasibility issues in the scheduling step, because the routing is oblivious to the compatibility and time-consistency problems its routing decisions may create. A further restriction of Model 1 is that it neither considers hub-connecting nor direct trains. These limitations are discussed in some more detail in Sections 4.2 and 4.4.

2.4 Model 2: Path Based Set Partitioning Model

When tight time windows are imposed, it may be impossible to ensure an even load at the hubs using Model 1. In order to overcome this problem and exploit the additional optimization potential of direct connections and hub connections, we present a model that describes the MHSOP as a whole.

A scheduled route is the fundamental notion in our problem, and it is only natural to base a model on this. A variable in this path-based approach represents a time consistent, load admissible, scheduled route, of which there are exponentially many. We select compatible pairs in such a way that a usage limit on each hub is respected. The resulting ILP for MHSOP is given in Fig. 2. In order to deal with a finite number of constraints and variables in the first place, we consider a set $T = \{0, \ldots, \tau - 1\}$ of τ points in time, and we define as time slots each interval between subsequent points. We assume trains to always arrive or leave the hubs at the ends of a time slot. This is a common and mild restriction, as we do not have full control over the precise times anyway.

In this model, start(r) and end(r) denote the start and end node of a scheduled route r. Each binary variable x_r^t indicates whether the scheduled pickup route r, arriving at hub end(r) during time slot t, is selected or not; similarly, the binary variable y_r^t indicates whether the scheduled delivery route r, leaving hub start(r) during time slot t is selected or not. Moreover, each binary variable $h_r^{\text{a}:t}$ indicates whether the hub-connecting scheduled route r, departing from hub start(r) and arriving at the destination hub end(r) at time a:t is selected or not. We may refer to one and the same variable also by its departure time d:t from its origin hub. Following the policy of SBB Cargo we do not consider pickups or deliveries on hub-connecting routes. Finally, for each shipment $s \in S$ we introduce a binary variable d_s modeling the possibility of transporting s with a dedicated engine directly from its origin to its destination. Such a direct path is not associated to any time since it is always possible to deliver a shipment on the direct path respecting the time windows.

Abusing notation slightly, we denote the set of all scheduled routes by R, irrespective of their type; furthermore, all summations in Figure 2 are meant to be over feasible routes separately for the summands. That is, a sum of type $\sum_{r \in R, t \in T} x_r^t + h_r^{a:t}$ is to be read as the sum of all variables corresponding to feasible pickup routes with arbitrary arrival time at any hub plus the sum of all variables corresponding to feasible hub-connecting routes with arbitrary arrival time at any hub. All sums of this type in the constraints should be seen as a shorthand notation for two separate sums. For later use for the corresponding dual variables, we refer to the constraints using the Greek letters indicated at the left of the model.

Constraints labeled $\hat{\pi}$ and $\check{\pi}$ are set partitioning constraints stating that each shipment has to be picked up and delivered, respectively.

The ϕ-constraints are global flow conservation constraints for the shipments at the hubs. Together with the σ-constraints they ensure the time consistent inflow-outflow of shipments at hubs.

The $\hat{\sigma}$-constraints enforce that a shipment that arrives after time t has a corresponding outbound train after time $t + T_{\text{shunt}}^h$. The $\check{\sigma}$-constraints represent the symmetric statement for outbound trains.

The β-constraints play a similar role for the engines as the σ-constraints do for the shipments, except that we allow engines to end their duty at the hub. Together, the σ and β constraints enforce the compatibility of the chosen scheduled routes.

The χ-constraints limit the usage for each hub $h \in H$ to cap$_h$ cars in each time slot. We remark that introducing these constraints provides an a-priori guarantee to avoid hub overloading. With this model the decision maker can evaluate different scenarios allowing for different levels of maximum hub load.

Out of the three types of constraints $\check{\sigma}$, $\hat{\sigma}$, and ϕ, every pair of types implies the third type. Therefore, we chose to discard constraints $\check{\sigma}$. This implication was expected, since $\hat{\sigma}$ and $\check{\sigma}$ constraints are nothing else than a variation of the classical *generalized flow conservation constraints for networks with intermediate storage* for flows over time problems, see for example [24].

We charge the engine cost to the pickup route; together with the β constraints this allows to use an engine for pickup operations only.

$$
\begin{array}{llll}
\min & \sum\limits_{r\in R, t\in T} \tilde{c}_r x_r^t + \tilde{c}_r y_r^t + \tilde{c}_r h_r^{\text{d:}t} + \sum\limits_{s\in S} \tilde{c}_s d_s & & \\
(\hat{\pi}_s) & d_s + \sum\limits_{t\in T, r\in R: s\in r} x_r^t & = 1 & \forall s \in S \\
(\check{\pi}_s) & d_s + \sum\limits_{t\in T, r\in R: s\in r} y_r^t & = 1 & \forall s \in S \\
(\phi_{sh}) & \sum\limits_{t\in T, \text{end}(r)=h, s\in r} x_r^t + h_r^{\text{a:}t} & & \\
 & - \sum\limits_{t\in T, \text{start}(r)=h, s\in r} y_r^t + h_r^{\text{d:}t} & = 0 & \forall s \in S, h \in H \\
(\hat{\sigma}_{hst}) & \sum\limits_{\substack{r: s\in r, \text{end}(r)=h \\ t_1 \geq t}} x_r^{t_1} + h_r^{\text{a:}t_1} & & \\
 & - \sum\limits_{\substack{r: s\in r, \text{start}(r)=h \\ t_2 \geq t + T^h_{\text{shunt}}}} y_r^{t_2} + h_r^{\text{d:}t_2} & \leq 0 & \forall t \in T, h \in H, s \in S \\
\Big[(\check{\sigma}_{hst}) & \sum\limits_{\substack{r: s\in r, \text{start}(r)=h \\ t_1 \leq t}} y_r^{t_1} + h_r^{\text{d:}t_1} & & \\
 & - \sum\limits_{\substack{r: s\in r, \text{end}(r)=h \\ t_2 \leq t - T^h_{\text{shunt}}}} x_r^{t_2} + h_r^{\text{a:}t_2} & \leq 0 & \forall t \in T, h \in H, s \in S \Big] \\
(\chi_{th}) & \sum\limits_{\substack{t_1 \leq t \\ \text{end}(r)=h}} \text{vol}(r)\, x_r^{t_1} + \text{vol}(r)\, h_r^{\text{a:}t_1} & & \\
 & - \sum\limits_{\substack{t_2 \leq t \\ \text{start}(r)=h}} \text{vol}(r)\, y_r^{t_2} + \text{vol}(r)\, h_r^{\text{d:}t_2} & \leq \text{cap}_h & \forall t \in T, h \in H \\
(\beta_{ht}) & \sum\limits_{\substack{r\in R, \text{end}(r)=h \\ t' \leq t}} x_r^{t'} + h_r^{\text{a:}t'} & & \\
 & - \sum\limits_{\substack{r\in R, \text{start}(r)=h \\ t' \leq t + T^h_{\text{shunt}}}} y_r^{t'} + h_r^{\text{d:}t'} & \geq 0 & \forall t \in T, h \in H \\
 & x_r^t, y_r^t, h_r^{:\,:t}, d_s \in \{0, 1\} & & \forall r \in R, t \in T, s \in S
\end{array}
$$

Fig. 2. ILP Model 2; see comment in the text for interpreting the summations.

Finally, we remark that shipments with a hub as an origin or destination need a special treatment. Although this aspect has a minor impact on the model, it introduces particular subtleties in the solution methods, which we do not address here.

The advantages of this formulation are that it considers both shipment and engine flows in one model, and that the flow conservation carries over to the fractional variables: this means that all fractionally valued routes in a solution have compatible counterparts, which simplifies the extraction of integral solutions.

Furthermore, using this decomposition, we can efficiently solve the issue of time consistency and volume admissibility in independent subproblems, as described in Section 3.2.

3 Solution Methods

In this section we investigate the various algorithmic challenges inherent in the three models: What are the (sub-)problems that arise, how do we address them both algorithmically and implementation-wise, and what further techniques do we use to achieve a good performance. Since Model 0 is meant to be optimized using a (commercial) standard solver, we focus on Model 1 and Model 2.

3.1 Model 1: A Branch-and-Cut Approach

The result of the decomposition approach in Model 1 is that the MHSOP is reduced to a partitioning problem, two independent TRPs, and a TSSP for each hub. We consider the problems in this order.

Partitioning. The partitioning problem of Section 2.3 can be formulated as follows.

We introduce binary variables θ_{sh}, indicating whether shipment s is served by hub h, and binary variables p_{vh} and d_{vh}, indicating whether hub h is connected to station v by a pickup or delivery route, respectively.

Let C_{vh}^{connect} be the cost of connecting hub h to station v, and q_{sh} the cost of the cheapest trip for transporting the shipment s over hub h.

The problem of partitioning the set of shipments, such that no hub serves more than a fraction f of the total volume, can be formulated as follows.

$$\min \sum_{v\in V, h\in H} C_{vh}^{\text{connect}} \cdot (p_{vh} + d_{vh}) + \sum_{s\in S, h\in H} q_{sh} \cdot \theta_{sh}$$

$$\text{s.t.} \quad \sum_{h\in H} \theta_{sh} \geq 1 \qquad \forall s \in S \tag{3a}$$

$$\theta_{sh} \leq p_{vh} \qquad \forall h \in H, \forall s \in S, v \in V : \text{orig}(s) = v; \tag{3b}$$

$$\theta_{sh} \leq d_{vh} \qquad \forall h \in H, \forall s \in S, v \in V : \text{dest}(s) = v; \tag{3c}$$

$$\sum_{s\in S} (vol(s) \cdot \theta_{sh}) \leq f \cdot \sum_{s\in S} vol(s) \qquad \forall h \in H \tag{3d}$$

Constraints (3a) enforce that all shipments are routed via a hub. Constraints (3b) and (3c) enforce consistency of hub assignments and routes: If a shipment s is routed via hub h, the origin and the destination station must be connected to h. Constraint (3d) enforces the volume balance for each hub. Here, we choose C_{vh}^{connect} as the combination of a fixed offset for connecting a hub to a station, plus the connection costs in terms of distance from the hub.

This problem is NP-complete, as it contains problems like three-partition and facility location [4, 21]. However, this formulation can be effectively optimized by a general purpose solver (see Section 4).

Routing Our solution approach to the TRP consists of a transformation to the following well-known Distance constrained Capacitated Vehicle Routing Problem, which allows us to use existing reliable software [40].

Definition 4 (Distance constrained Capacitated Vehicle Routing Problem (DCVRP)). *Given a complete network $N^{DCVRP} = (V, E, c, \ell)$, where c is a cost function on edges and ℓ is a length function on edges, a specified hub node $\hat{h} \in V$, demands $d_i, i \in V$ on the nodes, a maximum load $L^{DCVRP}_{\max}$, and a maximum distance $D^{DCVRP}_{\max}$.*

Find K elementary circuits starting in $\hat{h}$ with minimum total cost, such that each customer node is visited by exactly one circuit, the sum of the demands on each circuit does not exceed the load $L^{DCVRP}_{\max}$, and that no circuit exceeds the length $D^{DCVRP}_{\max}$.

In the following, we give a transformation Ψ of TRP into DCVRP, such that an optimal solution of any TRP instance I_{TRP} can be derived from an optimal solution of the corresponding transformed DCVRP instance $\Psi(I_{\text{TRP}})$.

An optimal solution to the routing problem consists of two parts, the pickup routes and the delivery routes. For simplicity, we again describe the transformation for the pickup case only.

Roughly speaking, the task of the transformation is to do the following: Translate a problem defined on a sparse graph for which the solution consists of a set of circuits covering the network to a problem on the complete graph for which the solution consists of a set of paths covering the network. Note that the common transformation $c_{\{i,j\}} \longleftarrow c_{\{i,j\}} - c_{\{\hat{h},i\}} - c_{\{\hat{h},j\}}$ by Clarke and Wright savings [12] only works in the other direction, in the sense that it transforms a problem with circuits into a problem with paths. Moreover, we have to correctly translate the length and load constraints.

The transformation Ψ applies the following types of modifications to instance I_{TRP} to achieve the above goals:

1. Add all missing edges to N^{TRP}. The length of such a new edge $e = (u, v)$ is set to the length of the shortest u, v path in N^{TRP}.
2. Increase the length of each edge of the network by $T^s_{\text{couple}} \cdot \bar{v}$.
3. Partly merge shipments with identical origin that will definitely be transported by the same train.
4. Replace each station with j shipments, $j > 1$, by a j-clique with edges of cost and length zero. Identify each shipment of such a node with one of the nodes of the new clique by assigning the shipment's volume to the demand of the clique-node.
5. Add K nodes with demand M' to the graph. The nodes are connected to the hub by edges of weight $-M$, and to the rest of the network by the full bipartite graph with zero-weight edges.

Step 1 to 4 aim at making the network complete, have the couple time included in the distances, and have one shipment for each node. The idea behind the K extra nodes introduced in Step 5 is to force all circuits to start or end with such a node followed by a "free jump" to the starting node of the corresponding TRP path. Therefore, each of the K vehicles must visit exactly one of these nodes. This is achieved by setting the weight of the hub-extra node edges to $-M, M \gg 0$, their demand to $M' > \sum_{s \in S} \text{vol}(s)$, $L^{\text{DCVRP}}_{\max} = M' + L_{\max}$, and $D^{\text{DCVRP}}_{\max} = D_{\max} - M$.

The correctness of transformation Ψ is established in the following lemma, whose proof can be found in [22].

Lemma 1. *Let a TRP instance I_{TRP} be given. Let σ_{DCVRP} be an optimal solution to the DCVRP instance $\Psi(I_{TRP})$ of cost c. Then, an optimal TRP solution for the pickup has cost $c + K \cdot M$ and can be reconstructed from σ_{DCVRP} in linear time. The same statement holds for an optimal solution for the delivery TRP.*

Given an optimal solution of an alike created instance of DCVRP, an optimal solution of the TRP is constructed by considering the paths restricted to the nodes of the original network.

The construction can be slimmed down a bit: first, it is sufficient that the set of K nodes introduced can induce a matching on any subset $Q \subset V$ of size K. Second, for the original graph nodes, all edges can be removed, for which picking up both shipments of the incident nodes causes a violation of the load or length constraints.

Solving the DCVRP. The above construction allows us to focus on solution approaches for DCVRP.

Many exact algorithms for the DCVRP were proposed in the 1980s [11, 30, 31]. There are several software packages for the general vehicle routing problem, commercial as well as free ones, see [6, 25, 38] for a survey. A reliable and successful open source package for the VRP is the branch and cut code by Ralphs et al. [40]. We base our implementation on this package, and thus describe the "two index formulation" of the DCVRP on complete undirected graphs [44], which we use to extend the VRP package.

$$\textbf{DCVRP:}\ \min \sum_{e\in E} c_e x_e$$

$$\text{s.t.} \sum_{e=\{i,j\}\in E} x_e = 2 \qquad \forall i \in V \setminus \{\hat{h}\} \tag{4a}$$

$$\sum_{e=\{\hat{h},j\}\in E} x_e = 2K \tag{4b}$$

$$\sum_{e=\{i,j\}\in E, i\in Q, j\notin Q} x_e \geq 2r(Q) \qquad \forall Q \subset V \setminus \{\hat{h}\}, Q \neq \emptyset \tag{4c}$$

$$x_e \in \{0,1\} \qquad \forall e \in E \tag{4d}$$

Binary variables x_e indicate whether a given edge $e \in E$ is chosen. Equations (4a) and (4b) enforce the correct degree at the nodes and at the hub, respectively.

Constraints (4c), the *capacity cut constraints*, play a similar role for the VRP as the subtour elimination constraints do for the TSP [32]. The left hand side, evaluated at a solution vector, gives the number of edges in that solution that cross the graph theoretic cut $[Q, V \setminus Q]$. Note that every vehicle serving customers in Q contributes with two edges to the size of the cut. The right hand side should therefore represent the minimum number of necessary crossings of vehicles due to the connectivity requirement, capacity reasons, and the distance constraints. The value $r(Q)$ can be understood as the maximum of two values: $d(Q)$, which accounts for the maximum distance constraints; and $\lambda(Q)$, which accounts for the capacity constraints (and also for the connectivity constraints).

There are several valid but not equivalent choices for a definition of $d(Q)$ and $\lambda(Q)$. In fact, there is a whole hierarchy of possible values for $\lambda(Q)$ that lead to different families of valid inequalities—the most common being $\lambda(Q) = \left\lceil \frac{\sum_{v\in Q} d_v}{L_{\max}} \right\rceil$. This gives the *rounded capacity inequalities*, see [36] for a more detailed discussion. The value $d(Q)$ is the minimum value $k \in \mathbb{N}$ such that the objective value v^k_{TSP} of a k-TSP problem on Q divided by $D^{\text{DCVRP}}_{\max}$ and rounded up equals k, see [44]:

$$d(Q) = \min \left\{ k \in \mathbb{N} \;\middle|\; k \geq \left\lceil \frac{k\text{-TSP}(Q)}{D^{\text{DCVRP}}_{\max}} \right\rceil \right\} . \tag{5}$$

Separation Heuristics The separation problem for rounded capacity inequalities and inequalities of Type (4c) is NP-complete. For this reason, we focus on effective *separation heuristics* that try to find violated inequalities of Type (4c) without guaranteeing of finding one if it exists. As the cutting plane generation is embedded into a branch and bound framework, this might yield weaker bounds, but does not compromise the correctness of the algorithm.

Since the capacity constraints are effectively handled by the existing software package we use, we focus on the distance constraints. We devised the following separation heuristics.Given a solution, we consider the support graph induced by all nonzero edge variables. Then, we remove the hub node and consider each connected component Q_i separately.

Since we are only interested in instances of DCVRP arising from our transformation Ψ, we tuned our cuts to handle these specific instances, thus discarding the nodes introduced in Step 5 and the involved edges of size $-M$ in the length computation. Nevertheless, the cuts can be applied to the general DCVRP. For the first set of cuts, we compute the length of each component by weighing the length of each edge by the value

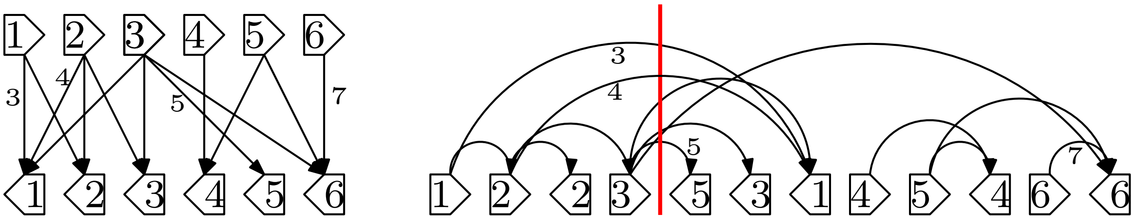

Fig. 3. An in-out graph together with a linear arrangement of cutwidth 15. All non-labeled edges have unit weight.

of its associated variable x_e. If the resulting length exceeds what can be traveled by the vehicles serving the component, we know that at least an additional vehicle is needed for the component, and we can enforce this with an inequality of type (4c). We introduce these cuts for three different cases: fractional components, integral components (and thus circuits violating the length bound), and all subpaths resulting from integral components that violate the length bound. As the solutions arise in a branch-and-cut setting, some of the branching decisions might not be optimal. Therefore, these cuts have local validity only in the branching tree. To overcome this problem, we introduce globally valid cuts as follows. We consider the graph induced by the nodes of an integral component Q_i and compute lower bounds for the distance of the tour needed to serve these nodes. If the lower bound exceeds the distance which can be served by the number of vehicles serving the component, we can introduce a valid cut. We use two easy methods to compute lower bounds for the tours: the relaxation of TSP to 1-trees, and its classical relaxation to the assignment problem, see [32].

The transformation Ψ, and the extension of the existing software to handle distance constraints allowed us to effectively handle the TRP problem. The results are presented in the experimental Section 4.

Solving the TSSP Problem The TSSP has an interesting connection to *Minimum Cut Linear Arrangement Problems*. To see the connection we consider an easier variant of TSSP in which we drop Condition 3, set $T^h_{\text{shunt}} = 0$, and only ask for the sequence of in- and outbound trains at the hub that minimizes the necessary hub capacity. The resulting sequencing task can be depicted by the bipartite *in-out graph* $G_{\text{io}} = (U \cup V, E)$ in Figure 3. The inbound trains (in R^x) correspond to nodes of the top partition, the outbound trains (in R^y) to nodes of the bottom partition. Each edge $e = (r^x_i, r^y_j)$ has a *volume* $\text{vol}(e)$ corresponding to the number of cars that train r^y_j receives from train r^x_i. We call G_{io} a *uniformly directed bipartite graph*, because all edges are directed from U to V.

The problem is equivalent to finding a *linear arrangement* of the graph G, i.e., an embedding of the graph onto the horizontal line, such that all edges are directed from left to right. For such an arrangement, the maximal number of edges crossing any vertical line is the *(cut-) width*, and it corresponds to the maximal number of cars residing in the shunting-yard. The width of a graph G is given by the minimal width of a linear arrangement of G.

Without the directions and the restriction to bipartite graphs, this problem is known as the minimum cut linear arrangement, a well studied NP-complete problem [21, problem GT44] that was shown to remain NP-hard for graphs of degree 3 [34], and even planar graphs of degree 3 [35]. In [22] the authors extend these results in the following way:

Theorem 1. *It is NP-hard to decide if a uniformly directed bipartite planar graph of out-degree 3 and in-degree 2 admits a linear arrangement of width ℓ.*

In spite of this result we can solve the instances of the TSSP problem that arise in our setting by a simple ILP formulation, because these are not too large. For this formulation, we discretize the time horizon into τ points in time $T = \{0, \ldots, \tau - 1\}$ and make use of the previously defined in-out graph G_{io}. We introduce binary variables a^t_r and $d^t_{r'}$ that model arrival (and departure) of the trains $r \in R^x$ ($r' \in R^y$, respectively) at times $t \in T$. Here, we assume that the shunting time T^H_{shunt} is given in time slots. We refer to E as the edge set of the in-out graph G_{io}.

$$
\begin{array}{llll}
\min & C & & \\
\text{s.t.} & a_r^t \le a_r^{t+1} & \forall r \in R^x, t \in T & \text{(6a)}\\
& d_r^t \le d_r^{t+1} & \forall r \in R^y, t \in T & \text{(6b)}\\
& a_r^t \ge d_{r'}^{t+T_{\text{shunt}}^H} & \forall t \in \{0,\dots,\tau - T_{\text{shunt}}^H - 1\}, & \\
& & \forall (r,r') \in E & \text{(6c)}\\
& d_{r'}^t = 0 & \forall r' \in R^y, & \\
& & \forall t \in \{0,\dots,T_{\text{shunt}}^H - 1\} & \text{(6d)}\\
& \sum_{\substack{e \in E\\ e=(r,r')}} \text{vol}(e)(a_r^t - d_{r'}^t) \le C & \forall t \in T & \text{(6e)}\\
& a_r^{t_i} = 0 & \forall r \in R^x : \text{arrive}_H(r) > t_i & \text{(6f)}\\
& d_{r'}^{t_i} = 1 & \forall r' \in R^y : \text{dep}(r') < t_i & \text{(6g)}\\
& \sum_{r \in R^x} a_r^t \ge \sum_{r \in R^y} d_r^{t+T_{\text{shunt}}^H} & \forall t \in \{0,\dots,\tau - T_{\text{shunt}}^H - 1\}, & \text{(6h)}\\
& a_r^0 = 0, \quad a_r^{\tau-1} = 1 & \forall r \in R^x & \\
& d_{r'}^0 = 0, \quad d_{r'}^{\tau-1} = 1 & \forall r' \in R^y & \text{(6i)}\\
& \text{all } a_{\cdot}, d_{\cdot} \in \{0,1\} & & \text{(6j)}
\end{array}
$$

Equations (6a), (6b) and (6i) impose that, for every edge e, the variables $a_e^{\cdot}$ and $d_e^{\cdot}$ form a monotone sequence starting with 0 and ending with 1.

The arrivals and departures of trains are scheduled at the 0-1 transition of the respective variables, i.e, for a pickup route $r^x \in R^x$ we set $\text{arrive}_H(r^x) = t'$ if $a_{r^x}^{t'+1} - a_{r^x}^{t'} = 1$, and symmetrically for a delivery routes. Constraints (6c) and (6d) enforce that an outbound train can only depart if all its cars have arrived and that T_{shunt}^H time units are available for shunting those cars. Constraints (6e) represent the capacity constraint over all time slots, which is the objective value. Constraints (6f) and (6g) introduce time constraints for the earliest arrival/latest departure of trains, i.e., from the time windows we infer a constraint of type $\text{arrive}_H(r) > t'$ on the arrival (departure) times at the hub and express this in the form of Constraints (6f) and (6g). Constraints (6h) enforce that a train can only depart from the hub if there is an engine available.

Our experiments show that for the TSSP instances that arise from the solutions to the TRPs on the SBB Cargo Express instance, we can calculate a schedule that minimizes the maximum hub load and respects the time windows in at most a few minutes.

3.2 Model 2: A Column Generation Based Approach

Since Model 2 involves an exponential number of variables, we adopt column generation techniques [19]: we start with a *restricted* problem, containing only the direct shipment variables d_s, then solve its linear relaxation (the so called restricted master problem, RMP), obtaining a vector of (optimal) dual variables. We use this dual information to identify new variables of negative reduced cost. If any such variables are found, they are included in the RMP and the whole process is iterated. Otherwise, the value of the linear relaxation of the RMP is a valid lower bound. Note that including all direct shipment variables has the advantage of making the RMP feasible from the beginning. In contrast to the decomposition based Model 1, the linear relaxation of Model 2 delivers a lower bound on the value of an optimal solution. Furthermore, good integer solutions can be found by combining a subset of the dynamically generated columns which satisfy the constraints of Model 2.

Since an optimal solution in which each shipment is served by one route always exists, we can safely relax the set partitioning constraints to set covering constraints. Then, the corresponding dual variables are restricted in sign, which leads to an easier cost structure in the generation of columns. Furthermore, it is also easier to obtain feasible RMPs.

The Pricing Problems The problem of finding columns of (most) negative reduced costs is called the pricing problem. For the sake of clarity, we denote the dual variable corresponding to each constraint by adding indices to the Greek letter indicating the constraint itself. Since the pricing problems involving pickup and delivery routes are completely symmetric, we discuss the pickup pricing only.

Here, the reduced cost of a (column encoding a) pickup route r to hub h' scheduled at time t' is

$$C_{\text{engine}} + \sum_{e \in r} l(e) \cdot \bar{c} - \sum_{s \in r} (\hat{\pi}_s + \phi_{sh'}) - \sum_{s \in r, t \leq t'} \hat{\sigma}_{h'st} - \text{vol}(r) \sum_{t \geq t'} \chi_{th'} - \sum_{t \geq t'} \beta_{h't} \tag{7}$$

where $\hat{\pi}_s, \beta_{ht} \geq 0$, $\chi_{th}, \hat{\sigma}_{hst} \leq 0$ and $\phi_{sh} \in \mathbb{R}$ represent the dual variables associated to the constraints of Model 2. This reduced cost has three components: a cost $\sum_{e \in r} l(e) \cdot \bar{c}$, which depends only on the arcs used in the route, a prize

$$\sum_{s \in r} \text{rc}(s) := \sum_{s \in r} \left(\hat{\pi}_s + \phi_{sh'} + \text{vol}(s) \cdot \sum_{t \geq t'} \chi_{th'} + \sum_{t \leq t'} \hat{\sigma}_{h'st} \right) ,$$

which depends only on the shipments picked up, and a constant contribution $C_{\text{engine}} - \sum_{t \geq t'} \beta_{h't}$, which depends only on the destination hub and arrival time slot.

Definition 5 (Pickup Pricing Problem). *Given a network N, a hub $h' \in H$, an arrival time t', parameters $L_{\max}$, T_{couple}, $\bar{v}$, $\bar{c}$, a set of shipments S, and a* prize *rc(s) for each shipment $s \in S$, find a time consistent, volume admissible, scheduled route in N of minimum reduced cost.*

This problem is a variation of the well-known NP-hard resource constrained shortest path problem with time windows (RCSPP) [20]. We devised a particular dynamic programming shortest path algorithm, similar to those presented in [7, 20] for solving the pickup pricing problem. Each label l represents a partial route, and encodes a 5-valued *state* $(\mu(l), \Pi(l), \tau(l), \nu(l), \sigma(l))$, where $\mu(l) \geq 0$ is the cost of the partial route, $\Pi(l) \geq 0$ the collected prize, $\tau(l) \geq 0$ the elapsed time, $\nu(l) \geq 0$ the used volume and $\sigma(l)$ the set of visited nodes in which pickup operations occur. Each label refers to a particular node of the network. As it is not surprising for a railroad problem, our routes cannot be always *elementary*, that is, without node and edge repetitions. However, we only construct routes which we call "shipment elementary," which means that no shipment is picked up twice. Forcing the routes to be shipment elementary is enought to preserve the quality of the the lower bound we obtain from the RMP.

Initialization. Let $S_1(v)$ and $S_2(v)$ be two subsets of pickup shipments at node v; then $S_1(v)$ *dominates* $S_2(v)$ if

$$\sum_{s \in S_1(v)} \text{rc}(s) \geq \sum_{s \in S_2(v)} \text{rc}(s) \;\wedge\; \sum_{s \in S_1(v)} \text{vol}(s) \leq \sum_{s \in S_2(v)} \text{vol}(s)$$

and all the shipments in $S_1(v)$ have no later pickup time with respect to each shipment in $S_2(v)$. This is not a restriction in our case, since all the shipments in the same station share the same earliest pickup time. That is, picking up all shipments in $S_1(v)$ instead of all shipments in $S_2(v)$ requires no more resource consumption and gives no smaller prize. We begin by enumerating all *non-dominated* subsets of shipments for each pickup station in the network. Furthermore, we create a set L of $|N|$ labels corresponding to the initial state $(0, 0, 0, 0, \emptyset)$, one for each node of the network.

Extension procedure. Let L be the set of labels created so far, and $l^* \in \text{argmin}_{l \in L}\{\mu(l) - \Pi(l)\}$. In each round the label l^* is *pushed* to every neighboring node. Let l^* refer to node i^*, let j be one of its adjacent nodes and let e be the arc connecting them: a set of labels is created and added to L. This set contains a label for each non-empty non-dominated subset of shipments $S_k(j)$ at node j, encoding a state

$$\left(\mu(l) + l(e) \cdot \bar{c},\;\; \Pi(l) + \sum_{s \in S_k(j)} \text{rc}(s),\;\; \tau(l) + l(e) \cdot \bar{v} + T_{\text{shunt}},\;\; \nu(l) + \sum_{s \in S_k(j)} \text{vol}(s),\;\; \sigma(l) \cup \{j\}\right)$$

and one additional label encoding a state $(\mu(l) + l(e) \cdot \bar{c}, \Pi(l), \tau(l) + l(e) \cdot \bar{v}, \nu(l), \sigma(l))$ corresponding to no pickup operation. Each label with $\nu(l) > L_{\text{max}}$ or $\tau(l) > t'$ is discarded. Label l^* is finally removed from L and stored in a separate list $\bar{L}$.

Label Pruning. As in [7], we delete any label l at node i with $\tau(l) + \bar{t}_{ih'} > t'$. Furthermore, let $\underline{l}$ be the value of the incumbent RCSPP solution. Following [33], during the creation of each label l we compute an upper bound $\bar{\Pi}$ on the best prize that can still be collected by filling the remaining volume; this requires solving a fractional knapsack problem [37]. If the value $\mu(l) - \Pi(l) - \bar{\Pi}$ is still higher than $\underline{l}$, label l can be discarded, since it cannot yield improvements on the incumbent solution.

Dominance rule. In our algorithm, a label l_1 *dominates* a label l_2, if they refer to the same node of the network, and $\mu(l_1) \leq \mu(l_2)$ and $\mu(l_1) - \Pi(l_1) \leq \mu(l_s) - \Pi(l_2)$ and $\tau(l_1) \leq \tau(l_2)$, and $\nu(l_1) \leq \nu(l_2)$. Moreover, in an optimal MHSOP solution no route performs pickup operations more than once at the same station, although a particular setting of the prizes $\text{rc}(s)$ may yield the generation of routes containing *cycles*. As mentioned above, forbidding such cycles (i.e., restricting the search to *elementary* routes) leads to substantially better lower bounds. Unfortunately, this comes at the price of making the RCSPP computation much harder; in fact, we must enforce $\sigma(l_1) \subseteq \sigma(l_2)$ as a further condition for label l_1 to dominate label l_2. We stress that our technique requires elementariness only on a small subset of the nodes of the network: this makes the problem tractable from a computational point of view. Following [5], the set $\sigma(l)$ of each label is represented as a vector of binary resources, one for each pickup station of the network; each of them is set as consumed as soon as pickup operations are performed at the corresponding station. Finally, as done in [27], we tighten the dominance rule by including in the set $\sigma(l)$ all stations that cannot be reached anymore due to resource limitations.

Termination. Let $\bar{L}(h')$ be the subset of labels in $\bar{L}$ refereing to node h'. When no new label is created, a label $l^* \in \text{argmin}_{l \in \bar{L}(h')}\{\mu(l) - \Pi(l)\}$ encodes an optimal solution to the pricing problem.

Dominance of time slots. We also check the following simple dominance rule for entire time slots. Let hub h' be fixed. If $t_2 > t_1$ and for all $s \in S$ we have that $\sum_{t_1 \leq t < t_2} \beta_{h't} = 0$ and $\sum_{t \leq t_2} \hat{\sigma}_{h'st} \geq \sum_{t \leq t_1} \hat{\sigma}_{h'st}$, that is $\sum_{t_1 < t \leq t_2} \hat{\sigma}_{h'st} = 0$, an optimal solution for t_2 cannot be worse than an optimal solution for t_1. Therefore, we may discard time-slot t_1 in the search for the most negative reduced cost column.

Pricing of the hub-connection routes. Since no pickup or delivery operations occur in hub-connections, these routes always follow the shortest path between the hubs. The pricing problem simplifies to a knapsack problem, which we solve using the MINKNAP algorithm [37], adapted to handle fractional prizes as described in [10]. Furthermore, the time-slot reduction procedures described above can be readily adapted to hub-connection pricing.

Finally, we remark that since the direct shipment variables are included in the initial RMP we do not need to consider them during pricing.

Acceleration Techniques The literature on column generation abounds with acceleration techniques [18]. We briefly discuss some of our methods.

Pre-Processing Strategies. We condensed the original SBB Cargo Express network to a smaller network without loosing optimality guarantees. The details of such reductions are discussed in Section 4.

Heuristic Pricing. As most of the time is spent in pricing we devised heuristic speedup techniques. At each pricing step, we stop the pricing algorithm as soon as a fixed number of negative reduced cost columns has been found. Moreover, we limit the number of generated labels, as commonly done in routing problems [45]. We reduce the number of demands by aggregating shipments during pickup and delivery pricing. The heuristics are discarded if no negative cost label is found.

Perturbation. For the calculation of each RMP we perturb the right hand side of the σ-constraints by small random values. Consistent with the literature [17] this yields a significant speed-up of the LP-solving steps.

Columns Management. In order to keep the RMP small, we subject the columns to aging. If a column keeps being nonbasic for a given number of pricing iterations it is removed from the RMP and added to a column pool. Before pricing, we scan this pool: if any previously generated column is found with a negative reduced cost, we insert it in the RMP and we skip pricing.

Stabilization. A common problem in column generation is that the dual variables tend to oscillate and to assume extreme values. Using an interior point method to solve the RMP is a possible remedy, but we encountered severe numerical problems when using the barrier algorithm in CPLEX. Instead, we adapted the interior point stabilization approach described in [41] to our problem, obtaining better convergence. However, balanced dual values required to solve harder pricing problems and this technique did not pay off with respect to the overall performance.

Primal Heuristics As solving only the linear relaxation of Model 2 to optimality takes a long time, we experimented with heuristics based on the *dive-and-fix* paradigm [46]. We augmented the standard rounding procedure by performing column generation steps and including problem specific rules, obtaining a *dive-and-generate* heuristic. Given a fractional solution for the RMP, the heuristic iterates the following steps until an integral solution has been found.

1. Round up the column c with the highest fractional value, given that it is consistent with the columns rounded up so far in the RMP constraints.
2. Round down all columns in the RMP that are not time-consistent with c.
3. Solve the remaining RMP using the dual simplex algorithm.
4. If the solution is integral stop.
5. Otherwise, remove the shipments included in c from the pickup, delivery or hub-connection pricing problem, according to the type of the route encoded by c.
6. Perform a fixed number of pricing iterations to include new columns into the RMP, and go back to Step 1.

Note that by rounding up and down fractional valued columns, it can happen that the RMP may become infeasible. If this is the case, we apply a technique called Farkas Pricing, which tries to restore the feasibility of the RMP by considering optimal dual rays of the (unbounded) dual problem. We detail this technique in the Appendix A.1. To the best of our knowledge, such technique has not been applied in column generation so far.

Since the number of pricing steps is limited, dive-and-generate may still fail in finding a feasible solution. Therefore, after each rounding step we execute a fast local improvement procedure which tries to complete the current partial solution by further rounding. To this aim, the procedure first greedily rounds up more columns, maintaining the time consistency between arrival and departure times of the shipments at the hubs. Uncovered shipments are transported by direct paths. Inconsistencies of shipments arriving at one hub and leaving from another hub, as well as inconsistencies on the engines serving each hub are solved by using hub connecting routes. Hub capacity problems are addressed by shifting the routes in time as far as possible. If capacity problems remain, the heuristic introduces direct paths to lower the necessary hub capacity.

We remark that our dive-and-generate approach can be applied without problem-specific knowledge to any ILP solved by column generation techniques. Such heuristic is unconventional if compared to other heuristics in column generation settings that rely more on the compact formulation or metaheuristics that are initialized and guided by the column generation process, see [16]. Indeed, we try to exploit first the skill of the pricing algorithm in generating high quality routes, second the involved structure of the RMP to combine the routes respecting time consistency and hub-capacity, third the problem-specific structure for local improvement.

4 Experiments

In this section we report on the experimental results with the three models. All computations were carried out on a standard Linux PC with a 3 GHz processor and 4GB memory. CPLEX 9 was used as ILP solver for Model 0 and as LP solver for Model 1 and Model 2.

We start by describing the SBB Cargo Express instance.

The planners of SBB Cargo Express service provided us with real data, i.e., the actual railroad network, their timetable and the demand matrix of a specific month. The railroad network for the SBB Cargo Express service has 651 nodes and 1488 edges, and they currently operate with two hubs, located in Däniken and Zürich Müllingen. In a preprocessing phase, we first calculate the all-pairs shortest paths among the nodes with shipments and the hubs. Edges that do not occur on any such shortest path can be safely ignored. In the resulting graph we contract degree-two nodes if they are neither a hub nor an origin nor a destination of any shipment. The preprocessing condensed the network to 121 nodes and 332 edges.

In Figures 4 and 5 we show respectively the original network and a detail of it, together with the condensed network that we extracted. Dark nodes represent stations with shipments in the SBB Cargo Express service, and stations without shipments that were retained in the condensed network. The two bigger square nodes correspond to hubs. The light colored nodes and edges are stations and connections that are not retained in the condensed network.

Since the given demand matrix only comprises a month total, we divided the supply in a daily average, and rounded fractional numbers. This resulted in a 200 shipments instance. Since the actual time windows were not available, we defined the earliest pickup and latest delivery times by relaxing the pickup and delivery times on the currently implemented plan by one hour.

In order to simulate the real setting, we fixed the parameters to the following realistic values: The maximum train load is 25 cars, the average train speed is set to $\bar{v} = 60km/h$. As a coupling time at stations we chose $T^{v}_{\text{couple}} = 27min$. The length of each time slot is set to $15min$. The shunting time at the hubs is set to $T^{h}_{\text{shunt}} = 27min$, equal at both hubs. We considered an hub to be overloaded if more than $\text{cap}_h = 80$ cars are in the yard during the same time slot. Finally, we set $C_{\text{engine}} = 1000$ and $\bar{c} = 1$. The three orders of magnitude difference in cost makes the minimization of distance a secondary objective over minimization of engines.

Taking into account only the number of shipments, the SBB instance is about the size of the largest optimally solved (much simpler) VRP instances; yet, trying to solve the entire tactical planning process turns it into a very challenging problem.

4.1 Model 0

We implemented Model 0 using the OPL Modeling language [26], and solved it using CPLEX. On a toy instance with 11 nodes, 23 edges and 11 shipments we did not get any feasible solution in 7 hours. However a feasible (not optimal) solution for the first 5 and the last 6 shipments separately can be found in about 20 minutes. In fact, this suggests a decomposition approach, and it was a motivation to develop and implement more involved approaches like Models 1 and 2.

4.2 Model 1

We implemented the branch-and-cut approach for TRP using SYMPHONY 5, a framework by Ralphs et al. [39]. We used CPLEX as LP solver for SYMPHONY, and LEDA 4.5 for computing the minimum spanning trees and the assignment problems needed for the cuts.

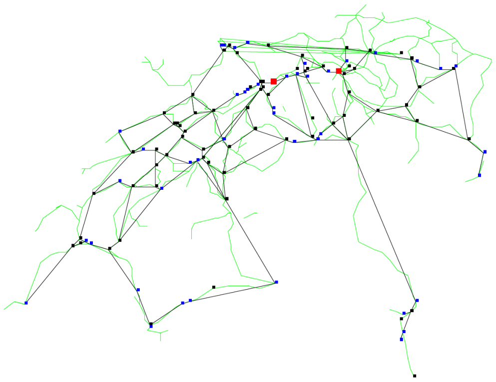

Fig. 4. The original railroad network together with the condensed network that we extracted.

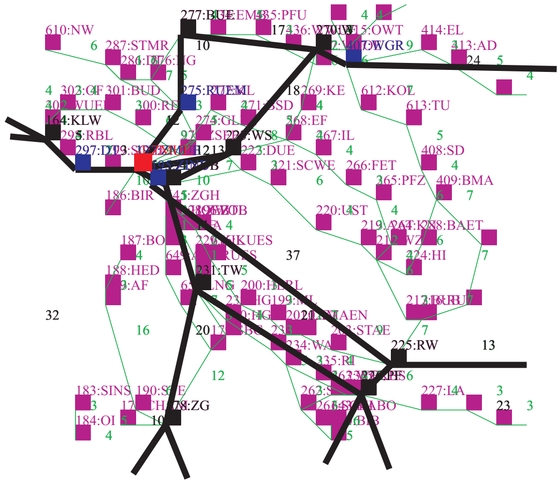

Fig. 5. An enlarged detail of the complete network of Figure 4.

The computation flow through the three steps of Model 1 is summarized in Table 1. The table is built by 5 blocks, one for each step in the solution procedure, as indicated in the heading row.

In the first block we report the number of cars composing shipments managed by the same hub after the partitioning step. For the partitioning we enforced a balance constraint of 55%, i.e., at most 55% of all shipments can be assigned to one hub. The resulting split led to four instances for TRP, pickup and delivery for Däniken and Zürich Müllingen.

In the second block we indicate, for each hub and for both the pickup and delivery phases, the number of nodes (N) shipments (S) and engines (E) in the corresponding DCVRP instance. The $D_{\max}^{\text{DCVRP}}$ values in the DCVRP subproblems (column '$D_{\max}$') were chosen by taking the maximum distance from each hub to a shipment assigned to it, and increasing it by the equivalent of time needed for shunting twice. The rationale behind this choice is to allow serving two shipments with the engine serving the farthest one. The shipments for the instances were obtained by aggregating, in cooperation with the SBB Cargo planners, some of the demands with the same origin for the pickup problem and with the same destination for the delivery problem.

In the third block we report the results obtained by solving the TRPs in terms of overall distance traveled by the fleet (column 'dist.'), CPU time needed to obtain such solution (column 'comp. time'), gap between the best known upper and lower bounds if optimality was not proved (column 'gap'). The engine fleet size of each instance was incremented sequentially until the number of infeasibilities with respect to the time windows did not decrease anymore. The solution process for all instances took about 10 days in total. Since we observed dramatically higher computation times for instances with fewer engines (and more infeasibilities), the best approach is to decrement the fleet size from an initially large value and stop when the number of infeasibilities increases.

The few remaining infeasibilities were solved by a very simple fixing heuristic similar to the one presented in Section 3.2 for Model 2. In the fourth block of the table we indicate the overall traveled distance, the number of used engines and the overall cost on the whole solution. This approach resulted in using an engine fleet of 11 engines for the shipments assigned to Zürich Mülligen (ZMUE), and 24 for the shipments assigned to Däniken (DK), and a fleet of 35 engines in total.

The resulting TSSP instance takes the time windows and the routes determined by TRP as an input. As reported in the fifth block of the table, in the timetable which minimizes the maximum hub load, at most 80 cars are in ZMUE and 106 cars are in DK at the same time. This causes a slight overload in DK. The solution was found in less than a minute.

To sum up we were able to produce a solution that is close to the operational constraints. This comes at the cost of a large computation time; it is also difficult to estimate losses in objective value by the three-phase decomposition.

Table 1. Computation flow through all phases of Model 1.

Partition		TRP Instances					TRP results			totals after fixing			TSSP
hub	cars	mode	$D_{\max}$	N	S	E	dist.	comp. time	gap	dist.	E	cost	max load
DK	253	pickup	378*km*	26	25	21	1660*km*	<1min	0%	7686*km*	35	42686	106
		delivery	296*km*	31	30	21	2590*km*	90min	0%				
ZMUE	208	pickup	276*km*	15	13	9	936*km*	<1min	0%				80
		delivery	319*km*	17	14	9	1628*km*	15h	5.9%				

4.3 Model 2

We implemented our column generation algorithm for Model 2 using the SCIP library by Achterberg [1], with CPLEX as LP solver. It is the first time SCIP is used to implement a CG algorithm.

Results First, the full instance, on which we apply Model 2, is considerably larger than the aggregated instances for the TRP, as it considers 200 shipments and the preprocessed network. With our current implementation we reach the tailing-off phase in the root node of the branch-and-price tree after a calculation time of around four days. At this time the value of the relaxation is 29217.

The best integral solution found with the dive-and-fix heuristic of Section 3.2 has a cost of 35276. This solution was achieved with 8 intermediate pricing steps after each variable fix. Similar results can be found with more steps, but experimental results showed that it is ineffective to price more than 12 times. The best solution was found after 85 hours of computation. However, very similar quality results can be found within 12 hours. The best solution uses 27 engines, two of which are direct connections. 12 engines travel to and from Däniken, 13 to and from Zürich–Mülligen. Furthermore, there are 8 hub-connecting trains in each direction. The total traveled distance is of 8276*km*.

The local improvement is mainly useful when the dive-and-fix heuristic fails, or for obtaining good solutions in the early computation phases.

4.4 Comparison of results

The advantage of Model 2 over Model 1 is twofold. First, although the approach worked fine for the hub ZMUE, we could not avoid some overload of the slightly bigger hub DK.

In spite of the shorter overall distance, Model 1 thus requires substantially more engines than Model 2. For the SBB Cargo Express instance, this results in a loss of 7410 (or 20%) with respect to the best solution obtained by Model 2. Second, the need to iterate the TRP and TSSP solution process with an increasing number of engines and handling the infeasibilities leads to a enormous computation time. Thus, the integrated approach of Model 2 performs significantly better on the given large real-world instance.

5 Conclusion and Future Work

Our computational experience confirms that capturing all the modeling details of multi hub-and-spoke systems, while still providing reliable solution methods is a challenging task.

Yet, on the other hand, we also demonstrated that this task is by far not hopeless.

Model 1 shows both the appealing and limiting features of distributing tactical decisions to separate levels, as is commonly done in practice. In fact, the solution methods of Model 1 rely on consolidated and effective algorithmic techniques and robust existing software packages. The optimal solution at each decision level can be found with an affordable computational effort. However, a tough parameter tuning has to be carried out to make the algorithms of the earlier decision levels produce feasible instances for the later decision levels.

On the other hand, with Model 2 we are able to consider all interacting tactical decisions as a whole, and provide a solution algorithm that consistently produces feasible solutions of provably good quality.

A direct comparison between our solutions and the currently implemented plan is not possible at this stage, as our experiments were carried out on average valued instances; moreover, we decided to ignore some of the seemingly secondary aspects in our models: We do not consider the problem of engine driver assignment. We do not consider the problem of switchbacks and furcations. It is rather straight-forward however to incorporate the handling of switchbacks and furcations in the pricing, since these only cause a route to take longer to be carried out.

Our column generation algorithm exploits and extends very recent optimization techniques: One issue we did not discuss is the question of branching rules. A natural branching rule in our setting is the assignment of shipments to hubs, based on the fractional assignment of the solution at hand, which we implemented. However, we only did few experiments on small instances with different branching rules, and focused on heuristically obtaining integral solutions already in the root node for the whole instance.

A near-optimal solution of a practical problem of this scale and complexity was entirely out of scope some years ago. Even though we focused on a single instance of a particular problem, we think that research like ours will contribute to algorithmic and modeling improvements, leading to state-of-the-art solvers which are capable of solving a problem like ours “out of the box.”

Acknowledgements

We thank Kathrin Schulte from SBB Cargo Basel for her great support in providing us with data and answering many questions about freight rail systems. Further, we thank Tobias Achterberg for his support and crash course for SCIP and Ted Ralphs for his support with SYMPHONY.

References

1. T. Achterberg. *Constraint Integer Programming*. PhD thesis, Technische Universität Berlin, 2007. `http://opus.kobv.de/tuberlin/volltexte/2007/1611/`.
2. R. K. Ahuja, K. C. Jha, and J. Liu. Solving real-life railroad blocking problems. to appear in Interfaces, 2007.
3. A. A. Assad. Models for rail transportation. *Transportation Research. Part A: General*, 14A:205–220, 1980.
4. G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation*. Springer, 1991.
5. J. Beasley and N. Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19:379–394, 1989.
6. U. Blasum and W. Hochstättler. Application of the branch and cut method to the vehicle routing problem. Technical Report zpr2000-386, Zentrum für angewandte Informatik, Köln, 2000.
7. N. Boland, J. Detheridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Operations Research Letters*, 34(1):58–68, 2006.
8. M. Campetella, G. Lulli, U. Pietropaoli, and N. Ricciardi. Freight service design for the italian railways company. In R. Jacob and M. Müller-Hannemann, editors, *ATMOS 2006 - 6th Workshop on Algorithmic Methods and Models for Optimization of Railways*. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2006. <http://drops.dagstuhl.de/opus/volltexte/2006/685> [date of citation: 2006-01-01].
9. A. Ceselli, M. Gatto, M. E. Luebbecke, M. Nunkesser, and H. Schilling. Implementation of the edge-based classical vehicle routing model and of the TSSP problem for SBB Cargo Express in OPL. Available as OPL-files at `http://www.inf.ethz.ch/personal/mnunkess/SBB-Cargo/`.
10. A. Ceselli and G. Righini. A branch and price algorithm for the capacitated p-median problem. *Networks*, 45(3):125–142, 2005.

11. N. Christofides, A. Mingozzi, and P. Toth. Exact algorithms for vehicle routing. *Mathematical Programming*, 20:255–282, 1981.
12. G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:568–581, 1964.
13. J.-F. Cordeau, P. Toth, and D. Vigo. A survey of optimization models for train routing and scheduling. *Transportation Science*, 32(4):380–404, November 1998.
14. T. Crainic, J.-A. Ferland, and J.-M. Rousseau. A tactical planning model for rail freight transportation. *Transportation Science*, 18(2):165–184, May 1984.
15. T. Crainic and G. Laporte. Planning models for freight transportation. *European Journal of Operational Research*, 97:409–438, 1997.
16. E. Danna and C. L. Pape. *Column Generation*, chapter Branch-and-Price Heuristics: A case study on the vehicle routing problem with time windows, pages 99–129. Springer-Verlag, 2005.
17. G. Desaulniers, J. Desrosiers, A. Lasry, and M. M. Solomon. *Computer-Aided Transit Scheduling*, chapter Crew Pairing for a Regional Carrier, pages 19–41. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, 1999.
18. G. Desaulniers, J. Desrosiers, and M. M. Solomon. *Essays and Surveys in Metaheuristics*, chapter Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems, pages 309–324. Kluwer, 2001.
19. G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors. *Column Generation*. Springer-Verlag, 2005.
20. D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.
21. M. R. Garey and D. S. Johnson. *Computers and Intractability*. Freeman, 1979.
22. M. Gatto, R. Jacob, and M. Nunkesser. Optimization of a railway hub-and-spoke system: Routing and shunting. Technical Report 477, ETH Zurich, 2005.
23. M. F. Gorman. An application of genetic and tabu searches to the freight railroad operating plan problem. *Annals of Operation Research*, 78:51–69, 1998.
24. A. Hall, S. Hippler, and M. Skutella. Multicommodity flows over time: Efficient algorithms and complexity. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP'03)*, LNCS 2719, pages 397–409. Springer-Verlag, June 2003.
25. R. Hall. On the road to recovery. *OR/MS Today*, June 2004. available at `www.lionhrtpub.com/orms/orms-6-04/frsurvey.html`.
26. P. V. Hentenryck. *The OPL optimization programming language*. MIT Press, 1999.
27. S. Irnich and G. Desaulniers. *Column Generation*, chapter Shortest Path Problems with Resource Constraints, pages 33–65. Springer-Verlag, 2005.
28. M. H. Keaton. Designing railroad operating plans: A dual adjustment mehod for implementing lagrangian relaxation. *Transportation Science*, 26(4):263–279, November 1992.
29. O. K. Kwon, C. D. Marthland, and J. M. Sussman. Routing and scheduling temporal and heterogeneous freight car traffic on rail networks. *Transportation Research. Part E: Logistics and Transportation review*, 34(2):101–115, 1998.
30. G. Laporte, M. Desrochers, and Y. Nobert. Two exact algorithms for the distance-constrained vehicle routing problem. *Networks*, 14:161–172, 1984.
31. G. Laporte, Y. Nobert, and M. Desrochers. Optimal routing under capacity and distance restrictions. *Operations Research*, 33:1050–1073, 1985.
32. E. E. Lawler, J. K. Lenstra, A. H. G. R. Kan, and D. B. Shmoys. *The Travelling Salesman Problem*. Wiley, 1985.
33. M. Lübbecke. Dual variable based fathoming in dynamic programs for column generation. *European Journal of Operational Reseach*, 162(1):122 – 125, 2005.
34. F. S. Makedon, C. H. Papadimitriou, and I. H. Sudborough. Topological bandwidth. *SIAM Journal on Algebraic and Discrete Methods*, 6(3):418–444, July 1985.
35. B. Monien and I. H. Sudborough. Min cut is NP-complete for edge weighted trees. *Theoretical Computer Science*, 58(1-3):209–229, 1988.
36. D. Naddef and G. Rinaldi. *The Vehicle Routing Problem*, chapter Branch-and-Cut Algorithms for the Capacitated VRP. SIAM Monographs on Discrete Mathematics and Applications. SIAM, 2002.
37. D. Pisinger. A minimal algorithm for the 0-1 knapsack problem. *Operations Research*, 45:758–767, 1997.
38. T. R. Ralphs. Branch and cut references. `http://branchandcut.org/reference.htm#implementations`.
39. T. R. Ralphs. Symphony 5.0. `www.branchandcut.org/SYMPHONY`.
40. T. R. Ralphs, L. Kopman, W. R. Pulleyblank, and L. E. Trotter. On the capacitated vehicle routing problem. *Mathematical Programming*, 94(2–3):343–359, 2003.

41. L. M. Rousseau, M. Gendreau, and D. Feillet. Interior point stabilization for column generation. *Operations Research Letters*, 35:660–668, 2007.
42. SBB. Cargo express. `www.sbbcargo.com/en/index/ang_produkte/ang_produkte_express.htm`.
43. P. Toth and D. Vigo. *The Vehicle Routing Problem*, chapter An overview of vehicle routing problems. SIAM Monographs on Discrete Mathematics and Applications. SIAM, 2002.
44. P. Toth and D. Vigo. *The Vehicle Routing Problem*, chapter Branch-and-Bound Algorithms for the Capacitated VRP. SIAM Monographs on Discrete Mathematics and Applications. SIAM, 2002.
45. P. Toth and D. Vigo. *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications. SIAM, 2002.
46. L. A. Wolsey. *Integer Programming*. Discrete Mathematics and Optimization. Wiley, 1998.

A Appendix

A.1 Model 2 - Farkas Pricing

Consider a restricted linear programming master problem in general form that is infeasible.

$$\begin{aligned} \min cx \\ b \leq Ax \leq d \\ e \leq x \leq f \end{aligned} \tag{RMP}$$

As in the proof of the Farkas' Lemma we can set the objective function to 0 and consider the dual linear program

$$\begin{aligned} \max u_b b - u_d d + r_e e - r_f f \\ u_b A - u_d A + r_e - r_f = 0 \\ u_b, u_d, r_e, r_f \geq 0 \ . \end{aligned}$$

This linear program is feasible, which is certified by $(u_b, u_d, r_e, r_f) = 0$ and must therefore be unbounded, as the primal problem is infeasible. From the complementary slackness conditions it is clear that out of the two bounds associated with each constraint and each variable only one can be nonzero if these are different. If they are the same, still only one needs to be nonzero. Therefore, we can set $u = u_b - u_d$ and $r = r_e - r_f$. Then we have the following set of (in)equalities that certifies the primal infeasibility.

$$\begin{aligned} u_b b - u_d d + r_e e - r_f f > 0 \\ uA + r = 0 \ . \end{aligned}$$

As we have only a restricted linear programming master problem this does not imply that the underlying problem is infeasible. The aim of Farkas pricing is to add further variables such that the resulting RMP is feasible again. An addition of a variable corresponds to the addition of a further column to A. In our case we have all the nonbasic variables at their lower bound $e = 0 < f$. Therefore, it follows from complementary slackness that $r_f = 0$ and $r_e \geq 0$, so that $r \geq 0$. Suppose we find a new variable corresponding to a column a_i such that $-ua_i < 0$. Then for the infeasibility certificate above to carry on we need that $r_i = -ua_i < 0$ which is a contradiction to $r_i \geq 0$ and thus destroys this infeasibility certificate. This does not imply that the new RMP is feasible. Still, it is clear from the finiteness of the number of variables that this procedure must find a primal feasible solution in a finite number of steps if the primal is indeed feasible. To get a column a_i with $-ua_i < 0$ we call the same pricing algorithm as before except that we set the objective function to 0. The resulting reduced costs $0 - ua_i$ are exactly what is needed here. The SCIP library is designed in such a way that it automatically switches to Farkas pricing if a RMP becomes infeasible.