Les Cahiers du GERAD ISSN: 0711–2440

Multiple depot vehicle scheduling with controlled trip shifting

L. Desfontaines,
G. Desaulniers

G–2017–101

November 2017

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2017
– Bibliothèque et Archives Canada, 2017

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2017
– Library and Archives Canada, 2017

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Multiple depot vehicle scheduling with controlled trip shifting

Lucie Desfontaines
Guy Deslauriers

GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal (Québec) Canada, H3C 3A7

lucie.desfontaines@giro.ca
guy.desaulniers@gerad.ca

November 2017
Les Cahiers du GERAD
G–2017–101

Abstract: The multiple depot vehicle scheduling problem (MDVSP) has been widely studied in the context of public transit systems. It consists of building vehicle schedules to accomplish a set of scheduled trips at minimum cost. This work considers a generalization of the MDVSP that allows slight modifications of the trip scheduled start times. By shifting some trips, one can indeed expect to cover all trips with fewer vehicles or less expensive deadheads (vehicle moves without passengers). However, reducing the operational costs in this way should not be too detrimental to the overall quality of the timetable and, therefore, the following criteria should be controlled: the number of shifted trips, the headways between the consecutive trips of a line, and the quality of some passenger connections. To solve this generalized problem, we develop a two-phase matheuristic: the first phase computes vehicle schedules with a column-generation heuristic; the second relies on a mixed integer program to find the best possible timetable considering the computed vehicle schedules. Penalties are introduced in the first phase to increase the chances of finding a better-quality timetable in the second phase. Computational tests on real-life datasets from a bus company show that the proposed matheuristic can solve the problem efficiently, yielding solutions with a significant reduction in the number of vehicles used compared to the solutions of the classical MDVSP and a limited alteration of the timetable.

Keywords: Vehicle scheduling, trip shifting, timetable quality control, 2-phase matheuristic, column generation

---

Acknowledgments: We are thankful to Charles Fleurent and Veronique Bouffard from GIRO Inc. for introducing this problem to us, providing the data, and discussing our progress throughout the project. We gratefully acknowledge the financial support provided by GIRO Inc. and the Natural Sciences and Engineering Research Council of Canada under the grant RDCPJ 4634633-14.

Declaration of interest: None for Guy Desaulniers. Lucie Desfontaines is now working at GIRO Inc. which partially financed this project while Lucie Desfontaines was a Master's student at Polytechnique Montréal.

1 Introduction

Scheduling is crucial for public transportation as it should guarantee at the same time a high-quality service to passengers and cost-effective running for operators. Decisions can be split in several optimization problems which are generally addressed sequentially: setting the stops and frequency of each line, designing a timetable, determining vehicle schedules and, finally, building driver duties and rosters (see Desaulniers and Hickman (2007) or Ibarra-Rojas et al. (2015)). This work focuses on the problem of building vehicle schedules and its interaction with the timetabling process for bus companies, but it could be adapted to other types of transit systems. When the vehicles of a bus company are spread in several depots, this scheduling problem is known as the Multiple Depot Vehicle Scheduling Problem (MDVSP), which has been proved to be NP-hard (Bertossi et al. (1987)). Given a timetable of bus trips where a trip is defined by a start location and time and an end location and time, the MDVSP consists of finding a set of bus schedules that covers every trip exactly once while satisfying vehicle availability at each depot and minimizing the operating costs. A bus schedule must start and end at the same depot and is composed of a sequence of trips. Typically, the operating costs include a fixed cost for each vehicle used, connection costs between consecutive trips, as well as costs for traveling between the depot and the first/last stop of each schedule.

In practice, it is often possible to reduce the operating costs by slightly changing the initial timetable (e.g., the timetable of one or several trips can be shifted forward or backward by up to a few minutes). Indeed, such trip shiftings can turn infeasible connections between two trips into feasible ones and reduce the total connection costs or the number of vehicles used. However, even if these shifts are small, this additional flexibility can lead to a final timetable of lesser quality. For instance, the prescribed headway between two consecutive trips along a same line can be altered, or the new schedule can make some purposely scheduled passenger connections infeasible or too long. Furthermore, given that the timetable was built carefully, the number of modified trip start times should be limited to facilitate the approval of the timetable changes.

In this paper, we address the MDVSP with trip shifting and limited alteration of the timetable quality for a bus company. This problem that we call the Multiple Depot Vehicle Scheduling Problem with Controlled Trip Shifting (MDVSP-CTS) was proposed to us by our industrial partner, the company GIRO Inc., which is currently the world-leading optimization software provider for public transit companies. For this problem, we propose a mathematical programming model and develop a matheuristic for solving it.

1.1 Literature review

We first give a quick overview of the existing solution algorithms for the MDVSP. We then review the algorithms proposed to solve the MDVSP with trip shifting, before discussing the case with some control of the timetable quality.

To solve the MDVSP, several exact mathematical programming methods (mostly integer programming ones) have been proposed, including those by Carpaneto et al. (1989), Ribeiro and Soumis (1994), Lbel (1998) and Kliewer et al. (2006b). Ribeiro and Soumis (1994) introduced the use of column generation for the MDVSP. Extensions of their algorithm can be found in Hadjar et al. (2006) and Groiez et al. (2013). Kliewer et al. (2006b) used a time-space network to model the problem which allowed to solve efficiently the problem using a commercial mixed-integer programming (MIP) solver. Because the MDVSP instances faced by bus companies often involve a very large number of trips (thousands of trips), several heuristics have also been developed. Pepin et al. (2009) compare five of them and conclude that truncated column generation yields the most interesting results. This heuristic has been further studied by Guedes and Borenstein (2015) and Guedes et al. (2016). Laurent and Hao (2009) also developed an efficient iterated local search algorithm which relies on *block* moves, where a block is a group of consecutive trips. The interested reader is referred to Bunte and Kliewer (2009) for a detailed review on vehicle scheduling problems with one or several depots.

Some studies address extensions of the classical MDVSP where timetable flexibility is considered, with the intention of reducing the operating costs. This can lead to significant cost savings for the bus company. When trip start times can be chosen in an interval around a target start time, this problem is called the MDVSP with time windows. It has been studied by Desaulniers et al. (1998) and Hadjar and Soumis (2009), both

using column generation. When the choice of start times is limited to a discrete set, the problem is called the MDVSP with Trip Shifting. Kliewer et al. (2006a) developed a multi-commodity network flow model for this problem that is based on a time-space network where each trip might be represented by multiple arcs if alternative start times create new possible connections between trips. Heuristic rules for eliminating further alternative trips are also devised to reduce the model size and speed up the solution by a commercial MIP solver.

Partial or complete integration of timetabling and vehicle scheduling was studied in relatively recent works. In most of them, a local search algorithm was developed either to solve the whole problem or part of it. For the single depot case which leads to a non NP-hard vehicle scheduling problem, Guihaire and Hao (2010) minimized a multi-objective function of the costs and the timetable quality (with respect to headways and passenger connections) with a local search algorithm, while Ibarra-Rojas et al. (2014) used two integer linear programs to find Pareto-optimal solutions considering the number of vehicles used and the quality of the passenger connections. A two-phase heuristic was also developed by Schmid and Ehmke (2015), who first finds good vehicle schedules and then determines the best feasible timetable (with respect to headways only) considering these schedules. For the multiple depot case, a large neighborhood search algorithm was proposed by Petersen et al. (2012) to solve a MDVSP with trip shifting aiming at improving the passenger connections. Van den Heuvel et al. (2008) implemented a simulated annealing heuristic to determine simultaneously a timetable (with even headways per line) and bus schedules when several types of vehicles can be chosen for each trip. Ceder (2011) proposed a heuristic way to simultaneously construct vehicle schedules and a timetable with even headways or even load per vehicle. Recently, Laporte et al. (2017) and Liu and Ceder (2017) additionally took into account passenger assignment. To handle such a complex problem, they considered restrictive assumptions on the vehicle scheduling part (no interlining, no deadheading) which turn the problem into a single-depot problem.

Finally, Kliewer et al. (2012) studied the simultaneous vehicle and crew scheduling problem with trip shifting, which involves three decision levels. They extended the work of Kliewer et al. (2006a) by devising a column generation algorithm where some constraints are relaxed in a Lagrangian way. They obtained promising results on medium-size instances.

1.2 Contributions and paper structure

As discussed above, only a few works have proposed algorithms to solve a MDVSP where the timetable can be modified with some controls on the quality of the resulting timetable (Petersen et al. (2012), Van den Heuvel et al. (2008), and Ceder (2011)). None of these papers have considered simultaneously the passenger connections and the headways as timetable quality criteria. Furthermore, because our industrial partner GIRO counts numerous customers (more than 200 companies around the world) which all have their own rules and constraints, one of our key intention is to develop an easily adaptable algorithm. This guided the following three choices. Firstly, we propose an algorithm based on column generation that allows to introduce, if needed, special constraints on bus schedules without changing the complete model. In addition, our model is based on a connection network, which can easily take into account specific preferences (such as penalties on interlining or long/short connections) as opposed to a time-space network. Finally, our algorithm is highly flexible on how timetable quality is controlled.

Our contributions are threefold. First, we formulate the MDVSP-CTS as a mixed linear integer program with an objective function that includes weighted terms for minimizing the operational costs and maximizing the timetable quality. Second and foremost, we introduce a two-phase matheuristic for solving the MDVSP-CTS. In the first phase, a column generation heuristic is used to build vehicle schedules. In the second phase, a mixed integer program is solved to determine the best timetable given the computed vehicle schedules. Penalties on specific connection arcs are introduced in the first phase to increase the chances of finding a better-quality timetable in the second phase. Finally, through computational experiments on real-life instances and instances derived from real-life ones, we show the effectiveness of the proposed algorithm to yield various approximate Pareto-optimal solutions.

This paper is structured as follows. In the next section, we define the MDVSP-CTS and formulate it as a mixed integer linear program. In Section 3, we describe the two-phase matheuristic. Our computa-

tional results are then reported in Section 4. Finally, Section 5 presents concluding remarks and future research directions.

2 Mathematical model

The proposed model for the MDVSP-CTS is presented in two parts: we describe first a model for the MDVSP with trip shifting and then its extension with controlled trip shifting.

2.1 MDVSP with trip shifting

Let $\mathcal{V} = \{v_1, v_2, \ldots, v_m\}$ be a set of m trips. Each trip $v \in \mathcal{V}$ is initially scheduled to start at time t_v. However, its start time can be shifted forward or backward and must be chosen from a discrete set of possible start times. For each possible start time t_v^i of v, we create a *copy* k_v^i of v and denote by K_v the set of copies of trip v (including the initial one).

The bus company possesses vehicles spread in a set of depots $\mathcal{D}$. The number of vehicles available at depot $d \in \mathcal{D}$ is denoted b_d. We assume that all the vehicles in a depot are identical. If this is not the case, the depot can be split into several depots that are located at the same place. There might be compatibility restrictions between the vehicles in a depot and the trips: for example, some trips may not be covered by an articulated bus. Let $\mathcal{V}^d$ be the set of trips that can be assigned to the buses at depot $d \in \mathcal{D}$.

The MDVSP with trip shifting consists of finding a least-cost set of vehicle schedules that covers exactly one copy of each trip $v \in \mathcal{V}$ and respects vehicle availability per depot $d \in \mathcal{D}$. A vehicle schedule is a feasible sequence of trips that starts and ends at the same depot $d \in \mathcal{D}$. A schedule begins with a *pull-out* trip: the vehicle leaves its depot and travels without passengers to the start location of its first trip. Symmetrically, a schedule ends with a *pull-in* trip when the vehicle returns to its depot. In this sequence, there is a *connection* between every pair of consecutive trips v and v' covered using copies $k_v^i \in K_v$ and $k_{v'}^j \in K_{v'}$, respectively. Such a connection is feasible if a vehicle can cover v starting at time t_v^i before completing v' starting at time $t_{v'}^j$. Connections are of three kinds: if v ends at the start location of v', then the connection simply consists of waiting at this location; if v' starts at another location, the vehicle can make a *deadhead connection*, i.e., it makes a transit without passengers and then possibly waits at the second location. Finally, if time allows it, a vehicle can also go back to its depot before carrying out the second trip. If the latter option is less expensive than the first two, then it will be chosen. Obviously, a schedule for a vehicle of depot $d \in \mathcal{D}$ is feasible only if it contains trips in set $\mathcal{V}^d$. The cost structure includes a large fixed cost for each vehicle used, a cost per minute for a vehicle waiting in a location outside of a depot, and another cost per minute of travel without passengers. In contrast, waiting at the depot is free.

The model proposed for the MDVSP with trip shifting relies on the following additional notation. Let $\mathcal{S}^d$, $d \in \mathcal{D}$, be the set of all feasible schedules for the vehicles in depot d and let $\mathcal{S} = \bigcup_{d \in \mathcal{D}} \mathcal{S}^d$. For each depot $d \in \mathcal{D}$ and schedule $s \in \mathcal{S}^d$, we denote by c_s the cost of schedule s (including the vehicle fixed cost) and by a_{vs}, $v \in \mathcal{V}$, a binary parameter that is equal to 1 if and only if s covers trip v. As the start times of the copies of a same trip are assumed to be very close, we assume that a schedule cannot contain two copies or more of the same trip. Finally, we define a binary variable y_s that takes value 1 if and only if schedule s is part of the solution.

Using the above notation, the MDVSP with trip shifting can be formulated as the following set partitioning model:

$$\min \quad \sum_{s \in \mathcal{S}} c_s y_s \tag{1}$$

$$\text{s.t.} \quad \sum_{s \in \mathcal{S}} a_{vs} y_s = 1, \qquad \forall v \in \mathcal{V} \tag{2}$$

$$\sum_{s \in \mathcal{S}^d} y_s \leq b_d, \qquad \forall d \in \mathcal{D} \tag{3}$$

$$y_s \in \{0, 1\}, \qquad \forall s \in \mathcal{S}. \tag{4}$$

The objective function (1) aims at minimizing the total operational costs. Constraints (2) ensure that each trip is covered by a single vehicle whereas constraints (3) represent the vehicle availability at each depot. Notice that model (1)–(4) is identical to the set partitioning formulation proposed by Ribeiro and Soumis (1994) for the classical MDVSP (without trip shifting) except that the definitions of $\mathcal{S}^d$ and a_{vs} take into account that several copies are allowed for each trip $v \in \mathcal{V}$.

As the size of $\mathcal{S}$ grows exponentially with the number of trips, it is often impossible in practice to solve the integer program (1)–(4) as it is. As described in Section 3, we rather use a column generation algorithm to avoid the complete generation of set $\mathcal{S}$. In this algorithm, schedules in $\mathcal{S}$ are generated dynamically by solving shortest path problems defined on the networks introduced below.

For each depot $d \in \mathcal{D}$, there is one network $G_d = (N_d, A_d)$ that is used to generate schedules in $\mathcal{S}^d$, where N_d and A_d are the node and arc sets, respectively. To describe such a network, we proceed by steps. We start by describing a simple network that has been used for the classical MDVSP (see, e.g., Ribeiro and Soumis (1994)) and modify it progressively afterwards to reach the final network structure that we use. In the simple network, N_d contains a trip node for each trip $v \in \mathcal{V}^d$ as well as a source node n_d^0 and a sink node n_d^1, representing the start and the end of a schedule. In A_d, there exists a *connection* arc linking the nodes representing trips v and v' if these trips can be performed in sequence by a vehicle of depot d. Set A_d also contains *pull-out* arcs linking node n_d^0 to every trip node and *pull-in* arcs linking every trip node to node n_d^1. This network structure ensures that any feasible schedule for a vehicle of depot d can be represented by a path from n_d^0 to n_d^1 in G_d.

Here, we adopt a slightly different network (see Figure 1) that enables to reduce considerably the size of G_d. More specifically, we first remove all connection arcs involving a break at the depot and add, for each trip v, nodes n_v^{out} and n_v^{in} which model a departure from the depot to reach the start of trip v exactly at time t_v and a return to the depot from the end of trip v. We then add a pair of pull-out arc (n_v^{out}, v) and pull-in arc (v, n_v^{in}) for each trip v. After ordering chronologically the nodes n_v^{out} and n_v^{in}, $v \in \mathcal{V}$, according to the time at depot they represent, we create a chain of *wait-at-depot* arcs which links each pair of consecutive nodes in this set. In this way, the connection arcs involving a break at the depot can now be removed and be replaced by a pull-in trip to the depot, a sequence of waiting arcs at the depot and a pull-out arc. For example, in Figure 1, the possible connection between v_1 and v_4 (or v_2 and v_4) is not represented with a direct connection arc but by a sequence of arcs. Finally, we also remove all pull-out arcs (n_d^0, v) and pull-in arcs (v, n_d^1) for $v \in \mathcal{V}$ and add an arc from n_d^0 to the node n_v^{out} associated with the earliest time and symmetrically an arc from the last n_v^{in} node to n_d^1.

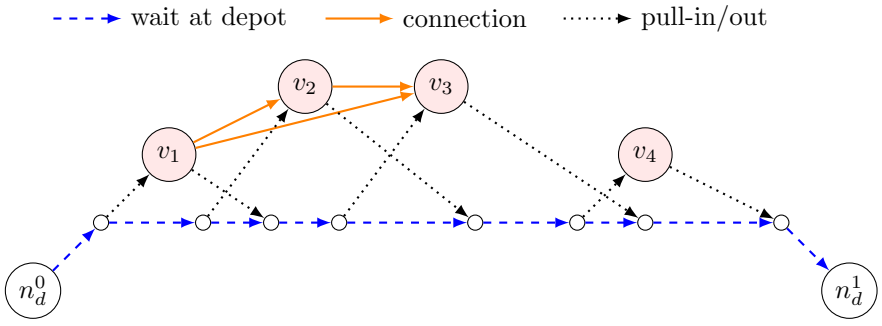
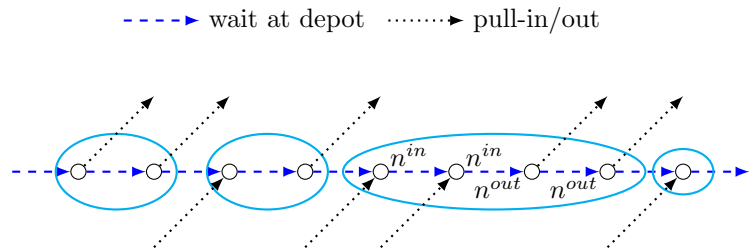


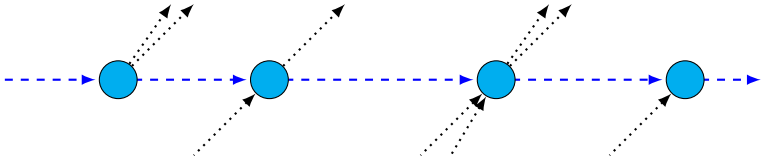
Figure 1: Network for the MDVSP with break aggregation at the depot

Next, we further reduce the number of arcs by aggregating some of the nodes n_v^{out} and n_v^{in}, and the wait-at-depot arcs. This reduction is illustrated in Figure 2: consecutive nodes that represent a return to the depot (type n^{in}) and the next group of consecutive nodes representing a departure from depot (type n^{out}) are aggregated into a unique node. This aggregation idea and the previous one are described in Desrosiers et al. (1995).

Now, if trip shifting is allowed, that is, K_v is not limited to a singleton for all trips $v \in \mathcal{V}$, the previous network is modified as follows: for each copy $k_v^i \in K_v$, we create a copy node. Node k_v^i is linked to node $k_{v'}^j$,



(a) Initial subnetwork



(b) Subnetwork after node aggregation

Figure 2: Aggregation of depot nodes

$v \neq v'$, if the sequence $k_v^i - k_{v'}^j$ is feasible. For example, if $K_v = \{k_v^1, k_v^2, k_v^3\}$ (numbered in ascending order of start times), we obtain three copy nodes for trip v and the corresponding connection arcs as illustrated in Figure 3. Because the start times of the copies of a same trip are, in general, close, the copy nodes of a same trip have almost identical incident connection arcs. To limit the number of these arcs in the network - and at the same time the number of equivalent solutions - we propose to adopt the subnetwork illustrated in Figure 4, that is, we create a chain of copy nodes that are linked by wait arcs. In doing so, we can aggregate some possible connections by a method similar to the one used to eliminate deadhead arcs in time-space networks (see Kliewer et al. (2006b)). Consider two trips v and v' which can be feasibly connected (v before v'); for each copy of v, we keep only the connection arc reaching the earliest possible copy of v'. Then, if several copies of v are linked to the same copy of v', we only keep the one starting from the latest copy of v.

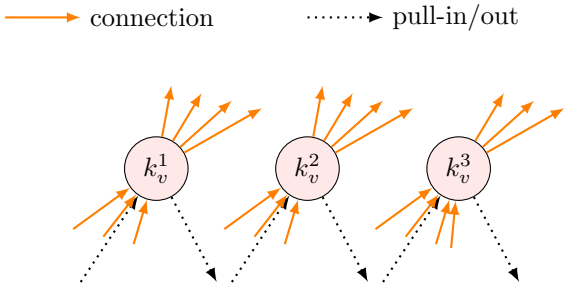


Figure 3: Representation of a trip v with $K_v = \{k_v^1, k_v^2, k_v^3\}$ and disconnected copy nodes

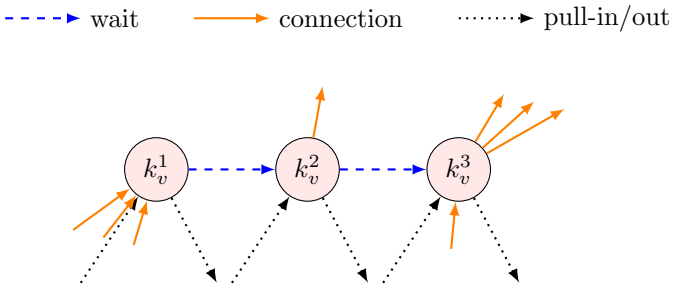


Figure 4: Subnetwork for a trip v with $K_v = \{k_v^1, k_v^2, k_v^3\}$

The final network G_d that we use to generate bus schedules for the MDVSP-CTS is illustrated in Figure 5: each trip that can be shifted is associated with a chain of arcs connecting copy nodes and the chains can be linked by one or several connection arcs. A path from n_d^0 to n_d^1 in G_d represents a feasible schedule for a vehicle assigned to depot d. The cost c_s of this schedule can be written as $c_s = \sum_{(i,j)\in A(s)} c_{ij}$, where $A(s)$ is the set of arcs in the path representing s and c_{ij} the cost associated with arc (i, j). The cost of an arc is equal to the sum of the vehicle travel cost and the driver salary cost (including cost for waiting outside the depot). The unique arc exiting the source node n_d^0 bears the vehicle fixed cost.

Note that we use a connection network (i.e., where connections between trips are explicitly represented by arcs) instead of a time-space network even it has been shown that the latter network structure can yield significant computational time reductions (see Kliewer et al. (2006b)). Our network choice ensues from the requirements of our industrial partner who seeks an easily adaptable model. Indeed, several operational constraints not considered in this paper (for instance, on connection times and on interlining) can be easily taken into account with a connection network, whereas they would require the addition of resource constraints in a time-space network. A detailed computational study for these different network types can be found in Desfontaines (2017) which shows that the proposed connection network and a time-space network yield similar computational performance in this context.

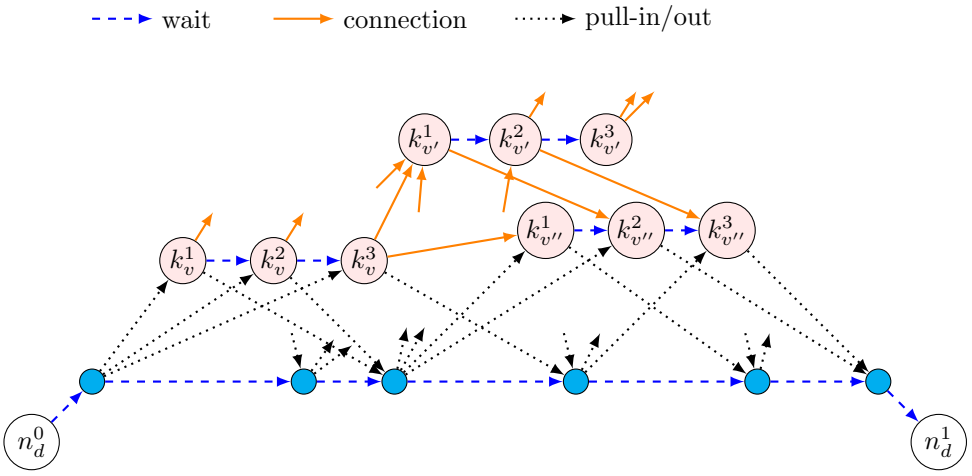


Figure 5: Final network structure for the MDVSP-CTS

2.2 Control of timetable quality

Even if the alternative trip start times are close to the initial start times, it is desirable that the modifications to the start times do not affect badly the quality of the initial timetable. In what follows, we consider three criteria to control timetable quality. The first one is the sum over all trips of the absolute differences between the initial trip start time and the final start time. This criterion aims at limiting the changes to the initial timetable. Secondly, we would like to preserve as much as possible the headways on each line. The last criterion consists in maintaining - or improving - the quality of targeted passenger connections. We suppose indeed that some trips are scheduled to ensure convenient connections between certain lines.

To model the controls on trip shifting and headways, we introduce an integer variable x_v for each trip $v \in \mathcal{V}$ which is equal to the amount of time (positive, negative, or zero) by which the start of trip v is shifted. This amount of time can be measured in minutes or in a larger time unit depending on the time step allowed for shifting, but it cannot exceed this time step. Notice that it is, therefore, possible to choose a more precise discretization for the final timetable (i.e., with the variables x_v) than for the definition of the copies in the sets K_v. For example, if the time step is equal to 2 minutes, then the set K_v for a trip $v \in \mathcal{V}$ may contain three copies with the following start times: $t_v - 2$, t_v, and $t_v + 2$. However, it is possible to restrict x_v to take a value in $\{-2, -1, 0, 1, 2\}$, allowing for additional start times $t_v - 1$ and $t_v + 1$. In the following, we assume that the possible start time deviations used to define the variables x_v are measured in the same time units as the trip copy start times (e.g., minutes).

The vehicle schedules selected in the solution imposes bounds on the variables x_v, $v \in \mathcal{V}$. Indeed, a schedule which covers trip v goes through the subnetwork associated with v as illustrated in Figure 4. For instance, the path representing this schedule may enter this subnetwork at copy node k_v^1, traverse the wait arc linking k_v^1 to k_v^2, and finally leave this subnetwork at copy node k_v^2. In this case, the schedule imposes the following bounds on x_v:

$$t_v^1 - t_v \leq x_v \leq t_v^2 - t_v. \tag{5}$$

In the general case, this box constraint can be written as:

$$\sum_{k \in K_v} \sum_{s \in \mathcal{S}} (t_v^k - t_v) e_{vs}^k y_s \leq x_v \leq \sum_{k \in K_v} \sum_{s \in \mathcal{S}} (t_v^k - t_v) o_{vs}^k y_s, \tag{6}$$

where e_{vs}^k (resp. o_{vs}^k), $s \in \mathcal{S}$, $v \in \mathcal{V}$, and $k \in K_v$, is a binary parameter that takes value 1 if schedule s enters (resp. leaves) the subnetwork of trip v by copy k and 0 otherwise.

To minimize the first criterion, we introduce the nonnegative variables q_v^+ and q_v^- for each $v \in \mathcal{V}$ that represent the amount of time by which trip v is shifted forward and backward, respectively, that is, we set

$$x_v = q_v^+ - q_v^-, \qquad \forall v \in \mathcal{V}. \tag{7}$$

To keep the initial timetable for trip v, q_v^+ and q_v^- must take value 0. To favor this, these variables bears a unit penalty cost α^T in the objective function.

Then we consider the set $\mathcal{H}$ of all pairs of trips for which it is desirable to keep the initial headway. Typically, this set contains all pairs of consecutive trips along the same line. For $(v, v') \in \mathcal{H}$, we define the nonnegative variables $h_{vv'}^+$ and $h_{vv'}^-$ to compute the positive and negative deviations from the initial headway between v and v', i.e.,

$$x_{v'} - x_v = h_{vv'}^+ - h_{vv'}^-, \qquad \forall (v, v') \in \mathcal{H}. \tag{8}$$

To keep the initial headway between trips v and v', $h_{vv'}^+$ and $h_{vv'}^-$ must take value 0. In the objective function, the variables $h_{vv'}^+$ and $h_{vv'}^-$ have a positive coefficient α^H to favor keeping the initial headways.

To take into account the last criterion, let $\mathcal{P}$ be the set of pairs of trips for which we would like to ensure a good passenger connection at a location where they intersect. Let $(v, v') \in \mathcal{P}$ be such a pair and we assume that, to yield a perfect passenger connection, the ideal relative shifting $x_{v'} - x_v$ between the start times of these trips should be equal to a pre-specified value $\tau_{vv'}$. To favor this trip shifting, let us introduce the nonnegative variables $r_{vv'}^+$ and $r_{vv'}^-$ which compute the positive and negative deviations from the ideal target shifting $\tau_{vv'}$, i.e.,

$$x_{v'} - x_v - \tau_{vv'} = r_{vv'}^+ - r_{vv'}^-, \qquad \forall (v, v') \in \mathcal{P}. \tag{9}$$

A connection between trips v and v' is considered ideal if $r_{vv'}^+ = r_{vv'}^- = 0$. If $x_v - x_{v'}$ slightly differs from $\tau_{vv'}$, the connection is often still possible but less convenient. On the contrary, if $x_{v'} - x_v$ is too far from $\tau_{vv'}$, the connection is considered inconvenient (either, it is missed or it requires a long waiting). To model these situations, we propose to define a penalty function $f_{vv'}^P$ that is applicable for the pair of trips $(v, v') \in \mathcal{P}$ as illustrated in Figure 6; the penalty for (v, v') is then given by $\alpha^P f_{vv'}^P(x_{v'} - x_v - \tau_{vv'})$, where α^P is a nonnegative weight which allows to adjust the passenger connection penalty costs against the other costs. For function $f_{vv'}^P$, the interval $[-U_{vv'}^-, U_{vv'}^+]$ designates the whole range of possible values of $x_{v'} - x_v - \tau_{vv'}$, whereas $[-u_{vv'}^-, u_{vv'}^+]$ is the range of values for which the connection is deemed convenient. A penalty $\alpha^P \beta_{vv'}$ is thus incurred for a deviation yielding an inconvenient connection (i.e., if $x_{v'} - x_v - \tau_{vv'}$ is outside the interval $[-u_{vv'}^-, u_{vv'}^+]$) while a unit penalty $\alpha^P \gamma_{vv'}^-$ (resp. $\alpha^P \gamma_{vv'}^+$) is paid for a negative (resp. positive) deviation in the interval $[-u_{vv'}^-, u_{vv'}^+]$.

To linearize function $f_{vv'}^P(x_{v'} - x_v - \tau_{vv'})$ for a connection induced by a pair of trips $(v, v') \in \mathcal{P}$, we introduce binary variables $z_{vv'}^+$ and $z_{vv'}^-$ which are equal to 1 if the deviation leads an inconvenient connection because $x_{v'} - x_v - \tau_{vv'} \in [-U_{vv'}^-, -u_{vv'}^-[$ or because $x_{v'} - x_v - \tau_{vv'} \in [u_{vv'}^+, U_{vv'}^+[$, respectively. Moreover, we define nonnegative variables $w_{vv'}^+$ and $w_{vv'}^-$ to compute the positive and negative deviations inside $[-u_{vv'}^-, u_{vv'}^+]$ which yields a linear penalty.

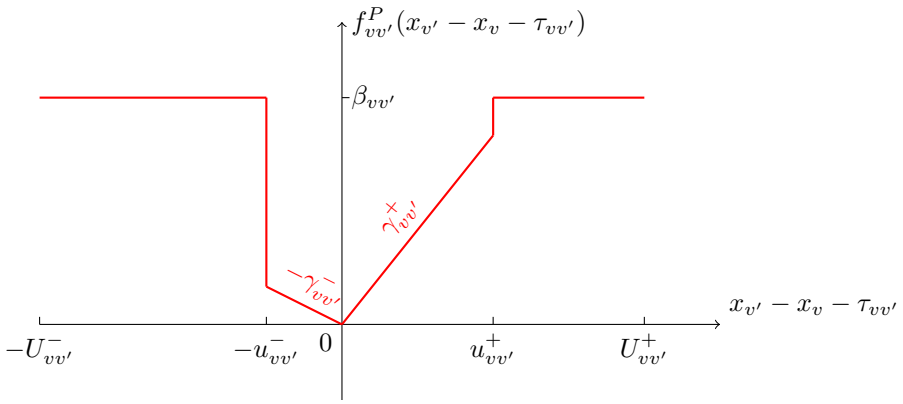


Figure 6: Definition of function $f^P_{vv'}(x_{v'} - x_v - \tau_{vv'})$

Given this notation, we can formulate the MDVSP-CTS as the following integer program:

$$\min \quad \sum_{s\in\mathcal{S}} c_s y_s + \alpha^T \sum_{v\in\mathcal{V}} (q^+_v + q^-_v) + \alpha^H \sum_{(v,v')\in\mathcal{H}} (h^+_{vv'} + h^-_{vv'}) + \alpha^P \sum_{(v,v')\in\mathcal{P}} (\beta_{vv'}(z^+_{vv'} + z^-_{vv'}) + \gamma^-_{vv'} w^-_{vv'} + \gamma^+_{vv'} w^+_{vv'}) \tag{10}$$

subject to

$$\sum_{s\in\mathcal{S}} a_{vs} y_s = 1, \qquad \forall v \in \mathcal{V} \tag{11}$$

$$\sum_{s\in\mathcal{S}^d} y_s \le b_d, \qquad \forall d \in \mathcal{D} \tag{12}$$

$$\sum_{k\in K_v} \sum_{s\in\mathcal{S}} (t^k_v - t_v) e^k_{vs} y_s \le x_v \le \sum_{k\in K_v} \sum_{s\in\mathcal{S}} (t^k_v - t_v) o^k_{vs} y_s, \qquad \forall v \in \mathcal{V} \tag{13}$$

$$x_v = q^+_v - q^-_v, \qquad \forall v \in \mathcal{V}, \tag{14}$$

$$x_{v'} - x_v = h^+_{vv'} - h^-_{vv'}, \qquad \forall (v,v') \in \mathcal{H} \tag{15}$$

$$x_v - x_{v'} - \tau_{vv'} = r^+_{vv'} - r^-_{vv'}, \qquad \forall (v,v') \in \mathcal{P} \tag{16}$$

$$w^+_{vv'} \ge r^+_{vv'} - z^+_{vv'} U^+_{vv'}, \qquad \forall (v,v') \in \mathcal{P} \tag{17}$$

$$w^-_{vv'} \ge r^-_{vv'} - z^-_{vv'} U^-_{vv'}, \qquad \forall (v,v') \in \mathcal{P} \tag{18}$$

$$r^+_{vv'} - u^+_{vv'} \le z^+_{vv'} (U^+_{vv'} - u^+_{vv'}), \qquad \forall (v,v') \in \mathcal{P} \tag{19}$$

$$r^-_{vv'} - u^-_{vv'} \le z^-_{vv'} (U^-_{vv'} - u^-_{vv'}), \qquad \forall (v,v') \in \mathcal{P} \tag{20}$$

$$x_v \in \mathbb{Z}, \quad q^-_v,\ q^+_v \ge 0, \qquad \forall v \in \mathcal{V} \tag{21}$$

$$h^-_{vv'},\ h^+_{vv'} \ge 0, \qquad \forall (v,v') \in \mathcal{H} \tag{22}$$

$$z^-_{vv'},\ z^+_{vv'} \in \{0,1\}, \quad r^-_{vv'},\ r^+_{vv'},\ w^-_{vv'},\ w^+_{vv'} \ge 0, \qquad \forall (v,v') \in \mathcal{P} \tag{23}$$

$$y_s \in \{0,1\}, \qquad \forall s \in \mathcal{S}. \tag{24}$$

In this model, we seek to minimize a weighted multi-objective function (10) of the operational costs and the penalties awarded for shifting trips, deviating from initial headways, and deviating from targeted timetables for ideal passenger connections. Constraints (11) and (12) ensure that all trips are covered with the available vehicles. Inequalities (13) link the schedule choice (variables y_s) to the possible deviations of the trip start times (variables x_v).

Constraints (14) and (15) define the variables q^+_v, q^-_v, $h^+_{vv'}$, and $h^-_{vv'}$, which allow to penalize trip shifting and deviations from the initial headways. Penalties for passenger connections are derived through the constraints (16)–(20). Constraints (17)–(20) linearize the penalty functions $f^P_{vv'}(x_{v'} - x_v - \tau_{vv'})$. In particular, for a pair of trips $(v,v') \in \mathcal{P}$, $w^+_{vv'}$ is set equal to $r^+_{vv'}$ by the corresponding inequality (17) unless the connection

is inconvenient, that is, $z^+_{vv'} = 1$. Similarly, the relations (18) determine the values of the variables $w^-_{vv'}$. If $r^+_{vv'} > u^+_{vv'}$, then $z^+_{vv'}$ is set to 1 thanks to the corresponding inequality (19). Similarly, the inequalities (20) set the values of the variables $z^-_{vv'}$. Finally, constraints (21)–(24) restrict the domains of the variables.

3 A two-phase matheuristic

For real-life instances, model (10)–(24) is of a very large size. Furthermore, the constraints (13)–(20) to control the timetable quality yield highly fractional linear relaxation solutions and, thus, very large branch-and-bound search trees. Therefore, to obtain computational times that are acceptable for the industry, we propose to solve the MDVSP-CTS using a two-phase matheuristic. In the first phase, the MDVSP with trip shifting is solved using a column generation heuristic described in Section 3.1. In the second phase, the trip timetable is adjusted according to the bus schedules determined in the first phase, with the aim of controlling the timetable quality. This adjustment is performed by solving a mixed integer program (see Section 3.2). Finally, in Section 3.3, we describe how penalties can be added to the first-phase model to favor a better control of the timetable quality.

3.1 Phase I: A column generation heuristic

The MDVSP with trip shifting solved in the first phase of the proposed matheuristic is modeled as the integer program (1)–(4). In practice, this model contains a huge number of variables. In fact, when the number of trips m grows, it becomes impossible to explicitly generate all schedules in $\mathcal{S}$. To overcome this drawback, we use a column generation algorithm to solve its linear relaxation (which is called the master problem or MP) and embed it in a diving heuristic to derive an integer solution. Given that high degeneracy occurs during the column generation process when solving large-scale instances, we also propose to use a perturbation method to fight degeneracy. All these components are detailed below.

3.1.1 Column generation

Column generation (see, e.g., Desaulniers et al. (2005); Lübbecke and Desrosiers (2005)) is an iterative algorithm that solves at each iteration a restricted master problem (RMP) and one or several pricing problems. The restricted master problem corresponds to the linear relaxation of (1)–(4) restricted to a relatively small subset of its variables, i.e., $\mathcal{S}$ is replaced by a subset $\hat{\mathcal{S}} \subset \mathcal{S}$. Once the RMP is solved in an iteration, the next step consists of trying to prove that the current solution of the RMP is also optimal for the MP, assuming that all variables y_s, $s \in \mathcal{S} \setminus \hat{\mathcal{S}}$, take value zero. To do so, one or several pricing problems are solved with the goal of finding negative reduced cost variables y_s with respect to the dual solution of the current RMP if some exist. When such variables are found, they are added to the RMP, i.e., the subset $\hat{\mathcal{S}}$ is augmented, and a new iteration is performed. Otherwise, the column generation process stops with an optimal solution to the MP as there are no more vehicle schedules $s \in \mathcal{S}$ associated with a negative reduced cost variable y_s.

For the MDVSP with trip shifting, there is one pricing problem per depot $d \in \mathcal{D}$ which is a shortest path problem defined on network G_d (see Section 2.1) but with modified arc costs as described in the following. Let $(\lambda_v)_{v\in\mathcal{V}}$ and $(\mu_d)_{d\in\mathcal{D}}$ be the dual variables associated with constraints (2) and (3) in the MP, respectively. The reduced cost $\tilde{c}_s$ of a schedule $s \in \mathcal{S}^d$ assigned to a vehicle of depot $d \in \mathcal{D}$ expresses as follows:

$$\tilde{c}_s = c_s - \sum_{v\in\mathcal{V}} a_{vs}\lambda_v - \mu_d = \sum_{(i,j)\in A(s)} \tilde{c}_{ij}, \tag{25}$$

where $\tilde{c}_{ij}$, $(i,j) \in A_d$, is the modified cost of arc (i,j) defined as follows:

$$\tilde{c}_{ij} = \begin{cases} c_{ij} - \mu_d & \text{if } i = n^0_d \\ c_{ij} - \lambda_v & \text{if } i \text{ is a copy node of trip } v \text{ and } j \text{ is not} \\ c_{ij} & \text{otherwise.} \end{cases} \tag{26}$$

At each iteration, these arc costs are updated according to the values taken by the dual variables in the solution of the current RMP. Then, each pricing problem is solved using a dynamic programming algorithm for a shortest path problem on an acyclic network.

3.1.2 Heuristic branching

To derive an optimal solution, column generation can be embedded in a branch-and-bound framework to yield an exact branch-and-price algorithm (see, e.g., Desaulniers et al. (2005); Barnhart et al. (1998)). In our computational study, we observe that too many branch-and-bound nodes needed to be evaluated before reaching an optimal solution, even for small-sized instances. Consequently, we opted for a branching heuristic, namely, a diving heuristic (see Joncour et al. (2010)) which proceeds as follows. At each iteration of this heuristic, a MP modified by fixed variables (except at the first iteration) is solved by column generation. If the computed solution is integer, the heuristic stops. Otherwise, a variable y_s among the ones taking the largest fractional value in this solution is selected and fixed to 1 before starting the next iteration. Consequently, this heuristic dives in the search tree and explores a single branch (possibly with multiple nodes) until ending column generation with an integer solution.

With this heuristic, the schedule of a vehicle is permanently fixed after each node and, thus, the number of nodes to explore is in the order of the number of vehicles used. However, one important potential drawback of this branching heuristic is that it is not possible to reconsider decisions made in ancestor nodes. This can potentially lead to poor solutions or no solution at all. Nevertheless, this situation did not occur in our computational experiments and very-close-to-optimality solutions were computed in almost all tested instances.

3.1.3 Constraint perturbation

One major difficulty with the set partitioning formulation is that it usually leads to degenerate problems, especially when the number of trips covered by each schedule is high. Oukil et al. (2007) and Benchimol et al. (2012) studied the effect of degeneracy on MDVSP instances and developed stabilized column generation and dynamic constraint aggregation methods to fight it. In this work, to reduce degeneracy, we rather use a simple constraint perturbation strategy (Charnes (1952)) that allows slight over-covering and under-covering of the trips.

In this approach, the MP is replaced by a perturbed version of it. To this end, we introduce perturbation variables η_v^+ and η_v^- for each trip, which indicate by how much trip $v \in \mathcal{V}$ is under- or over-covered, respectively. These variables are penalized in the objective function with small positive coefficients δ^+ and δ^- and restricted by upper bounds ϵ_v^+ and ϵ_v^-, which are randomly chosen as uniform variables in $[0, 0.01]$. The perturbed MP is given by:

$$\min \quad \sum_{s\in\mathcal{S}} c_s y_s + \sum_{v\in\mathcal{V}} (\delta^+\eta_v^+ + \delta^-\eta_v^-) \tag{27}$$

$$\text{s.t.} \quad \sum_{s\in\mathcal{S}} a_{vs} y_s + \eta_v^+ - \eta_v^- = 1, \qquad \forall v \in \mathcal{V} \tag{28}$$

$$\sum_{s\in\mathcal{S}^d} y_s \le b_d, \qquad \forall d \in \mathcal{D} \tag{29}$$

$$0 \le \eta_v^+ \le \epsilon_v^+, \qquad \forall v \in \mathcal{V} \tag{30}$$

$$0 \le \eta_v^- \le \epsilon_v^-, \qquad \forall v \in \mathcal{V} \tag{31}$$

$$0 \le y_s \le 1, \qquad \forall s \in \mathcal{S}. \tag{32}$$

This MP is less degenerate than the non-perturbed MP because many η_v^+ and η_v^- variables become basic variables. Note that fixing some y_s variables to one in the diving heuristic also fixes some of the perturbation variables to zero. Hence, the perturbed MP can be used throughout the solution process. However, we have chosen to remove perturbation when the value of the largest fractional-valued variable is less than the percentage of the integer-valued variables in the basis. In this way, the most uncertain decisions (i.e., the last ones) are made taking into account the most accurate information from the current linear relaxation. Usually, when perturbation is removed, many variables have been fixed and degeneracy is not a major issue for the rest of the solution process.

3.2 Phase II: Solving a mixed integer program

Now let us detail the second phase and, in particular, the way decisions of the first phase are taken into account in the second. The first phase fixes bus schedules, that is, sequences of trips and returns to the depot. Trip starting times should then be chosen with respect to those sequences.

Denote by $\mathcal{CV}$ the set of all pairs of trips performed consecutively (without returning to the depot) in a schedule selected in Phase I. In Phase II, if $(v, v') \in \mathcal{CV}$, then the values of x_v and $x_{v'}$ should be chosen to ensure that the connection between trips v and v' is feasible. Let $T_{vv'}$ be the minimum time required between the starting times of v and v' to ensure feasibility. The following inequalities must be imposed:

$$(t_{v'} + x_{v'}) - (t_v + x_v) \geq T_{vv'}, \qquad \forall (v, v') \in \mathcal{CV}. \tag{33}$$

Besides, Phase I also fixes pull-out and pull-in trips. To guarantee the feasibility of these trips without increasing waiting time outside a depot, additional constraints are imposed. Let $\mathcal{PO}$ and $\mathcal{PI}$ be the sets of all trips which are performed immediately after a pull-out trip and immediately before a pull-in trip, respectively. For each $v \in \mathcal{PO} \cup \mathcal{PI}$, we denote by $i(v)$ the index of the copy $k_v^{i(v)} \in K_v$ of the trip v selected in the Phase I solution. The additional constraints are as follows:

$$x_v \geq t_v^{i(v)} - t_v, \qquad \forall v \in \mathcal{PO} \tag{34}$$

$$x_v \leq t_v^{i(v)} - t_v, \qquad \forall v \in \mathcal{PI}. \tag{35}$$

Given these constraints which restrict the trip start times according to the vehicle schedules selected in Phase I, an optimal timetable subject to these constraints can be found by solving the following integer program:

$$\min \quad \alpha^T \sum_{v \in \mathcal{V}} (q_v^+ + q_v^-) + \alpha^H \sum_{(v,v') \in \mathcal{H}} (h_{vv'}^+ + h_{vv'}^-) + \alpha^P \sum_{(v,v') \in \mathcal{P}} (\beta_{vv'}(z_{vv'}^+ + z_{vv'}^-) + \gamma_{vv'}^- w_{vv'}^- + \gamma_{vv'}^+ w_{vv'}^+) \tag{36}$$

$$\begin{aligned}
\text{s.t.} \quad & (t_{v'} + x_{v'}) - (t_v + x_v) \geq T_{vv'}, && \forall (v, v') \in \mathcal{CV} && (37)\\
& x_v \geq t_v^{i(v)} - t_v, && \forall v \in \mathcal{PO} && (38)\\
& x_v \leq t_v^{i(v)} - t_v, && \forall v \in \mathcal{PI} && (39)\\
& d_v^{min} \leq x_v \leq d_v^{max}, && \forall v \in \mathcal{V} && (40)\\
& x_v = q_v^+ - q_v^-, && \forall v \in \mathcal{V}, && (41)\\
& x_{v'} - x_v = h_{vv'}^+ - h_{vv'}^-, && \forall (v, v') \in \mathcal{H} && (42)\\
& x_v - x_{v'} - \tau_{vv'} = r_{vv'}^+ - r_{vv'}^-, && \forall (v, v') \in \mathcal{P} && (43)\\
& w_{vv'}^+ \geq r_{vv'}^+ - z_{vv'}^+ U_{vv'}^+, && \forall (v, v') \in \mathcal{P} && (44)\\
& w_{vv'}^- \geq r_{vv'}^- - z_{vv'}^- U_{vv'}^-, && \forall (v, v') \in \mathcal{P} && (45)\\
& r_{vv'}^+ - u_{vv'}^+ \leq z_{vv'}^+ (U_{vv'}^+ - u_{vv'}^+), && \forall (v, v') \in \mathcal{P} && (46)\\
& r_{vv'}^- - u_{vv'}^- \leq z_{vv'}^- (U_{vv'}^- - u_{vv'}^-), && \forall (v, v') \in \mathcal{P} && (47)\\
& x_v \in \mathbb{Z}, \quad q_v^-,\ q_v^+ \geq 0, && \forall v \in \mathcal{V} && (48)\\
& h_{vv'}^-,\ h_{vv'}^+ \geq 0, && \forall (v, v') \in \mathcal{H} && (49)\\
& z_{vv'}^-,\ z_{vv'}^+ \in \{0, 1\}, \quad r_{vv'}^-,\ r_{vv'}^+,\ w_{vv'}^-,\ w_{vv'}^+ \geq 0, && \forall (v, v') \in \mathcal{P}. && (50)
\end{aligned}$$

The objective function (36) minimizes the sum of the penalties for shifting trips, deviating from the initial headways and deviating from the targeted timetables for ideal passenger connections. Constraints (37), (38) and (39) ensure compatibility with the Phase I decisions. Relations (40) bound the variables x_v to stay within their range of possible shifting values, where d_v^{min} and d_v^{max} are the minimal and maximal possible

deviations from the initial timetable for trip $v \in \mathcal{V}$. Finally, contraints (41)–(50) are identical to their counterparts (14)–(23).

Note that, instead of using constraints (37), we could have also considered inequalities (13) with fixed values for the variables y_s. This would have also led to a feasible timetable. However, constraints (37) are less restrictive as they allow to revise the trip copies selected in Phase I. Preliminary computational tests showed that using (37) instead of (13) yields a significant improvement in timetable quality.

3.3 Penalties for improved control of timetable quality

To limit the impact of the schedule choice in Phase I on the timetable selection in Phase II, we propose to add penalties on some arcs in the networks G_d, $d \in \mathcal{D}$, as illustrated in Figure 7 for a trip subnetwork with three copy nodes. These penalties are designed to limit the use of connection, pull-in and pull-out arcs that would constrain too much the solution space in Phase II. Inequalities (13) show, indeed, that for each trip $v \in V$, the value of x_v is restricted according to the first and the last copy node used to enter and leave the subnetwork representing trip v in the path (schedule) covering trip v.

For example, if $K_v = \{k_v^1, k_v^2, k_v^3\}$ contains three copies for a trip v and k_v^2 is the initial one, we add a relatively large penalty L to all pull-in and connection arcs leaving node k_v^1 as, with these arcs, (13) restricts x_v to a single value. Symmetrically, a penalty L is added to all pull-out and connection arcs entering node k_v^3. We also add a smaller penalty l to all connection and pull-in/out arcs linked to the copy node k_v^2, those arcs having a smaller but still important influence on the value that can be taken by x_v.

In the next section, we present the results of an empirical study to evaluate the influence of these penalties. We will test networks with three and five copies for each trip. The penalties are set as in the example above for the case with three copies per trip. When five copies are considered instead of three, the penalties $2L$, L, l, $l/2$, and 0 (resp. 0, $l/2$, l, L, and $2L$) are added to the arcs leaving (resp. entering) the first, second, ..., and fifth copy node of a trip, respectively. This means that an arc leaving the second copy node has an additional penalty L and an arc entering this node bears an additional penalty $l/2$.

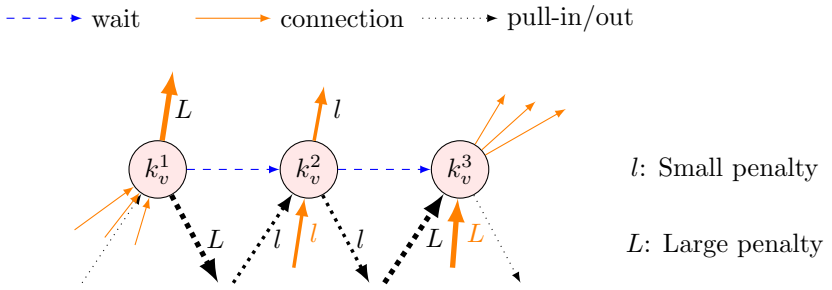


Figure 7: Penalties l and L for a trip v subnetwork with copy nodes $K_v = \{k_v^1, k_v^2, k_v^3\}$

4 Computational results

Our computational study is based on four real-life MDVSP instances provided by GIRO Inc. These instances involves 1042 to 2116 trips and will be referred to as R1, R2, R3 and R4. In order to perform computational tests on a large number of smaller problems, we built five artificial instances by randomly selecting 500 trips from R1. All instances have two depots.

We consider the following operational costs: 1000 per vehicle used, 0.4 per minute of travel (either on a connection or a pull-in/pull-out arc), and 0.2 per minute of waiting time outside a depot. Besides it is forbidden to have a connection between two trips which lasts more than 45 minutes without returning to the depot. If not otherwise specified, we choose $\alpha^T = 1$, $\alpha^H = 2$, $\alpha^P = 4$ as penalty costs for the timetable optimization.

To evaluate the impact of trip shifting, we consider three different patterns for the trip copies. In the first one, labeled 3_2min, all trips have three copies: the first one is shifted 2 minutes forward, the second is the initial trip and the last one is delayed by 2 minutes. In other words, for each trip $v \in \mathcal{V}$, the possible shifting times $t_v^i - t_v$ for $k_v^i \in K_v$ are $\{-2, 0, 2\}$. We also define the copy pattern 5_1min with the shifting times $\{-2, -1, 0, 1, 2\}$ and 5_2min with the shifting times $\{-4, -2, 0, 2, 4\}$. In Phase II, we keep the same time step as the one used for the copies in Phase I, i.e., 2 minutes for patterns 3_2min and 5_2min, and 1 minute for 5_1min. Moreover to ease the comparison between the different patterns, all results regarding timetable quality are provided in minutes (so those from patterns 3_2min and 5_2min are multiplied by 2).

We did not have access to data on the passenger connections and, therefore, we did a limited number of tests involving the control of passenger connections, always using the trip copy pattern 3_2min. For instances with 500 trips, we created a subset of 50 passenger connections by randomly selecting 50 pairs of trips from different lines with overlapping timetables. For every pair $(v, v') \in \mathcal{P}$, we use the following parameter values: $\beta_{vv'} = 1$, $u^+_{vv'} = 1$, $\gamma^+_{vv'} = \frac{1}{3}$, $u^-_{vv'} = 0$, and $\alpha^-_{vv'} = 0$. Furthermore, the value of $\tau_{vv'}$ is randomly chosen in $\{-2, -1, 0, 1, 2\}$ (or, equivalently, $\{-4, -2, 0, 2, 4\}$ in the time units used to define the trip copy start times) with the following corresponding probabilities: $\{0.05, 0.2, 0.5, 0.2, 0.05\}$. This way, we consider that, on average, half of the passenger connections are perfectly synchronized in the initial timetable and the other half can be improved

All computational tests were conducted on a Linux machine equipped with an Intel Core i7-3770 processor clocked at 3.40 GHz and a RAM of 8Gb. The column generation heuristic was coded using the GENCOL library, version 4.5. All linear programs in the column generation heuristic and mixed integer programs in Phase II were solved using the commercial solver CPLEX 12.4.

4.1 Tests on 500-trip instances

In this section, we report the results that we obtained on the five instances with 500 trips that we created from instance R1. We conducted experiments with and without considering the control of the passenger connections because they were generated randomly. We study first the performance of the 2-phase heuristic in terms of computational time and solution quality (Table 1) and then the trade-off between the scheduling cost and the timetable quality (Table 2) when no control on the passenger connections is considered ($\mathcal{P} = \emptyset$). The influence of adding passenger connection control is discussed afterwards (Table 3).

For all tests that we conducted (including for the real-life instances), the Phase II model (36)–(50) was solved to optimality in less than one second. Because of their small magnitude, the Phase II computational times will not be reported hereafter, neither will we discuss the quality of the Phase II solutions which are optimal.

Table 1 reports computational results that can be used to evaluate the performance of the Phase I column generation heuristic and the quality of the computed solutions. These results were obtained using different values of the penalties l and L for the three tested copy patterns in the case with no passenger connection control. The tested values for these penalties look somewhat arbitrary but they were chosen so as to generate solutions that exhibit different scheduling cost and timetable quality, approximating a Pareto-frontier for each copy pattern. Each line in this table (except those specifying averages over all tests realized for a copy pattern) corresponds to a given pattern and a given choice of penalty values and provides average results computed over the five instances with 500 trips. Besides the pattern and penalty values, the columns display: the value of the computed solution (UB); a lower bound on the optimal value (LB); the difference in percentage between these two values (GAP); the total number of linear relaxation solved (# LrS); the total computation time (Total); the time for solving the root node (Root); and the time dedicated to the pricing problems (Pricing). All time are in seconds. Note that the lower bounds were obtained separately by solving to optimality the master problem at the root node without constraint perturbation.

From these results, we observe first that the optimality gaps are, on average, very small in most cases, with the largest gaps occurring for the most flexible copy pattern. This shows that the branching heuristic is efficient at finding near-optimal solutions. For all instances and scenarios, the average number of linear relaxations solved is equal more or less to the corresponding number of vehicles used (see Table 2) plus

Table 1: Heuristic performance in terms of solution quality and computational time

Copy Pattern	Penalties l	Penalties L	UB	LB	GAP (%)	# LrS	CPU time (s) Total	Root	Pricing
3_2min	0.0	0.0	33579.48	33577.26	6.6×10^{-3}	33.2	123.8	79.5	12.9
	0.1	0.5	33731.14	33731.14	0.0	33.4	114.2	83.0	11.2
	1	2	34053.00	34053.00	0.0	33.8	135.3	90.5	12.1
	5	10	34443.36	34440.78	7.5×10^{-3}	33.8	151.8	102.9	13.4
	0.5	5	34035.40	34035.40	0.0	33.6	132.8	96.1	12.5
	1	5	34115.92	34115.74	5.3×10^{-4}	33.4	127.3	92.2	12.1
	1	10	34166.44	34166.42	6.0×10^{-5}	33.6	131.9	94.5	11.8
	0.5	10	34086.88	34086.88	0.0	33.2	138.1	97.2	13.5
	10	50	34898.24	34898.24	0.0	34.4	154.3	110.4	13.4
	100	500	37849.56	37849.56	0.0	36.0	126.2	92.2	11.9
		Avg.			$\mathbf{1.5 \times 10^{-3}}$	**33.8**	**133.6**	**93.8**	**12.5**
5_1min	0	0	33570.20	33569.25	2.8×10^{-3}	33.4	126.8	82.6	18.7
	1	2	34043.70	34043.59	3.2×10^{-4}	33.4	134.2	97.8	17.4
	1	5	34108.06	34107.99	2.1×10^{-4}	33.8	134.1	100.6	17.8
	1	10	34170.22	34170.22	0.0	33.4	144.8	104.2	18.4
	10	50	35012.36	35012.36	0.0	34.0	164.2	114.3	21.4
	5	50	34859.20	34859.11	2.6×10^{-4}	33.8	160.0	116.5	20.1
		Avg.			$\mathbf{6.0 \times 10^{-4}}$	**33.6**	**144.0**	**102.7**	**19.0**
5_2min	0	0	29149.28	28789.77	1.2	29.4	226.8	133.4	31.3
	1	5	30482.38	30172.55	1.0	30.0	257.6	153.3	32.9
	1	10	30944.64	30706.13	0.78	29.8	284.9	154.2	34.7
	5	50	34243.40	34225.89	0.051	31.6	248.2	153.7	37.3
	1	50	33514.64	33485.15	0.088	31.4	200.2	135.1	26.1
	10	50	34850.08	34849.10	2.8×10^{-3}	32.2	222.8	147.0	30.5
	10	100	35783.96	35783.96	0.0	33.6	197.3	135.9	30.5
		Avg.			**0.4**	**31.1**	**234.0**	**144.6**	**31.9**

two, namely, the root node linear relaxation and the linear relaxation solved after removing constraint perturbation. This indicates that the computed master problem solutions are highly fractional and that very large search trees should be explored to find an optimal solution and prove its optimality. The average computational times show that most of the time (around 60-70% on average) is spent solving the root relaxation and that solving the RMPs is more time consuming than solving the pricing problems. This is not surprising given that degeneracy yielded by relatively dense columns (around 15 trips per schedule on average) remains high despite the use of constraint perturbation. Furthermore, notice that the addition of intermediate trip copies (pattern 5_1min versus 3_2min) has a limited impact (+8%) on the total computational time. In contrast, adding more copies with larger trip shiftings (pattern 5_2min versus 3_2min) seems to have a greater impact (+74%) on the computational time which may be due to slightly denser columns (almost 18 trips per schedule on average).

For the same instances and scenarios, we report in Table 2 computational results that can be used to evaluate the trade-offs between schedule cost and timetable quality. The first line in this table corresponds to the classical MDVSP, i.e., without trip shifting. Each other line refers to a given pattern and penalty choice. Notice that the first line for each pattern (with $l = L = 0$) gives the best scheduling cost computed for the MDVSP with trip shifting and no control of the timetable quality. Here again, each line reports average results computed over five instances with 500 trips. The columns in Table 2 provide in order: the copy pattern; the values of l and L; the total scheduling cost (Value I), i.e., the value computed using the objective function (1) in Phase I but without considering the penalties l and L; the operational cost (Op. cost), i.e., without the vehicle fixed costs; the number of vehicles used (# bus); the optimal value of the Phase II solution (Value II); the sum of minutes of trip shifting (T mod.), i.e., $\sum_{v \in \mathcal{V}} (q_v^+ + q_v^-)$; and the sum of minutes of modified headways (H mod.), i.e., $\sum_{(v,v') \in \mathcal{H}} (h_{vv'}^+ + h_{vv'}^-)$.

Table 2: Scheduling cost versus timetable quality for different choices of penalties (l, L)

Copy Pattern	Penalties l	Penalties L	Scheduling Value I	Scheduling Op. cost	Scheduling # Bus	Timetable quality Value II	Timetable quality T mod.	Timetable quality H mod.
MDVSP	0	0	36441.0	1441.0	35.0	0.0	0.0	0.0
3_2min	0	0	33579.5	1179.5	32.4	1822.4	692.8	564.8
	0.1	0.5	33635.7	1235.7	32.4	693.6	223.2	235.2
	1	2	33834.6	1434.6	32.4	224.8	63.2	80.8
	0.5	5	33875.8	1475.8	32.4	97.6	24.8	36.4
	1	5	33905.5	1505.5	32.4	85.6	21.6	32.0
	5	10	34171.4	1771.4	32.4	65.6	15.2	25.2
	1	10	33941.0	1541.0	32.4	57.2	12.4	22.4
	0.5	10	33922.1	1522.1	32.4	55.6	13.2	21.2
	10	50	34232.2	1832.2	32.4	43.6	9.2	17.2
	100	500	36469.6	1869.6	34.6	4.0	0.8	1.6
5_1min	0	0	33570.2	1170.2	32.4	1736.2	697.0	519.6
	1	2	33856.8	1456.8	32.4	119.2	31.2	44.0
	1	5	33919.8	1519.8	32.4	52.4	13.2	19.6
	1	10	33944.5	1544.5	32.4	40.4	10.0	15.2
	10	50	34220.4	1820.4	32.4	34.6	8.6	13.0
	5	50	34141.2	1741.2	32.4	34.2	10.2	12.0
5_2min	0	0	29149.3	949.3	28.2	4139.2	1346.4	1396.4
	1	5	29690.9	1490.9	28.2	614.0	197.2	208.4
	1	10	29753.0	1553.0	28.2	563.6	180.4	191.6
	1	50	31403.6	1603.6	29.8	256.4	72.4	92.0
	5	50	31745.4	1945.4	29.8	230.8	68.4	81.2
	10	50	32249.1	2049.1	30.2	177.2	51.6	62.8
	10	100	33733.0	2133.0	31.6	85.2	20.4	32.4

First, one can observe from these results that allowing trip shifting without controlling the timetable quality can lead to significant reductions of the scheduling cost, with savings of at least 2.6 vehicles and 18% of the operational costs. However, to achieve such savings, a large number of trips need to be shifted, resulting in an important perturbation of the initial timetable. On the other hand, considering positive penalties l and L in Phase I can yield a very significant improvement of the timetable quality depending on their values. For example, for pattern 3_2min with $l = 0.5$ and $L = 5$, only 12.4 trips are shifted by 2 minutes on average $(\sum_{v\in\mathcal{V}}(q_v^+ + q_v^-) = 24.8)$, while using an average of only 32.4 vehicles to cover all bus trips.

The different trade-offs between the scheduling cost (Value I) and the timetable quality (Value II) are illustrated in Figures 8 and 9, where the latter figure zooms around the points for patterns 3_2min and 5_1min. Each point represents a different choice of penalties (l, L). The points associated with a given pattern give an approximation of the Pareto-frontier for this pattern and highlight the compromise that must be made between scheduling cost and timetable quality when deciding which solution to implement. Clearly, we see that much better timetable quality can be achieved without increasing much the scheduling cost obtained without any control. From Figure 8, we can also observe the huge impact of allowing more flexibility in trip shifting by considering the copy pattern 5_2min. The number of vehicles used and the operational costs can be substantially decreased compared to the solutions produced with pattern 3_2min.

We also investigate the impact of adding the control of passenger connections. To that end, we present in Table 3 the Phase II results obtained for the scenario with $l = 0.5$, $L = 10$ and copy pattern 3_2min when the three criteria (shiftings, headways, and passenger connections) are simultaneously taken into account. For these tests, we varied the values of the parameters α^T, α^H, and α^P to favor differently the three criteria from one test to another. The results correspond again to averages over the five 500-trip instances. In this table, the first three columns provide the parameter values, the next two columns (T mod. and H mod.) are defined as above, and the last column (P mod.) gives the sum of the unweighted penalties relative to the

passenger connections, i.e., $\sum_{(v,v')\in\mathcal{P}} f^P_{vv'}(x_{v'} - x_v - \tau_{vv'})$. When no trip shifting is allowed, this sum is equal to 19.1. The bold values indicate the best value obtained during our tests for each criterion.

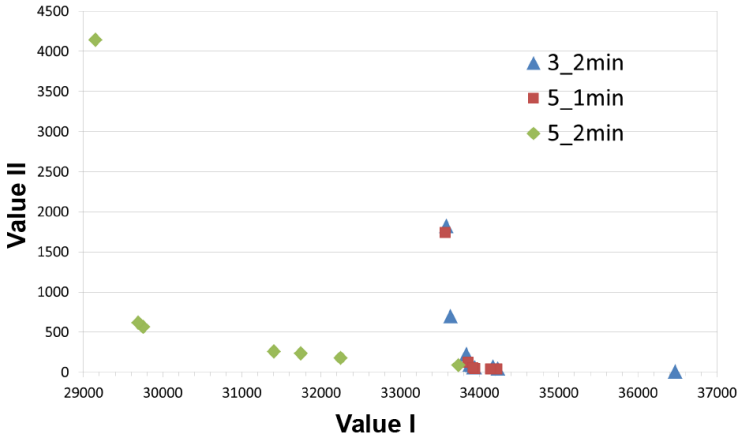


Figure 8: Timetable quality in function of scheduling cost for different penalties (l, L) and copy patterns 3_2min, 5_1min and 5_2min

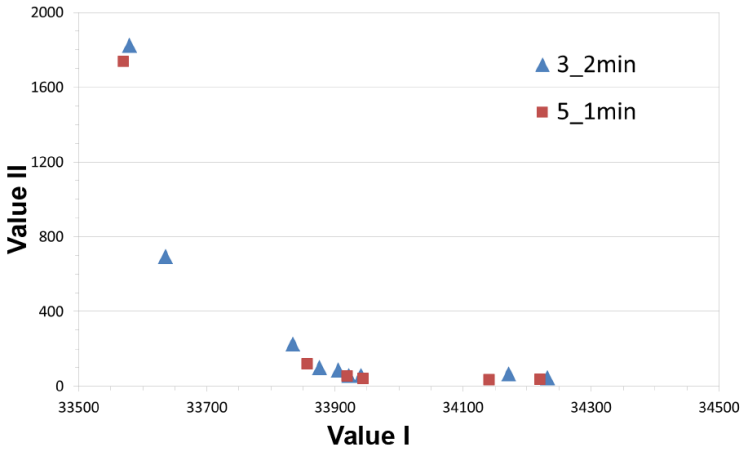


Figure 9: Timetable quality in function of scheduling cost for different penalties (l, L) – Focus on copy patterns 3_2min and 5_1min

Table 3: Timetable quality with passenger connection control ($l = 0.5$, $L = 10$, and pattern 3_2min)

α^T	α^H	α^P	T mod.	H mod.	P mod.
1	2	0.1	12.4	21.6	19.3
1	2	4	25.6	26.4	14.7
1	2	10	61.6	68.0	**4.4**
1	10	0.1	47.6	**14.4**	20.3
1	10	10	66.0	18.4	16.0
10	1	0.1	**12.0**	22.0	19.1

These results show that different trade-offs can be achieved depending on the objectives of the planner. Furthermore, when the focus is put on the control of the passenger connections ($\alpha^T = 1$, $\alpha^H = 2$, and $\alpha^P = 10$), then it is possible to greatly improve the initial timetable for this criterion and often reach the targeted timetables for ideal passenger connections.

4.2 Tests on real-life instances

In this section, we investigate the computational results obtained by the proposed 2-phase heuristic on the four real-life instances R1 to R4. All tests were performed with the copy pattern 3_2min and did not include passenger connection control. First, we present in Table 4 the Phase I results when solving the MDVSP with uncontrolled trip shifting ($l = L = 0$). The reported statistics are the same as in Table 1 in addition to the number of trips in each instance (m). From these results, we can attest from the gaps that the computed solutions are optimal or almost optimal for all instances except instance R1. However, for this instance, the lower bound was computed using constraint perturbation (which further relaxes the master problem) as we were unable to solve the unperturbed root node master problem within a reasonable amount of time. Consequently, the 1.26% gap reported for this instance clearly overestimates the real optimality gap. The computational times clearly show that the instance R1 is the most difficult to solve. This is due to the average column density which is much higher for this instance: in the computed solution, there are on average more than 25 trips per schedule compared to less than 13 for the other instances. Such a high density yields much more degeneracy in the RMPs which require a large proportion of the total computational time (more than 96% for R1). Note that similar results were obtained when we solved these instances with various values for l and L.

Table 4: Heuristic performance for real-life instances (with $l = L = 0$)

						CPU time (s)		
Instance	m	**UB**	**LB**	**GAP(%)**	**# LrS**	**Total**	**Root**	**Pricing**
R1	1042	41922.8	*41394.3*	*1.26*	40	6530.6	1950.8	248.1
R2	1042	120056.6	120056.6	0.0	109	250.5	196.5	20.6
R3	1219	103907.8	103907.7	9.6×10^{-5}	101	1188.2	747.7	227.2
R4	2116	236960.6	236957.5	1.3×10^{-3}	225	2642.7	2177.8	233.9

Finally, we analyze in Table 5 different solutions for these four instances obtained by varying the penalties l and L. For each instance, the first line corresponds to the MDVSP solution and the subsequent lines are presented in decreasing order of the scheduling cost (Value II). We thus obtained once again a set of possible solutions with different scheduling cost and timetable quality. The values of the penalties l and L seems to have a similar effect for all instances. Notice though that $(l, L) = (0.5, 10)$ dominates $(l, L) = (1, 10)$ for instances R1 and R2 and $(l, L) = (1, 5)$ for R3. It would be interesting to further investigate the influence of these penalty values on both objectives in order to be able to choose suitable values to get a desired timetable quality. This choice probably depends on the number of trips and the number of trips per vehicle. From the results in Table 5, we deduce that allowing trip shifting can lead to substantial reductions (between 1.7 and 6.8%) in the number of vehicles used for these real-life instances. With these reduced numbers of vehicles, it is possible to obtain a timetable of high quality, but at the expense of increased operational costs.

5 Conclusion

In this work, we introduced a new model and a matheuristic for solving the MDVSP-CTS. A primary focus of our work was to propose a model that can be easily adapted to many practical situations. Our choices of the underlying network structure (connection-based networks), methodology (column generation), and problem decomposition (the timetable quality control does not perturb much the MDVSP with trip shifting) were guided by this adaptability goal. Still our 2-phase heuristic demonstrates good performances in terms of computational time. Furthermore, even if we resort to a heuristic branching strategy in Phase I, we observed that it led to optimal or near-optimal solutions in most cases.

To extend the present work, we envision the following directions. The first one is to explore possible strategies to speed up the column generation heuristic used to solve the MDVSP with trip shifting in Phase I. Another possible avenue consists of better understanding the integration of the two phases of the algorithm, for eventually finding a way to treat these phases iteratively and tend towards an optimal solution. Finally, another possible extension that has a great potential is to tackle the simultaneous vehicle and crew scheduling problem with controlled trip shifting.

Table 5: Comparison of scheduling cost and timetable quality for real-life instances

			Scheduling			Timetable quality		
Instance	l	L	**Value I**	**Op. cost**	**# Bus**	**Value II**	**T mod.**	**H mod.**
	MDVSP		45343.8	1343.8	44	0	0	0
	0	0	41922.8	922.8	41	2318	1290	514
	1	2	42400.8	1400.8	41	310	122	94
R1	0.5	10	42408.4	1408.4	41	190	62	64
	1	10	42463.4	1463.4	41	200	84	58
	1	5	42465.2	1465.2	41	182	74	54
	10	50	44131.2	2131.2	42	62	14	24
	MDVSP		122391.0	5391.0	117	0	0	0
	0	0	120056.6	5056.6	115	2278	1062	608
	1	2	120335.0	5335.0	115	242	78	82
R2	1	5	120422.8	5422.8	115	74	30	22
	0.5	10	120432.4	5432.4	115	60	24	18
	1	10	120449.0	5449.0	115	60	24	18
	10	50	121021.0	6021.0	115	52	20	16
	MDVSP		109374.2	4374.2	105	0	0	0
	0	0	103907.8	3907.8	100	3698	1718	990
	1	2	104257.6	4257.6	100	390	154	118
R3	0.5	10	104336.0	4336.0	100	68	28	20
	1	5	104349.0	4349.0	100	94	46	24
	1	10	104397.2	4397.2	100	66	34	16
	10	50	105678.8	5678.8	100	58	26	16
	MDVSP		246239.6	10239.6	236	0	0	0
	0	0	236960.6	8960.6	228	4862	2458	1202
	1	2	237828.0	9828.0	228	678	274	202
R4	1	5	238210.2	10210.2	228	222	90	66
	0.5	10	238264.8	10264.8	228	194	58	68
	1	10	238317.0	10317.0	228	180	64	58
	10	50	239147.8	11147.8	228	148	44	52

References

C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. Operations Research, 46(3):316–329, 1998.

P. Benchimol, G. Desaulniers, and J. Desrosiers. Stabilized dynamic constraint aggregation for solving set partitioning problems. European Journal of Operational Research, 223(2):360–371, 2012.

A. A. Bertossi, P. Carraresi, and G. Gallo. On some matching problems arising in vehicle scheduling models. Networks, 17(3):271–281, 1987.

S. Bunte and N. Kliewer. An overview on vehicle scheduling models. Public Transport, 1(4):299–317, 2009.

G. Carpaneto, M. Dell'amico, M. Fischetti, and P. Toth. A branch and bound algorithm for the multiple depot vehicle scheduling problem. Networks, 19(5):531–548, 1989.

A. A. Ceder. Optimal multi-vehicle type transit timetabling and vehicle scheduling. Procedia-Social and Behavioral Sciences, 20:19–30, 2011.

A. Charnes. Optimality and degeneracy in linear programming. Econometrica, 20(2):160–170, 1952.

G. Desaulniers and M. D. Hickman. Public transit. In G. Laporte and C. Barnhart, editors, Transportation, Handbooks in Operations Research and Management Science, Volume 14, pages 69–127. Elsevier, Amsterdam, 2007.

G. Desaulniers, J. Lavigne, and F. Soumis. Multi-depot vehicle scheduling problems with time windows and waiting costs. European Journal of Operational Research, 111(3):479–494, 1998.

G. Desaulniers, J. Desrosiers, and M. M. Solomon. Column generation, volume 5. Springer, New York, 2005.

L. Desfontaines. Problème d'horaire d'autobus avec dépôts multiples et modification contrôlée des heures de début des voyages. Master's thesis, Polytechnique Montréal, Canada, 2017.

J. Desrosiers, Y. Dumas, M. M. Solomon, and F. Soumis. Time constrained routing and scheduling. In Network Routing, Handbooks in Operations Research and Management Science, Volume 8, pages 35–139. Elsevier Science B.V., Amsterdam, 1995.

M. Groiez, G. Desaulniers, A. Hadjar, and O. Marcotte. Separating valid odd-cycle and odd-set inequalities for the multiple depot vehicle scheduling problem. EURO Journal on Computational Optimization, 1(3):283–312, 2013.

P. C. Guedes and D. Borenstein. Column generation based heuristic framework for the multiple-depot vehicle type scheduling problem. Computers & Industrial Engineering, 90:361–370, 2015.

P. C. Guedes, W. P. Lopes, L. R. Rohde, and D. Borenstein. Simple and efficient heuristic approach for the multiple-depot vehicle scheduling problem. Optimization Letters, 10(7):1449–1461, 2016.

V. Guihaire and J.-K. Hao. Transit network timetabling and vehicle assignment for regulating authorities. Computers & Industrial Engineering, 59(1):16–23, 2010.

A. Hadjar and F. Soumis. Dynamic window reduction for the multiple depot vehicle scheduling problem with time windows. Computers & Operations Research, 36(7):2160–2172, 2009.

A. Hadjar, O. Marcotte, and F. Soumis. A branch-and-cut algorithm for the multiple depot vehicle scheduling problem. Operations Research, 54(1):130–149, 2006.

O. Ibarra-Rojas, F. Delgado, R. Giesen, and J. Muoz. Planning, operation, and control of bus transport systems: A literature review. Transportation Research Part B: Methodological, 77:38–75, 2015.

O. J. Ibarra-Rojas, R. Giesen, and Y. A. Rios-Solis. An integrated approach for timetabling and vehicle scheduling problems to analyze the trade-off between level of service and operating costs of transit networks. Transportation Research Part B: Methodological, 70:35–46, 2014.

C. Joncour, S. Michel, R. Sadykov, D. Sverdlov, and F. Vanderbeck. Column generation based primal heuristics. Electronic Notes in Discrete Mathematics, 36:695–702, 2010.

N. Kliewer, S. Bunte, and L. Suhl. Time windows for scheduled trips in multiple depot vehicle scheduling. In Proceedings of the EWGT2006 joint conferences, pages 340–346, 2006a.

N. Kliewer, T. Mellouli, and L. Suhl. A timespace network based exact optimization model for multi-depot bus scheduling. European Journal of Operational Research, 175(3):1616–1627, 2006b.

N. Kliewer, B. Amberg, and B. Amberg. Multiple depot vehicle and crew scheduling with time windows for scheduled trips. Public Transport, 3(3):213–244, 2012.

G. Laporte, F. A. Ortega, M. A. Pozo, and J. Puerto. Multi-objective integration of timetables, vehicle schedules and user routings in a transit network. Transportation Research Part B: Methodological, 98:94–112, 2017.

B. Laurent and J.-K. Hao. Iterated local search for the multiple depot vehicle scheduling problem. Computers & Industrial Engineering, 57(1):277–286, 2009.

A. Lbel. Vehicle scheduling in public transit and Lagrangean pricing. Management Science, 44(12):1637–1649, 1998.

T. Liu and A. A. Ceder. Integrated public transport timetable synchronization and vehicle scheduling with demand assignment: A bi-objective bi-level model using deficit function approach. Transportation Research Procedia, 23: 341–361, 2017.

M. Lübbecke and J. Desrosiers. Selected topics in column generation. Operations Research, 53(6):1007–1023, 2005.

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

A.-S. Pepin, G. Desaulniers, A. Hertz, and D. Huisman. A comparison of five heuristics for the multiple depot vehicle scheduling problem. Journal of Scheduling, 12(1):17–30, 2009.

H. L. Petersen, A. Larsen, O. B. Madsen, B. Petersen, and S. Ropke. The simultaneous vehicle scheduling and passenger service problem. Transportation Science, 47(4):603–616, 2012.

C. C. Ribeiro and F. Soumis. A column generation approach to the multiple-depot vehicle scheduling problem. Operations Research, 42(1):41–52, 1994.

V. Schmid and J. F. Ehmke. Integrated timetabling and vehicle scheduling with balanced departure times. OR Spectrum, 37(4):903–928, 2015.

A. Van den Heuvel, J. Van Den Akker, and M. Van Kooten. Integrating timetabling and vehicle scheduling in public bus transportation. Reporte Técnico UU-CS-2008-003, Department of Information and Computing Sciences, Utrecht University, Holanda, 2008.