The vehicle allocation problem: alternative formulation and branch-and-price method

Cesar Alvarez Cruz [a], *Alysson M. Costa* [b], *Pedro Munari* [a], *Reinaldo Morabito* [a]

Production Engineering Department, Federal University of São Carlos [a],

School of Mathematics and Statistics, The University of Melbourne [b],

cesarknbv@gmail.com, alysson.costa@unimelb.edu.au, munari@dep.ufscar.br, morabito@ufscar.br

Abstract

The Vehicle Allocation Problem (VAP) consists of repositioning empty vehicles across a set of terminals over a given planning horizon so as to maximize the profits generated from serving demand for transportation of goods between pair of terminals. This problem has been classically modeled using an extended space-time network which captures the staging of the decision-making process. The present paper proposes a new mixed-integer programming (ILP) model based on the idea of representing the demands to be met as nodes on a graph. We also derive a Dantzig-Wolfe reformulation which is solved with a branch-and-price (BP) method. The proposed BP uses a stabilized interior-point column generation approach and a branching procedure that imposes constraints in the master problem, thus not damaging the structure of the subproblems. Additionally, we show that these subproblems can be solved efficiently using a shortest path algorithm on a directed acyclic graphs. Computational results are carried out on a realistic-sized benchmark instances, commonly used in the literature. The results show the efficacy of the proposed strategies. In particular, the BP method solved the whole set of instances to proven optimality for the first time and in faster competitive times.

Keywords: Vehicle allocation problem, Dantzig-Wolfe decomposition, branch-and-price, empty repositioning, logistics.

1 Introduction

In customized-service freight transportation, customers request services for transporting loads between terminals during a given period. After completing these services, empty vehicles may accumulate faraway from where they are currently or prospectively needed, thus, creating geographical imbalances between the supply and demand of empty vehicles (Hall, 1999). The Vehicle Allocation Problem (VAP) emerges as a model for carriers to support the decision making of how to efficiently reposition their fleet of empty vehicles over a given planing horizon so as to serve these demands for transportation of loads between terminals while maximizing the profits resulting from these operations.

The VAP can be seen as part of the family of problems that regard the (re)positioning of empty containers in the shipping industry (Holmberg et al., 1998; Joborn et al., 2004; Narisetty et al., 2008; Gorman, 2015; Heydari and Melachrinoudis, 2017; Hungerländer and Steininger, 2019) and empty rail-cars in the railroad industry (Cheung and Chen, 1998; Shintani et al., 2007; Song and Dong, 2015; Meng and Wang, 2011). Some important details set apart the problems in these two contexts from the trucking industry, all of them related to the limitations presented by the physical network. In the trucking system, whenever a load and a driver are available, the truck can deadhead directly to where the load is located. On the contrary, even when empty resources are available (containers and railcars) at a given terminal, fixed schedules for moving resources (ships and trains) have to be followed and respected. In addition, the quantity of available resources may exceed the capacity of the moving resources, which further limits the leverage of empty resource across the network.

The specific case of the trucking industry, namely the VAP, has been studied both from deterministic and stochastic perspectives. Stochastic approaches take explicitly in consideration the uncertainties associated with the demands in the system but are obviously more difficult to solve for large-scale problems. They also require data that is often absent in industrial scenarios. Deterministic approaches, on the other hand, are valued by practitioners as they are usually easier to understand and implement. In this latter case, uncertainty can still be incorporated by the use of "what-if" scenarios and parameter sensitivity analysis.

The VAP was first considered by Powell (1986), who modeled a stochastic version of the problem in which demands are uncertain using a space-time extended network: each terminal is replicated over the planning horizon using an appropriate time granularity. The model considers three kinds of decisions: the sending of empty vehicles, the sending of loaded vehicles and the holding of vehicles idle in between periods. The author uses the Franke-Wolfe algorithm (Frank and Wolfe, 1956) to solve to optimality an instance with 10 terminals and 7 periods. Dejax and Crainic (1987) propose a taxonomy of empty repositioning problems, studying models and algorithms for the motor carrier, the container and the railroad industries, looking to increase the knowledge on these systems and identify possible trends for future research. Frantzeskakis and Powell (1990) and Cheung and Powell (1996) approached the VAP with approximate multi-stage stochastic programming approaches, using linear and concave approximations of the recourse function, respectively. Powell and Carvalho (1998a,b), in turn, proposed an approximated method called Logistics Queueing Network (LQN) based on the linear approximation for solving the dynamic programming formulation of the problem.

Vasco and Morabito (2016) considers multiple types of vehicles (multicommodity) resulting from prohibiting certain routes for certain trucks, a modelling strategy that can be used to represent real-life scenarios of Brazilian truck operators. The authors presented a MIP formulation and a number of metaheuristics, such as GRASP, simulated annealing and ant colony optimization, to obtain good feasible solutions for instances that general-purpose ILP solvers could not provide competitive solutions. Cruz et al. (2020) applied the Dantzig-Wolfe decomposition to the formulation presented in Vasco and Morabito (2016) and proposed a branch-and-price (BP) algorithm based on the resulting model. The algorithm was able to solve to optimality large-scale instances representative of Brazilian logistics operators that general-purpose solvers could not solve.

This article introduces an alternative formulation for the VAP tackled by Vasco and Morabito (2016) and Cruz et al. (2020). This model is based on the idea of representing the problem on a graph whose nodes correspond to each demand arc of the space-time network and whose arcs correspond to empty repositioning paths for traveling between these demand arcs. The size of this new ILP model depends on different parameters than the one used in earlier works (namely, the number of requests and number of vehicle leaving out the number of terminals and periods) This proves to be useful in solving the realistic-sized instances used in Vasco and Morabito (2016) and Cruz et al. (2020) with general-purpose ILP solvers. We show that the proposed compact formulation leads to a better performance of the solvers for the above-mentioned instances, which are representative of large-scale logistics operators, in comparison with the compact formulation of Vasco and Morabito (2016). As a second contribution, we use the Dantzig-Wolfe decomposition to reformulate the alternative formulation and show that the resulting pricing problem can be solved efficiently using shortest path algorithms on directed acyclic graphs. Finally, as a third main contribution, we propose a branch-and-price (BP) method based on a stabilized column generation algorithm that resorts to well-centered dual solutions obtained by an interior point method (Gondzio et al., 2013, 2016; Munari and Gondzio, 2013, 2015). We explore a hierarchical branching procedure composed by rules that impose constraints in the master problem, in the search for optimal integer solutions. The computational results of the BP method show time-efficiency computational advantages over other exact methods proposed in the literature, namely the approaches proposed in Cruz et al. (2020). Indeed, the new approach allowed for the obtention, for the fist time, of optimal solution to all benchmark instances.

The remainder of this paper is organized as follows. In Section 2, we first present a description of the VAP and introduce the relevant notation; then, in the sake of completeness, we state the ILP model used by Vasco and Morabito (2016) and Cruz et al. (2019, 2020) and present an illustrative example to guide the reader. In Section 3, we propose the alternative formulation based on a different network representation of the problem. In Section 4, we present the Dantzig-Wolfe decomposition for this new formulation, and describe the specialized algorithm for the pricing problem. In Section 5, we describe the proposed BP method and its hierarchical branching procedure. In Section 6, computational experiments based on realistic-sized problem instances are presented and analyzed, showing the potential practical benefits of the proposed approach. Finally, in Section 7, we draw our conclusions and suggest some future research perspectives.

2 The vehicle allocation problem

In this section, we describe the relevant parameters for the VAP and define the notation used in the remainder of the paper. Then, we present the arc-demand formulation used by Vasco and Morabito (2016) and Cruz et al. (2020). We close this section with an illustrative example.

2.1 Problem description and notation

Let N be the set of terminals, T be the set of time periods, and V be the set of vehicles. The fleet of vehicles is considered heterogeneous as each vehicle can be a company-owned vehicle or it belongs to an independent contractor or a third-party carrier, among others, in practical settings (Vasco and Morabito (2016)). The vehicles can be different in terms of the cost of moving the empty vehicle between two terminals, in terms of the profit (income minus direct operational costs) obtained by meeting a load demand between two terminals with the vehicle, in terms of the availability of the vehicle in a terminal at some time period, and in terms of the possible restrictions for moving the vehicle between two terminals (as defined below by problem parameters c_{ijv}, p_{ijv}, m_{itv} and A_{ijv}, respectively). However, it is assumed that the company-owned and the contracted vehicles have the same load capacity because the consolidated load demand between two terminals in a time period is described in terms of load units, each unit composed of enough freight to completely fill a vehicle (for instance, in some Brazilian road carriers, the load unit corresponds to a 30-ton truck). Moreover, it is assumed that the vehicles spent the same time for traveling between two terminals as, for simplicity, this time represents the average travel time based on an average speed, no matter the vehicle used (as defined below by problem parameter τ_{ij}). Consider the following parameters:

- τ_{ij} : travel time from terminal i to terminal j, $\forall i, j \in N$.
- d_{ijt} : demand for transportation services (number of loaded vehicles) from i to j beginning at time t, $\forall i, j \in N$, $\forall t \in T$.
- p_{ijv} : profit (income minus direct operational costs) obtained by serving the route from terminal i to terminal j with vehicle v, $\forall i, j \in N$, $\forall v \in V$.
- c_{ijv} : cost of moving the empty vehicle $v, \forall v \in V$ from terminal i to terminal j, $\forall i, j \in N$, $\forall v \in V$.
- m_{itv} : takes the value of 1 if vehicle v enters (i.e., become available) the system at terminal i at time t, $\forall i \in N$, $\forall t \in T$, .
- A_{ijv} : restriction of movement between terminals i and j for vehicle type v, being 1: if the vehicle is allowed to move, and 0: otherwise, $\forall i \in N$, $j \in N$ and $v \in V$.

The VAP consists of determining the best allocation of vehicles to serve the requested transportation services, in order to maximize the total profit that takes into account the profit from transportation services and the cost of moving empty vehicles. Each transportation service is a request between nodes i and j, at time t, that requires d_{ijt} vehicles and results in a profit p_{ijv}, when executed by vehicle v. There is no requirement on fulfilling all services, hence, up to d_{ijt} vehicles can be assigned to the corresponding request in an optimal solution. Every time a vehicle v goes empty from i to j (repositioning), a cost c_{ijv} is incurred. Note that vehicle v cannot move (either loaded or empty) from i to j if $A_{ijv} = 0$. Such restrictions are relevant from a practical point of view, as pointed out in Vasco and Morabito (2016), as they restrict certain routes on account of the following situations: (i) routes that are not on the way, or do not pass through towns where the driver's family live, (ii) lack of truck stops for maintenance, (iii) nonexistence of points for the exchange of drivers, (iv) an independent contractor or a third-party carrier that is not familiar with some routes, (v) physical conditions of the roads versus the maintenance conditions of the vehicles, (vi) risk of robbery or truck hijacking on some routes, (vii) in some cases outsourced fleet drivers do not like to be held idle away from their home city ($A_{iiv} = 0$) and so on.

The problem is usually represented using an extended time-space network defined by nodes that correspond to each pair (i, t), for $i \in N$ and $t \in T$; arcs connecting any two nodes (i, t) and $(i, t+1)$, for $i \in N$ and $t \in T$, with $t < |T|$; and arcs connecting any two nodes (i, t) and (j, t') such that $t' \geq t + \tau_{ij}$, for all $i, j \in N$ and $t, t' \in T$. Fig. 1 shows an example of a time-space network for $N = \{1, ..., 6\}$ and $T = \{1, ..., 6\}$. In this example, there are four requests for freight transportation services ($d_{422} = 3, d_{562} = 1, d_{233} = 3, d_{515} = 2$) and two vehicles (i.e., $|V| = 2$). Vehicle 1 becomes available at terminal 5 and time period 1 ($m_{611} = 1$) while vehicle 2 becomes available at terminal 4 and time period 2 ($m_{422} = 1$). The bold arrows represent the arcs where there is demand for loaded trips and the short arrows represent supply of vehicles at the terminals. There is an extra column of terminals ($t > 6$) representing all terminal-period pairs extending beyond the planning horizon. In addition, there is a sink n_f capturing all the flow coming from the extended space-time network, which is helpful is defining the proposed formulation of Section 3.

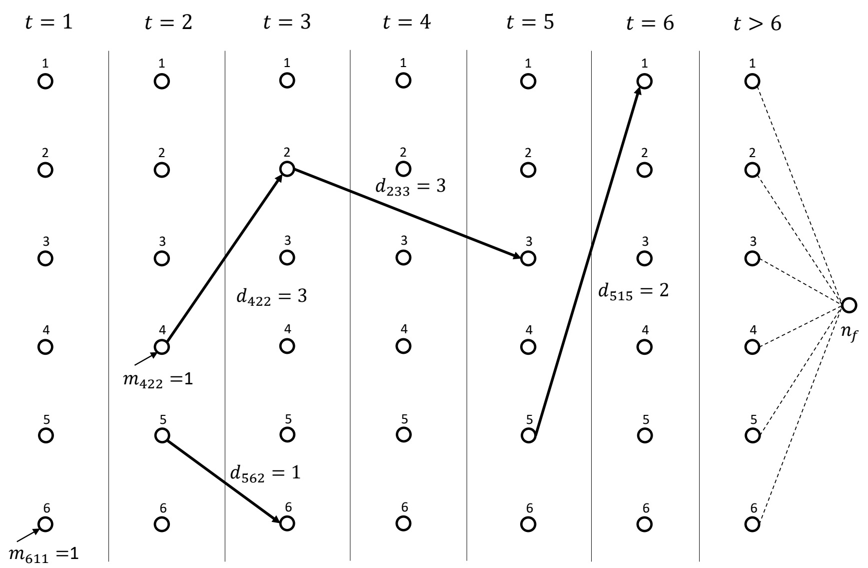


Fig. 1: The problem's parameters.

2.2 Arc-demand formulation

We present an ILP formulation for the VAP that can be seen as an extension of the multicommodity minimum cost flow problem over a space-time directed graph (Vasco and Morabito, 2016; Cruz et al., 2019, 2020). We explicitly describe this model as later on in the paper we present comparison results of computational times between this model and the alternative model proposed in Section 3. Hereafter, we refer to the following model as the arc-demand formulation because demand for transportation of goods is assigned to the arcs in the space-time graph. To state this model, we use the following decision variables:

- x_{ijtv} : takes the value of 1 if vehicle $v \in V$ start moving a load from terminal i to terminal j beginning at time t to satisfy the demand d_{ijt}, $\forall i \in N$, $j \in N$ and $t \in T$, and 0 otherwise;
- y_{ijtv} : takes the value of 1 if vehicle $v \in V$ start moving a empty from terminal i to terminal j beginning at time t, $\forall i \in N$, $j \in N$ and $t \in T$, and 0 otherwise.

From the above definitions, the arc-demand formulation is written as follows:

$$\max \quad \sum_{i \in N} \sum_{\substack{j \in N \\ i \neq j}} \sum_{t \in T} \sum_{v \in V} (p_{ijv} x_{ijtv} - c_{ijv} y_{ijtv}) \tag{1}$$

$$\text{s.t.} \quad \sum_{j \in N} (x_{ijtv} + y_{ijtv}) - \sum_{\substack{k \in N, \\ k \neq i, \\ t > \tau_{ki}}} \left(x_{ki(t-\tau_{ki})v} + y_{ki(t-\tau_{ki})v} \right) - y_{ii(t-1)v} = m_{itv}, \qquad \forall i \in N, \forall t \in T, \forall v \in V, \tag{2}$$

$$\sum_{v \in V} x_{ijtv} \leq d_{ijt}, \qquad \forall i, j \in N, \forall t \in T, \tag{3}$$

$$\sum_{t \in T} x_{ijtv} = 0 \text{ and } \sum_{t \in T} y_{ijtv} = 0, \text{ if } A_{ijv} = 0, \qquad \forall i, j \in N, \forall v \in V, \tag{4}$$

$$x_{ijtv}, y_{ijtv} \in \{0, 1\}, \qquad \forall i, j \in N, \forall t \in T, \forall v \in V. \tag{5}$$

The objective function (1) seeks to maximize the total profit over the multiperiod planning horizon, which is equal to the income generated from the loaded vehicle trips minus the costs of the empty vehicle trips. Constraints (2) guarantee the conservation of the flow for each vehicle v (loaded and/or empty) at each terminal i at time t. Constraints (3) establish an upper bound to the loaded trips between a given pair of terminals, which is equal to the demand on that route. Constraints (4) establish the trips (either loaded or empty) that cannot be made by each vehicle.

Constraints (5) state the domain of the decision variables.

To illustrate an optimal solution of model (1)-(5), consider the example defined at the end of Section 2.1, with the extended space-time network given in Fig. 1. We assume that trips are prohibited between terminals 1 and 2 for both vehicles, i.e., $A_{121} = A_{211} = A_{122} = A_{212} = 0$. Additionally, consider the travel times defined in Table 1 and the costs and profits defined in Tables 2 and 3 for each vehicle. Fig. 2 illustrates an optimal solution for this example, whose value is 7.8. We represent loaded trips using bold arrows and empty trips using dashed arrows. Dashed arcs between the same terminal represent vehicles being held idle. In the solution, vehicle 1 follows path $\{(6,5,1),(5,6,2),(6,5,3),(5,5,4),(5,1,5),(1,1,6)\}$ and vehicle 2 follows path $\{(4,2,2),(2,3,3),(3,3,5),(3,3,6)\}$. Note that vehicle 2 executes two empty trips between terminals 6 and 5 for repositioning, one at time period 1 and the other at time period 3.

	Travel time (τ_{ij})					
i/j	1	2	3	4	5	6
1	1	2	1	1	1	2
2	2	1	2	1	2	3
3	1	2	1	1	3	2
4	1	1	1	1	1	3
5	1	2	3	1	1	1
6	2	3	2	3	1	1

Tab. 1: Travel times between terminals of the example.

	Cost for vehicle 1 (c_{ij1})						Cost for vehicle 2 (c_{ij2})					
i/j	1	2	3	4	5	6	1	2	3	4	5	6
1	0	1	2	2	2	2	0	3	3	2	2	2
2	1	0	2	2	2	2	3	0	3	3	2	2
3	2	2	0	2	1	1	3	3	0	1	2	2
4	2	2	2	0	1	1	2	3	1	0	3	3
5	2	2	1	1	0	1	2	2	2	3	0	3
6	2	2	1	1	1	0	2	2	2	3	3	0

Tab. 2: Cost of empty vehicle movement of the example.

	Profit for vehicle 1 (p_{ij1})						Profit for vehicle 2 (p_{ij2})					
i/j	1	2	3	4	5	6	1	2	3	4	5	6
1	0	1.8	3.6	3.6	3.6	3.6	0	4.2	4.2	3.6	3.6	3.6
2	1.8	0	3.6	3.6	3.6	3.6	4.2	0	4.2	4.2	3.6	3.6
3	3.6	3.6	0	3.6	1.8	1.8	4.2	4.2	0	4.5	3.6	3.6
4	3.6	3.6	3.6	0	3.6	3.6	3.6	4.2	4.5	0	4.2	4.2
5	3.6	3.6	1.8	3.6	0	1.8	3.6	3.6	3.6	4.2	0	4.2
6	3.6	3.6	1.8	3.6	1.8	0	3.6	3.6	3.6	4.2	4.2	0

Tab. 3: Profits of the example.

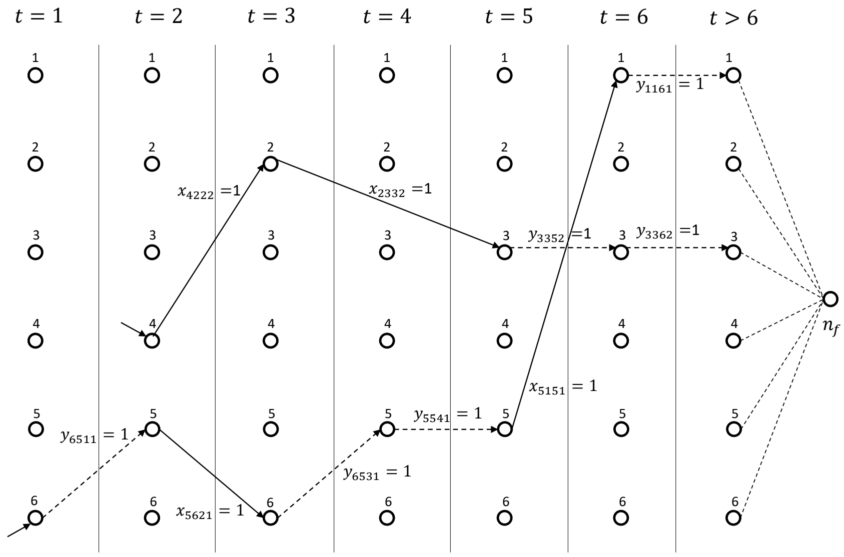


Fig. 2: An optimal solution for the example.

It should be observed that if the fleet of vehicles is homogeneous (i.e., all vehicles have the same cost, profit and availability parameters c_{ij}, p_{ij} and m_{it}), the resulting model (without constraints (3) and (4) and with integer variables $0 \leq x_{ijt} \leq d_{ijt}$ and $y_{ijt} \geq 0$ defined as the number of vehicles that start moving a load from terminal i to terminal j at time t to satisfy demand d_{ijt} and the number of vehicles that start moving empty from terminal i to terminal j at time t, respectively) can be optimally solved as a linear programming model (Ghiani et al. (2004))

3 Node-demand formulation

To introduce the alternative formulation, we consider the notation presented in Section 2 and the additional sets and parameters defined as follows. Let R be the set of all requests for freight transportation services along the planning horizon. There is one request $r \in \{1, .., |R|\}$ for each triplet (i, j, t) such that $d_{ijt} > 0, \forall i, j \in N, t \in T$. Each request $r \in R$ is determined by the departing terminal $i_r \in N$, the delivery terminal $j_r \in N$, the starting travel period $t_r \in T$ and the number of vehicles needed to meet the demand $D_r = d_{i_r j_r t_r}$. Based on this, the VAP consists in determining feasible sequences of requests, such that each sequence is executed by a given available vehicle in a way that maximizes the overall profit. In addition, two artificial requests (depots) are added: the 0 depot from which all sequences of requests have to depart and the $|R|+1$ depot where all sequences of requests arrive at. We use k_v and h_v to denote the location and period, respectively, at which vehicle v appears according to the previous definition m_{itv} used in the arc-demand formulation of

Section 2.2. For example, $m_{422} = 1$ is represented as $k_1 = 4$ and $h_1 = 2$.

We then use a *request network* to represent the problem, instead of the space-time network used in the arc-demand formulation. In this request network, there is one node for each request $r \in R$, and one arc (r, s) for each pair of requests $r, s \in R$ such that it is feasible to serve request r and then s consecutively, using the same vehicle (of at least one type). When the delivery terminal of r is the same as the departing terminal of s (i.e., $j_r = i_s$), then the traversal of arc (r, s) means that the vehicle finished request r and is ready to start request s, without repositioning. Otherwise, the arc traversal corresponds to an empty movement of the vehicle, repositioning from j_r to i_s.

Figure 3 illustrates the request network for the same example introduced in Section 2.2. Circles represent requests whereas triangles represent both artificial depots. The numbers above the circles represent the demand D_r while the triplets within contain the departing terminal $i_r \in N$, the delivery terminal $j_r \in N$ and the starting travel period $t_r \in T$, which will be necessary in the profit/cost definition. Note that by ordering the request nodes according to their starting time period t_r, we inherit the acyclic property from the formulation of Section 2. The arcs in this network represent the transition of vehicles either between two requests or between a request and a depot (and viceversa). Note that this transition can represent no event at all when the delivery terminal of r (j_r) is the same as the departing terminal of s (i_s). Finally, in comparison to the space-time network, this alternative network shows a reduced number of nodes and arcs for instances with a moderate number of requests. For example, the 30 node and 432 arc space-time network presented in Fig. 1 at the end of Section 2.1 is reduced to a network of only 6 nodes and 30 arcs, totaling 30 decision variables and 18 constraints. Fig. 4 shows the same optimal solution illustrated in Fig. 2 but considering the request network.

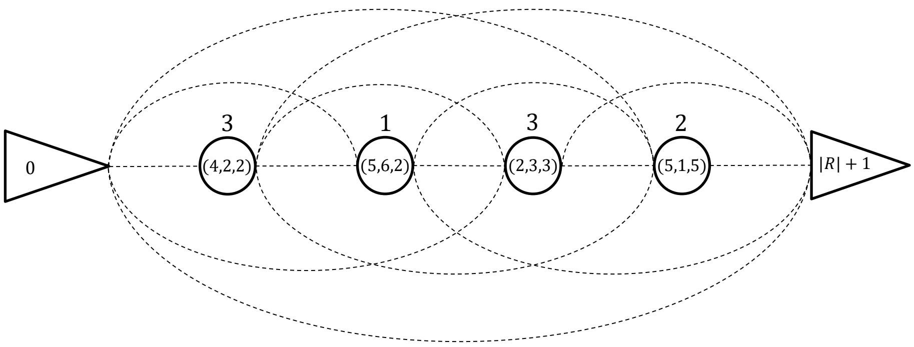


Fig. 3: Graphic representation of a request network for the same example in Fig. 1.

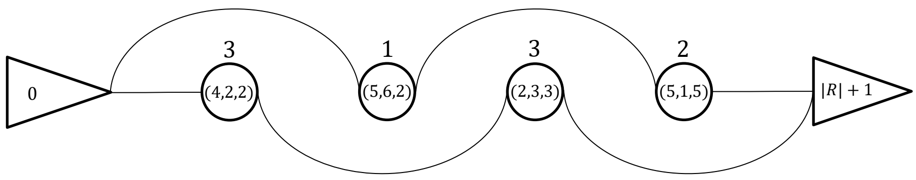


Fig. 4: An optimal solution in the request network for the same example in Fig. 2.

In the same way as the demands, profits and costs from vehicles movements have to be redefined in terms of requests. Let P_{rsv} be either the profit or the cost associated to meeting request $s \in R$ consecutively after meeting request $r \in R$ with vehicle $v \in V$. As mentioned before, the traversal of arc (r, s) may require an empty movement of the vehicle, which may shadow the profit related to meet the requests and, hence, $P_{rsv} \geq 0$ corresponds to a profit, or to a cost otherwise. These values are straightforward to compute if the triangle inequality holds for the costs and if we assume that any vehicle can move from a given terminal to any other. However, these are not valid in the VAP because of the restriction of movements. Indeed, if $A_{j_r i_s v} = 0$ for $j_r \neq i_s$, the empty movement may still be possible, but through the shortest path between these terminals, which is a consequence of the fact that optimal solutions for each vehicle $v \in V$ in the time-space network are longest paths since they are maximizing profits (see Lemma 4.1 in Cruz et al. (2020)). Hence, we use the shortest path of each vehicle type $v \in V$ to calculate the profits of the following events that can occur between two nodes of the request network: (a) traveling empty between the entering location of the vehicle (k_v) and a request; (b) traveling empty between the delivery terminal of one request and the departing terminal of its consecutive request; (c) traveling empty between the delivery terminal of a request and the end of the planning horizon; and (d) traveling empty between the entering location of the vehicle and the end of the planning horizon. Note that in events (c) and (d) traveling empty is still needed when holding vehicles in inventory is not possible (as is the case in some of the realistic-sized instances considered in this paper).

From the previous definition of possible events between nodes, P_{rsv} is defined in terms of parameters p_{ijv} and c_{ijv} as follows:

$$P_{rsv} = \begin{cases} p_{i_r j_r v} - \text{SP}((j_r, t_r + \tau_{i_r j_r}), (i_s, t_s), v), & \text{if } r, s \in R \\ p_{i_r j_r v} - \text{SP}((j_r, t_r + \tau_{i_r j_r}), n_f, v) & \text{if } r \in R \wedge s = |R| + 1 \\ -\text{SP}((k_v, h_v), (i_s, t_s), v), & \text{if } r = 0 \wedge s \in R \\ -\text{SP}((k_v, h_v), n_f, v), & \text{if } r = 0 \wedge s = |R| + 1 \end{cases}$$

where $\text{SP}(A, B, v)$ denotes the shortest path for a vehicle of type v between nodes A and B of the time-space network and arc lengths correspond to empty vehicle costs c_{ijv}. This can be efficiently computed in advance and is used as an input parameter in the model. Whenever a vehicle ends attending a request or enters the system at a given terminal where it is not allowed to be held idle, it is still necessary to compute the shortest path to n_f if there is no request to attend afterwards (Cases 2 and 4 of the profit definition). Figure 5 shows the illustrative representation of these events in the request network and their relationship to the definition of P_{rsv}.

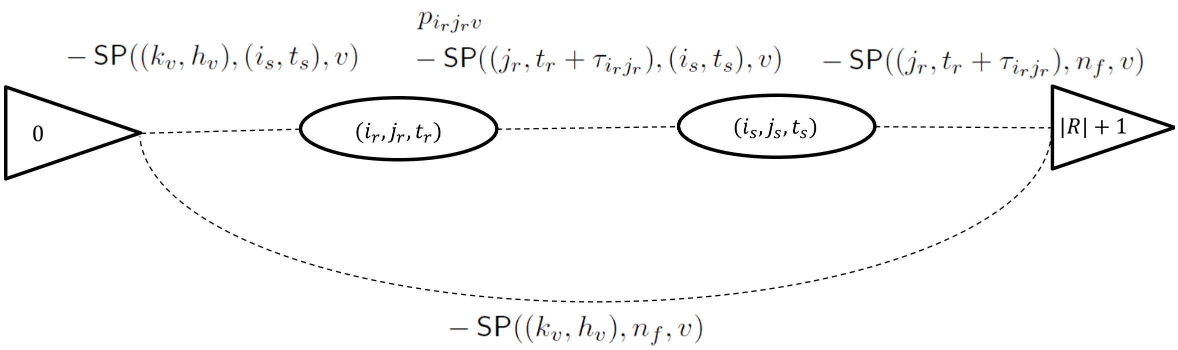


Fig. 5: Redefinition of profit/cost P_{rsv}

The decision variables in the alternative formulation, defined as x_{rsv}, take the value of 1 if vehicle v serve request s after serving request r, and 0 otherwise. Using the presented definitions, the node-demand formulation of the VAP writes as:

$$\max \quad \sum_{v \in V} \sum_{r \in R \cup \{0\}} \sum_{\substack{s \in S \cup \{|R|+1\} \\ s > r}} P_{rsv} x_{rsv} \tag{6}$$

$$\text{s.t.} \quad \sum_{\substack{s \in R \cup \{|R|+1\} \\ s > r}} x_{rsv} - \sum_{\substack{s \in R \cup \{0\} \\ s < r}} x_{srv} = \left\{ \begin{array}{ll} 1 & \text{if } r = 0 \\ -1 & \text{if } r = |R| + 1 \\ 0 & \text{if } r \neq 0 \wedge r \neq |R| + 1 \end{array} \right\}$$

$$\forall r \in R \cup \{0\} \cup \{|R|+1\}, \forall v \in V, \quad (7)$$

$$\sum_{v \in V} \sum_{\substack{s \in R \\ s > r}} x_{rsv} \leq D_r, \quad \forall r \in R, \quad (8)$$

$$x_{srv} = 0 \quad \text{if} \quad A_{i_r j_r v} = 0, \quad \forall r \in R \cup \{|R|+1\}, \forall v \in V, \quad (9)$$

$$x_{rsv} = 0 \quad \text{if} \quad A_{i_r j_r v} = 0, \quad \forall r \in R \cup \{0\}, \forall v \in V, \quad (10)$$

$$x_{rsv} \in \{0,1\}, \quad \forall r \in R \cup \{0\}, \forall s \in R \cup \{|R|+1\}, \forall v \in V. \quad (11)$$

The objective function (6) seeks to maximize the total profit from meeting the requests and returns the same optimal value on the objective function of formulation (1)-(5) from Section 2. Constraints (7) enforce that all vehicles depart from and arrive at the artificial depots as well as ensure the flow conservation of the vehicles on the request nodes. Constraints (8) enforce the number of vehicles that can be used to serve a given request. Constraints (9) and (10) prohibit the movements out of and into r, respectively, when vehicle type v cannot cross the arc where the demand is placed. Finally, constraints (11) impose the domain of decision variables. Note that if it is not possible to go from a node A to another node B in the space-time network, the value of the shortest path between these two nodes in the profits definition is ∞ and, hence, the profit between the requests related to these nodes becomes $-\infty$. We overcome the inconveniences resulting from setting Big-M profits by fixing $x_{rsv} = 0$ whenever $P_{rsv} = -\infty$.

The number of variables and constraints in the proposed model is equal to $|V||R|(|R|-1)/2$ and $|V||R| + |R|$, respectively (without considering the variables that are zeroed out). On the other hand, the number of variables and constraints in formulation (1)-(5) is equal to $|V||T||N|^2$ and $|V||T||N| + |T||N|^2$, respectively. As can be observed, the proposed model does not depend explicitly on the number of periods $|T|$ and terminals $|N|$ on the network, which can be of great advantage for solving some realistic large-scale instances with a moderate quantity of requests $|R|$ and vehicles $|V|$.

Finally, it is worth mentioning that formulation (6)–(11) has some similarities with a specialized formulation of the Full Truckload Vehicle Routing Problem (FTVRP) (Desrosiers et al., 1984, 1988; Arunapuram et al., 2003; Gronalt et al., 2003). However, there is a structural difference that does not enable us to use the solution methods previously developed for the FTVRP in an efficient manner for the VAP. In the FTVRP, it is necessary to serve (visit) all demand for shipping transport (customers), hence feasible and optimal solutions are Hamiltonian paths over directed graphs. Since in the VAP we consider the possibility of partially serving demand, solutions are simple paths in the time-space network. Furthermore, in the alternative formulation proposed in this paper, we are able to show that the graph is acyclic, thus, we can explore this property by using efficient algorithms in our decomposition approach. Finally, to the best of our knowledge, there is no other solution approach using a node-demand network for the VAP.

4 Dantzig-Wolfe decomposition

In this section, the Dantzig-Wolfe decomposition is presented for the new formulation of the VAP. We chose to decompose according to each vehicle since, from a practical point of view, it is best for operations control to create set of decisions according to vehicle groups or individual vehicles, and from a modeling perspective, quality results have been obtained in related network problems when decomposing according to vehicle characteristics (Cruz et al., 2019, 2020). Thus, by taking the demand-satisfying constraints (10) as the coupling constraints, the remaining constraints can be grouped into the sets of solutions $X_v, \forall v \in V$, given by:

$$X_v = \left\{ \begin{array}{lll} x_v \quad | \displaystyle\sum_{\substack{s \in R \cup \{|R|+1\} \\ s > r}} x_{rsv} - \sum_{\substack{s \in R \cup \{0\} \\ s < r}} x_{srv} = \left\{ \begin{array}{ll} 1 & \text{if } r = 0 \\ -1 & \text{if } r = |R|+1 \\ 0 & \text{if } r \neq 0 \wedge r \neq |R|+1 \end{array} \right\}, & & \\ & & \forall r \in R \cup \{0\} \cup \{|R|+1\}, \\ x_{srv} = 0 \quad \text{if} \quad A_{i_r j_r v} = 0, & & \forall r \in R \cup \{|R|+1\}, \\ x_{rsv} = 0 \quad \text{if} \quad A_{i_r j_r v} = 0, & & \forall r \in R \cup \{0\}, \\ x_{rsv} \in \{0, 1\}, & & \forall r \in R \cup \{0\}, s \in R \cup \{|R|+1\}. \end{array} \right\}$$

Hence, the resulting equivalent formulation to (6)–(11) writes as:

$$\begin{aligned} \max \quad & \sum_{v \in V} \sum_{r \in R \cup \{0\}} \sum_{s \in S \cup \{|R|+1\}} P_{rsv} x_{rsv} \\ \text{s.t.} \quad & \sum_{v \in V'} \sum_{\substack{s \in R \\ s > r}} x_{rsv} \leq D_r, \quad \forall r \in R, \\ & x_v \in X_v, \quad \forall v \in V, \end{aligned}$$

Each set X_v is a bounded polyhedron as it is defined exclusively by binary variables. Then, using the discretization approach (Vanderbeck and Wolsey, 2010; Lübbecke and Desrosiers, 2005), we can write any solution $x_v \in X_v$ as an integer combination of the extreme points of X_v as follows:

$$x_v = \sum_{q \in Q_v} \lambda_{vq} \bar{x}_{vq}, \text{ with } \sum_{q \in Q_v} \lambda_{vq} = 1, \lambda_{vq} \in \{0,1\}, \tag{12}$$

where $\bar{x}_{vq}$ denotes the extreme points of X_v, and Q_v is the set of indices of all extreme points. By substituting this representation into (6)-(11), the result is the Master Problem (MP):

$$\max \quad \sum_{v \in V} \sum_{r \in R \cup \{0\}} \sum_{s \in S \cup \{|R|+1\}} P_{rsv} \left(\sum_{q \in Q_v} \lambda_{vq} \bar{x}_{vq} \right) \tag{13}$$

$$\text{s.t.} \quad \sum_{v \in V'} \sum_{\substack{s \in R \\ s > r}} \left(\sum_{q \in Q_v} \lambda_{vq} \bar{x}_{vq} \right) \le D_r, \quad \forall r \in R, \qquad (u_r) \tag{14}$$

$$\sum_{q \in Q_v} \lambda_{vq} = 1, \quad \forall v \in V, \qquad (w_v) \tag{15}$$

$$\lambda_{vq} \in \{0,1\}, \quad \forall v \in V, \forall q \in Q_v, \tag{16}$$

where u_r and w_v represent the dual variables regarding the coupling and convexity constraints, respectively, for the linear programming (LP) relaxation of the MP. Given the huge size of Q_v in realistic problem instances and the fact that most variables are likely to assume the value of zero in an optimal solution, we solve the LP relaxation of the MP using the Column Generation (CG) technique. More specifically, we initialize the LP relaxation of the MP with just a subset of extreme points of X_v (in our implementation, we use the empty path from node 0 to node $|R|+1$ in the request network, for each vehicle v), resulting in what is called the Restricted Master Problem (RMP). Hence, the RMP starts with a relatively small number of columns (variables) and we may generate new columns iteratively using the dual solutions and the following pricing subproblems:

$$Z_{sp(v)} = \quad \max \quad \sum_{r \in R \cup \{0\}} \sum_{s \in S \cup \{|R|+1\}} (P_{rsv} - u_r) x_{rsv} \tag{17}$$

$$\text{s.t.} \quad \sum_{\substack{s \in R \cup \{|R|+1\} \\ s > r}} x_{rsv} - \sum_{\substack{s \in R \cup \{0\} \\ s < r}} x_{srv} = \left\{ \begin{array}{ll} 1 & \text{if } r = 0 \\ -1 & \text{if } r = |R|+1 \\ 0 & \text{if } r \neq 0 \wedge r \neq |R|+1 \end{array} \right\}$$

$$\forall r \in R \cup \{0\} \cup \{|R|+1\} \tag{18}$$

$$x_{srv} = 0 \quad \text{if} \quad A_{i_r j_r v} = 0, \quad \forall r \in R \cup \{|R|+1\}, \tag{19}$$

$$x_{rsv} = 0 \quad \text{if} \quad A_{i_r j_r v} = 0, \quad \forall r \in R \cup \{0\}, \tag{20}$$

$$x_{rsv} \in \mathbb{R}_+, \quad \forall r \in R \cup \{0\}, s \in R \cup \{|R|+1\}, \tag{21}$$

for each $v \in V$. Hence, the reduced cost of a variable λ_{vq} is given by $(Z_{sp(v)} - w_v)$. In this particular case of the addressed problem, the optimal solution obtained at the end of the CG procedure has the same objective value as an optimal solution of the linear programming (LP) relaxation of the original model (6)–(11), as the pricing subproblems (17)-(21) have the integrality property. Nevertheless, the reformulation is still computationally attractive as the size of the MP is significantly reduced in comparison to the size of model (6)–(11) and the pricing subproblems can be efficiently solved using specialized shortest path algorithms on directed acyclic graphs, as discussed in the next section.

5 Branch-and-price method

In this section, we propose a BP method to solve the MP (13)–(16), which consists in using the CG technique within each node of branch-and-bound tree (Lübbecke and Desrosiers, 2005). Our implementation is based on efficient approaches to enhance the overall performance of the method, such as a stabilized interior-point column generation technique (Subsection 5.1); a specialized algorithm for solving the pricing subproblems (Subsection 5.2); and a hierarchical branching strategy that impose constraints in the master problem (Subsection 5.3).

5.1 Interior-point column generation technique

At each node of the BP tree, we need to solve the LP relaxation associated to this node, which is given by the MP (13)–(16) with possibly additional branching constraints. To avoid the well-known pathological behaviors resulting from using extreme dual solutions provided by the simplex method when solving the RMP (Vanderbeck, 2005), and in accordance with the promising results obtained by Cruz et al. (2019, 2020), we use the Primal-Dual Column Generation Method (PDCGM) (Gondzio et al., 2013, 2016) for solving the LP relaxation of the node. The PDCGM is a stabilized column generation technique that relies on a primal-dual interior point method to solve each RMP. Hence, the obtained dual solutions are well-centered points in the dual feasible set and promote a better overall performance of the method, particularly for earlier CG approaches developed for the VAP (Cruz et al., 2019, 2020) and related problems (Munari and Gondzio, 2013, 2015; Alvarez and Munari, 2017).

After the CG method finishes, we check if the optimal solution provided is fractional and, if it is, we run a simple model-based heuristic in an attempt to quickly obtain an improved incumbent solution. This heuristic consists of (1) imposing integrality on the variables in the RMP solved in the last iteration of the CG method; and (2) solving the resulting integer programming problem by a general-purpose solver using a short time limit.

5.2 Pricing subproblem

Each subproblem (17)-(21) is a maximum cost flow problem over a directed acyclic graph in which we have to flow one vehicle from the starting depot (node 0) to the end depot (node $|R|+1$), hence it is a longest path problem (as we are maximizing profits) over a directed acyclic network. Since the graph for each vehicle is already topologically sorted, we can use a linear update for optimilaty conditions as described in Algorithm 1. The values $y_{r,s}$ correspond to the updated arc profits through the dual values from the coupling constraints of the master problem and whenever $A_{i_r j_r v} = 0$, we set $y_{r,s} = -\inf$.

Algorithm 1 DAG's algorithm

procedure DAG$(0, |R|+1)$
    Initialize $d[k] = -\infty$ and $p[k] = 0, \forall k \in R \cup \{|R|+1\}$;
    $d[0] = 0$;
    for $r \leftarrow 0 : R$ **do**
        for $s \leftarrow r+1 : |R|+1$ **do**
            let $y_{rs} = P_{rsv} - u_r$ be the profit of arc (r, s) for vehicle $v \in V$
            if $d[s] > d[r] + y_{rs}$ **then** $d[s] \leftarrow d[r] + y_{rs}$
                $p[s] \leftarrow r$
            end if
        end for
    end for
end procedure

5.3 Branching strategy

The branching strategy used in the proposed BP method consists of a hierarchical rule based on the total served demand and on the arc flow in a fractional solution. More specifically, given an optimal solution $\bar{\lambda}$ of the RMP solved in the last iteration of the CG technique, we verify if the total demand served by all vehicles for a given $r \in R$ is fractional, i.e.

$$\sum_{v \in V} \sum_{\substack{s \in R \\ s > r}} \sum_{q \in Q_v} \bar{x}_{qrsv} \bar{\lambda}_{vq} \in \mathbb{R}_+ \backslash \{\mathbb{Z}_+\}. \tag{22}$$

Among all indices that satisfy (22), we select the index r with the earliest departing time and then generate two child nodes, one with the first and the other with the second of the following additional branching constraints:

$$\sum_{v \in V} \sum_{\substack{s \in R \\ s > r}} \sum_{q \in Q_v} \bar{x}_{qrsv} \lambda_{vq} \geq \left\lceil \sum_{v \in V} \sum_{\substack{s \in R \\ s > r}} \sum_{q \in Q_v} \bar{x}_{qrsv} \bar{\lambda}_{vq} \right\rceil, \tag{23}$$

$$\sum_{v \in V} \sum_{\substack{s \in R \\ s > r}} \sum_{q \in Q_v} \bar{x}_{qrsv} \lambda_{vq} \leq \left\lfloor \sum_{v \in V} \sum_{\substack{s \in R \\ s > r}} \sum_{q \in Q_v} \bar{x}_{qrsv} \bar{\lambda}_{vq} \right\rfloor, \tag{24}$$

where the unary operators $\lceil\,\rceil$ and $\lfloor\,\rfloor$ represent the ceiling and floor functions for rounding fractional numbers. Since they are imposed in the LP relaxation of the MP, these branching constraints do not damage the structure of the pricing subproblems. To account for the dual solution in the pricing subproblems in a given node k, we only need to include the following summation in the objective function of vth subproblem, for each $v \in V$:

$$-\sum_{r \in B^k} \sum_{s \in S \cup \{|R|+1\}} \bar{b}_r x_{rsv},$$

where B^k is the set of all indices $r \in R$ such that a branching constraint of type (23) or (24) is imposed in node k (coming from all its ancestral nodes in the tree), and $\bar{b}_r$ is the dual solution associated to the rth branching constraint of these types.

This branching rule alone does not guarantee an integer solution as the summation in (22) may be integer even for a solution $\bar{\lambda}$ with fractional components. Hence, we resort to an additional rule when the summation in (22) is integer for all $r \in R$. This rule is based on the values of variables x_{rsv}, which are computed from solution $\bar{\lambda}$ using the summation in (12). Requiring integrality of x_{rsv} is equivalent to requiring integrality of the MP variables λ, as we rely on discretization (Lübbecke and Desrosiers, 2005). Hence, given a tuple of indices (r, s, v) such that x_{rsv} is fractional, we branch by enforcing $x_{rsv} = 0$ in one child node, and $x_{rsv} = 1$ in the other child node. The selection of

(r, s, v) is done giving priority to variables with earlier departing times, choosing randomly among different vehicles to break ties. To avoid damaging the structure of the subproblems, we impose the new bounds of x_{rsv} at the MP level, by inserting one of the following constraints:

$$\sum_{q \in Q^v} \bar{x}_{qrsv} \lambda_{vq} \leq 0 \quad \text{or} \quad \sum_{q \in Q^v} \bar{x}_{qrsv} \lambda_{vq} \geq 1, \tag{25}$$

which correspond to imposing $x_{rsv} \leq 0$ and $x_{rsv} \geq 1$, respectively. Then, similarly to the previous branching rule, we only need to modify the objective function of the pricing subproblems to account for the dual solution related to these new branching constraints. More specifically, in a given node k of the tree, we include the following summation in the objective function of the vth subproblem:

$$- \sum_{(r,s,v) \in U^k \cup L^k} w_{rsv} x_{rsv}$$

where w_{rsv} is the dual solution associated to constraints of type (25), and U^k and L^k are the set of index tuples (r, s, v) related to the upper and lower bound constraints in (25) that are imposed in node k. Note that the values w_{rsv} can be easily incorporated into the subproblems as additional costs at the arcs of the request network.

In summary, we have a hierarchical branching scheme in which we first verify if there is at least one index r that satisfy (22) and, if no such index exists, we verify if there is at least one tuple (r, s, v) such that x_{rsv} is fractional. Finally, it is worth mentioning that we use the best-first search rule to decide the next node to be processed along the search tree, i.e, the method selects the node with the worst dual bound (maximum dual bound in maximization problems).

6 Computational Experiments

In this section, we present the results of computational experiments with the proposed approaches as well as the state-of-the-art approaches in literature for solving the VAP, using small and large-scale realistic-sized instances collected from Vasco and Morabito (2016) and Cruz et al. (2020). First, we present the results of solving the two considered compact formulations, namely the node-demand formulation proposed in Section 3 and the arc-demand formulation presented in Section 2.2. Then, we show the results of the experiments with the BP method proposed in Section 5. All methods were implemented in C++ and use the Concert Library of the IBM CPLEX Optimization

Studio v.12.8.1. Additionally, for the BP method we use the PDCGM library (Gondzio et al., 2016) as the CG solver. All experiments reported in this section were run on a PC with CPU Intel Core i7-4790S 3.20GHz and 16 GB of RAM. A symbol "*" indicates that CPLEX could not solve or even mount the model due to lack of computer memory. A time limit of 7200 seconds was imposed on each run.

We use two set of instances to develop the performance analysis of the above-mentioned solution methods. The first set comprises 30 instances from the work of Vasco and Morabito (2016) which share the same characteristics, namely, 30 terminals, 36 periods, 300 requests for demand and 130 vehicles. These characteristics are used to name the instances in the following format: *terminals-periods-requests-vehicles*-p*Id*, where *Id* is a unique identifier of the instance. The second set comprises 200 instances from the work of Cruz et al. (2019) with different number of terminals, periods, requests and vehicles. Therefore, in naming each instance we use the same notation of the previous set, however, omitting the last digit denoting the identifier of the instance (i.e., *terminals-periods-requests-vehicles*). These instances will be referred to according to the range of the number of terminals: $[10, 19]$, $[20, 29]$, $[30, 39]$, $[40, 49]$ and $[50, 59]$. For each of these terminal ranges, the number of vehicles varies in the following way: for $[10, 19]$ it increases from 20 to 50 vehicles; for $[20, 29]$, from 130 to 150; for $[30, 39]$, from 200 to 250; for $[40, 49]$, from 130 to 170; and for $[50, 59]$, from 100 to 250. The numbers of periods are 10, 30, 36 and 36, while the numbers of are 50, 200, 300, 500 and 700 for each terminal range, respectively. The detailed results of solving these instances with the methods are presented in the Appendix A.

6.1 Comparison of arc-demand formulation and node-demand formulation

In this section, we compare the performance of the general-purpose ILP solver of CPLEX in solving the compact formulations of Sections 2.2 and 3. Table 4 shows the computational times for solving the LP relaxation and the ILP model of the arc-demand formulation (1)–(5) and the node-demand formulation (6)–(11) for the set of instances [20,29], [30,39] of Cruz et al. (2020) and the 30 large-scale instances of Vasco and Morabito (2016). Table 4 includes only the computational times of the LP relaxation and the corresponding ILP model for both formulations of instances [20,29], [30,39] from Cruz et al. (2020). Computational times for the arc-demand formulation of instances from Vasco and Morabito (2016) are not included as CPLEX could not mount the corresponding models. It is worth mentioning that the respective ILP arc-demand models of these instances have

26,292,240 variables and 349,164 constraints. Objective values for all instances are detailed in Appendix A. Results for instances [10,19] are omitted as all computational times do not exceed 1 second. In the case of instances [40,49], CPLEX only solved 2 and 11 instances to optimally for the arc-demand and node-demand formulations, respectively, which are also detailed in Appendix A.

Tab. 4: Computational times for solving instances from Vasco and Morabito (2016) and instances with terminals in ranges $[20, 29]$ and $[30, 39]$ using the arc-demand and node-demand formulations.

Instance	*Arc-demand*		*Node-demand*		*Instance*	*Arc-demand*		*Node-demand*		*Instance*	*Node-demand*	
	LP	*ILP*	*LP*	*ILP*		*LP*	*ILP*	*LP*	*ILP*		*LP*	*ILP*
20-20-100-200	3.49	16.23	1.56	2.64	30-30-200-300	100.44	392.92	7.26	9.54	53-36-130-300-p1	28.81	4.15
20-20-130-200	4.33	19.13	1.26	2.26	30-30-230-300	104.13	435.42	6.84	9.18	53-36-130-300-p2	5.05	3.19
20-20-150-200	5.45	22.10	1.73	2.97	30-30-250-300	153.42	750.87	13.72	16.14	53-36-130-300-p3	4.84	2.34
21-20-100-200	5.36	19.09	1.58	2.48	31-30-200-300	98.37	436.45	7.82	10.46	53-36-130-300-p4	29.11	4.55
21-20-130-200	5.01	21.65	1.30	2.27	31-30-230-300	135.00	716.02	10.75	13.15	53-36-130-300-p5	2.75	2.01
21-20-150-200	5.51	23.34	1.75	2.96	31-30-250-300	139.71	685.50	12.64	14.95	53-36-130-300-p6	12.30	5.79
22-20-100-200	5.82	23.76	1.46	2.56	32-30-200-300	132.54	497.74	7.67	9.83	53-36-130-300-p7	3.42	2.81
22-20-130-200	5.71	25.56	1.88	2.89	32-30-230-300	130.23	621.84	9.94	11.45	53-36-130-300-p8	14.68	4.01
22-20-150-200	7.14	34.33	2.68	4.20	32-30-250-300	154.61	730.32	14.49	16.02	53-36-130-300-p9	3.85	3.51
23-20-100-200	4.65	19.48	1.11	1.84	33-30-200-300	110.64	456.08	9.10	10.48	53-36-130-300-p10	2.64	3.75
23-20-130-200	6.44	27.30	1.62	2.90	33-30-230-300	268.12	933.43	12.97	14.46	53-36-130-300-p11	3.39	2.09
23-20-150-200	8.00	31.67	2.15	3.53	33-30-250-300	101.29	417.62	7.85	9.35	53-36-130-300-p12	15.54	4.50
24-20-100-200	5.30	24.04	1.33	2.29	34-30-200-300	131.44	639.94	10.56	11.93	53-36-130-300-p13	6.87	3.37
24-20-130-200	11.08	42.33	2.56	3.51	34-30-230-300	164.23	903.13	14.56	16.45	53-36-130-300-p14	4.34	2.90
24-20-150-200	12.68	52.91	2.38	3.87	34-30-250-300	*	*	12.80	14.17	53-36-130-300-p15	4.04	4.48
25-20-100-200	5.20	23.05	1.35	2.53	35-30-200-300	115.12	536.07	7.17	9.46	53-36-130-300-p16	5.05	3.30
25-20-130-200	8.86	38.92	2.30	4.29	35-30-230-300	*	*	10.95	12.96	53-36-130-300-p17	2.24	2.20
25-20-150-200	9.41	37.72	2.08	2.97	35-30-250-300	*	*	10.39	12.87	53-36-130-300-p18	5.99	3.22
26-20-100-200	5.49	22.13	1.37	2.50	36-30-200-300	116.12	615.03	8.72	11.07	53-36-130-300-p19	2.23	2.00
26-20-130-200	7.66	33.17	1.59	2.93	36-30-230-300	*	*	12.44	15.01	53-36-130-300-p20	5.51	2.78
26-20-150-200	19.33	67.78	2.39	4.36	36-30-250-300	*	*	18.74	20.48	53-36-130-300-p21	16.13	7.36
27-20-100-200	7.05	32.24	1.66	2.38	37-30-200-300	144.08	826.57	9.75	11.46	53-36-130-300-p22	16.86	5.56
27-20-130-200	11.71	55.14	2.34	3.48	37-30-230-300	*	*	9.46	11.01	53-36-130-300-p23	13.97	4.41
27-20-150-200	9.95	44.40	1.81	2.89	37-30-250-300	*	*	10.59	12.85	53-36-130-300-p24	2.60	2.31
28-20-100-200	10.32	42.01	1.89	2.89	38-30-200-300	*	*	7.21	9.07	53-36-130-300-p25	6.48	3.43
28-20-130-200	8.64	36.32	1.44	2.65	38-30-230-300	*	*	11.67	13.96	53-36-130-300-p26	3.50	2.63
28-20-150-200	11.36	57.47	2.41	4.07	38-30-250-300	*	*	12.50	14.76	53-36-130-300-p27	2.69	2.67
29-20-100-200	7.09	31.58	1.28	2.15	39-30-200-300	*	*	8.57	11.01	53-36-130-300-p28	3.82	2.79
29-20-130-200	11.48	49.86	1.88	2.77	39-30-230-300	*	*	13.52	16.77	53-36-130-300-p29	2.21	2.29
29-20-150-200	10.10	44.10	1.88	3.22	39-30-250-300	*	*	10.49	12.32	53-36-130-300-p30	3.63	2.69

From Table 4, we observe that CPLEX in solving the model of the node-demand formulation outperformed that of the arc-demand formulation as it solved faster all instances in the sets [20,29] and [30,39]. The solution of the LP relaxation of arc-demand model required up to 20 and 269 seconds for the $[20, 29]$ and $[30, 39]$ instances, respectively, while that of the node-demand model required up to 3 and 19 seconds for the $[20, 29]$ and $[30, 39]$ instances, respectively. In the case of the ILP model, the difference in computational times is more pronounced. It took up to 58 and 934 seconds for the arc-demand model in both set of instances, while it took up to 5 and 21 seconds for node-demand model across both set of instances. Note that CPLEX could not mount the models of 13 instances in the set $[30, 39]$. In the case of the larger instances from Vasco and Morabito

(2016), by using the proposed node-demand model CPLEX was able to solve the LP relaxation of all instances in less than 8 seconds and the ILP models in less than 30 seconds. It is worth mentioning, however, that many of the 300 requests describing these instances overlap on the same subset of arcs; hence, all of the instances have a number of demand arcs in the range 50 to 70.

Table 5 presents the average computational times of solving the LP ($Av\,LP$) and ILP ($Av\,ILP$) models, and the number of instances solved (*Count*) for the arc-demand and node-demand formulations. Instances in the range [50,59] are not included as neither the arc-demand nor the node-demand models could be mounted by CPLEX. From this table we observe that the computational times resulting from using the node-demand formulation are better in each set of instances. Additionally, for the set [30,39], [40,49] and the ones from Vasco and Morabito (2016), we were able to solve a larger proportion of instances using the proposed node-demand formulation.

Tab. 5: Average computational times and count of instances solved to optimality by CPLEX using the arc-demand and node-demand formulations.

Instance	*Arc-demand*			*Node-demand*		
	$Av\,LP$	$Av\,ILP$	*Count*	$Av\,LP$	$Av\,ILP$	*Count*
[10,19]	0.12	0.31	50	0.04	0.01	50
[20,29]	7.99	33.96	30	2.97	1.80	30
[30,39]	135.26	623.23	17	12.76	10.70	30
[40,49]	321.51	1891.44	2	27.98	24.44	11
Vasco (2016)	*	*	0	7.82	3.44	30

6.2 Comparison of the BP methods based on the reformulations the arc-demand and node-demand formulations

In this section, we analyze the performance of the BP method proposed in Section 5, based on the reformulation of the node-demand model. Tables 6, 7 and 8 shows the results of the BP method proposed in this paper and the BP method proposed in Cruz et al. (2020), which is based on the reformulation of the arc-demand model. Columns in all these tables refer to: T_{root} is the time taken to solve the root node only and T_{BP} is the time taken by the BP. The detailed results for all sets of instances are shown in Appendix A.

Table 6 shows the results of both BP methods in instances from the work of Vasco and Morabito (2016). We observe that even though both BP were able to process and solve the root node of all instances, the node-demand BP method clearly outperforms the arc-demand BP method in terms of computational times. All instances were solved in less than 45 seconds by the arc-based BP method, while it took less than 1 second for the BP method based on the node-demand model. The six instances with boldfaced names are the only ones within this set that have positive integrality gaps, i.e., the optimal value obtained at the root node is strictly smaller than the optimal value of the instance. For these instances, the node-demand BP method took less than 3 seconds, while the arc-demand BP method took more than 46 seconds.

Tab. 6: Results of the BP methods based on the arc-demand and node-demand formulations for instances from Vasco and Morabito (2016).

Instance	*Arc-demand*		*Node-demand*		*Instance*	*Arc-demand*		*Node-demand*	
	T_{Root}	T_{BP}	T_{Root}	T_{BP}		T_{Root}	T_{BP}	T_{Root}	T_{BP}
53-36-130-300-p1	33.0	46.9	0.5	0.5	53-36-130-300-p16	36.0	36.0	0.2	0.2
53-36-130-300-p2	29.1	29.1	0.2	0.2	53-36-130-300-p17	33.5	33.5	0.2	0.2
53-36-130-300-p3	31.0	409.1	0.4	1.2	**53-36-130-300-p18**	36.6	83.0	0.5	0.8
53-36-130-300-p4	36.8	36.8	0.3	0.3	53-36-130-300-p19	37.2	37.2	0.2	0.2
53-36-130-300-p5	29.1	29.1	0.2	0.2	53-36-130-300-p20	36.3	36.3	0.2	0.2
53-36-130-300-p6	33.0	33.0	0.3	0.3	53-36-130-300-p21	44.8	44.8	0.3	0.3
53-36-130-300-p7	31.5	31.5	0.3	0.3	53-36-130-300-p22	37.6	37.6	0.4	0.4
53-36-130-300-p8	29.2	127.9	0.2	1.5	53-36-130-300-p23	34.4	34.4	0.3	0.3
53-36-130-300-p9	36.7	36.7	0.3	0.3	53-36-130-300-p24	27.3	27.3	0.2	0.2
53-36-130-300-p10	25.2	25.2	0.2	0.2	53-36-130-300-p25	33.9	33.9	0.3	0.3
53-36-130-300-p11	36.0	46.5	0.2	2.0	53-36-130-300-p26	30.2	30.2	0.2	0.2
53-36-130-300-p12	34.7	34.7	0.3	0.3	53-36-130-300-p27	27.1	27.1	0.2	0.2
53-36-130-300-p13	37.5	80.6	0.1	0.9	53-36-130-300-p28	37.9	37.9	0.2	0.2
53-36-130-300-p14	31.8	31.8	0.2	0.2	53-36-130-300-p29	30.2	30.2	0.2	0.2
53-36-130-300-p15	38.6	38.6	0.2	0.2	53-36-130-300-p30	33.7	33.7	0.2	0.2

Table 7 shows the results of the BP methods in instances with terminals in ranges $[20, 29]$ and $[30, 39]$. First, we compare the performance of the methods in solving the root node. In the set $[20, 29]$, the node-demand BP method took less than 1 second, while the arc-demand BP method took less than 4 seconds. In the set $[30, 39]$, the difference is slightly more pronounced as it still took less than 1 second for the node-demand BP method and more than 9 seconds for the arc-demand BP method for all instances. Regarding the performance of the BP methods to solve the instances to proven optimality, the node-demand BP method solved all instances in the set $[20, 29]$ in less than 37 seconds, whereas the arc-demand BP method took 73 seconds (both maximum computational times occurred in instance 29-20-130-200). In the set $[30, 39]$, it took less than 383 seconds for the node-demand BP method and 1540 seconds for the arc-demand BP method (both maximum computational times occurred in instance 31-30-230-300). It is worth highlighting that in all instances in this table, the proposed BP method was faster than the arc-demand BP method of Cruz et al. (2020).

Finally, Table 8 shows the results of the BP methods in instances with terminals in ranges $[40, 49]$ and $[50, 59]$. Instances with boldfaced times (BP) correspond to instances that were not solved to

Tab. 7: Results of the BP methods based on the arc-demand and node-demand formulations for instances with terminals in ranges [20, 29] and [30, 39].

Instance	*Arc-demand*		*Node-demand*		*Instance*	*Arc-demand*		*Node-demand*	
	T_{Root}	T_{BP}	T_{Root}	T_{BP}		T_{Root}	T_{BP}	T_{Root}	T_{BP}
20-20-100-200	1.48	5.36	0.09	1.15	30-30-200-300	8.23	16.41	0.36	3.96
20-20-130-200	1.51	7.51	0.11	4.26	30-30-230-300	9.41	21.33	0.46	6.41
20-20-150-200	1.70	2.63	0.12	1.10	30-30-250-300	10.03	108.11	0.51	71.58
21-20-100-200	1.75	3.09	0.08	0.98	31-30-200-300	8.46	11.79	0.30	2.08
21-20-130-200	2.05	3.51	0.13	1.02	31-30-230-300	9.68	1539.31	0.41	382.25
21-20-150-200	1.88	3.56	0.12	1.06	31-30-250-300	10.34	13.13	0.49	4.01
22-20-100-200	1.96	5.71	0.10	1.36	32-30-200-300	9.11	39.08	0.35	9.51
22-20-130-200	1.99	4.04	0.11	0.94	32-30-230-300	10.83	18.04	0.40	2.51
22-20-150-200	2.12	5.47	0.12	1.40	32-30-250-300	17.02	394.39	0.57	176.37
23-20-100-200	1.90	2.99	0.09	0.75	33-30-200-300	9.86	254.35	0.35	21.98
23-20-130-200	1.94	3.02	0.09	1.17	33-30-230-300	11.13	32.99	0.40	6.21
23-20-150-200	2.20	5.03	0.12	1.93	33-30-250-300	11.59	65.80	0.41	16.85
24-20-100-200	2.37	6.28	0.08	1.32	34-30-200-300	14.44	63.03	0.45	14.86
24-20-130-200	3.51	13.02	0.14	5.79	34-30-230-300	12.19	56.14	0.45	11.04
24-20-150-200	3.26	9.43	0.15	2.36	34-30-250-300	12.77	21.54	0.44	4.00
25-20-100-200	2.33	5.93	0.07	1.09	35-30-200-300	11.06	170.60	0.34	34.44
25-20-130-200	2.64	4.07	0.12	0.99	35-30-230-300	13.41	45.84	0.40	6.39
25-20-150-200	2.91	6.05	0.14	1.80	35-30-250-300	13.90	61.65	0.44	13.50
26-20-100-200	2.64	4.16	0.08	0.73	36-30-200-300	13.64	29.92	0.39	4.82
26-20-130-200	2.76	3.93	0.10	0.70	36-30-230-300	14.57	28.51	0.44	6.79
26-20-150-200	3.94	21.92	0.14	5.35	36-30-250-300	14.50	21.88	0.47	2.76
27-20-100-200	3.31	4.54	0.08	0.76	37-30-200-300	13.07	16.96	0.29	6.33
27-20-130-200	3.95	29.05	0.14	6.51	37-30-230-300	16.65	1250.38	0.39	273.13
27-20-150-200	3.29	6.69	0.12	1.94	37-30-250-300	16.07	48.15	0.41	7.76
28-20-100-200	3.23	42.24	0.08	8.75	38-30-200-300	14.08	465.01	0.34	75.33
28-20-130-200	3.32	4.63	0.10	1.19	38-30-230-300	15.72	25.18	0.44	3.72
28-20-150-200	3.73	5.49	0.12	1.15	38-30-250-300	18.31	26.90	0.48	4.05
29-20-100-200	3.69	7.48	0.08	1.56	39-30-200-300	18.71	69.13	0.39	9.22
29-20-130-200	3.60	72.53	0.12	36.02	39-30-230-300	17.87	238.48	0.43	44.26
29-20-150-200	3.91	6.73	0.11	0.76	39-30-250-300	17.57	60.95	0.46	9.59

proven optimality within the time limit by the BP method of Cruz et al. (2020). These instances are 50-36-130-700 and 50-36-250-700, with upper bounds (lower bounds) of 57051.5 (57048.0) and 111448.0 (111444.0), respectively; and both with final relative gaps of 0.0001%. Apart from these two, all remaining instances reached optimality within the time limit. In the set [40, 49], for solving the root node only, the node-based BP method took less than 1 second while the arc-demand BP method took more than 15 seconds, considering all instances. In the set [50, 59], it took less than 3 seconds for the node-demand BP method while it took more than 23 seconds for the arc-demand BP method for all instances. Regarding the integer optimal solutions, the node-demand BP method clearly outperformed the arc-demand BP. In the set, [40, 49] it took less 70 seconds for the method based on the node-demand model and less than 394 seconds for the method based on the arc-demand model. In the set [50, 59], apart from the instances not solved by the arc-demand BP method, the node-demand BP method took less than 778 seconds, while the arc-demand BP

Tab. 8: Results of the BP methods based on the arc-demand and node-based formulations for instances with terminals in ranges $[40, 49]$ and $[50, 59]$.

Instance	*Arc-demand*		*Node-demand*		*Instance*	*Arc-demand*		*Node-demand*		*Instance*	*Arc-demand*		*Node-demand*	
	T_{Root}	T_{BP}	T_{Root}	T_{BP}		T_{Root}	T_{BP}	T_{Root}	T_{BP}		T_{Root}	T_{BP}	T_{Root}	T_{BP}
40-36-130-500	15.23	47.90	0.46	11.66	50-36-100-700	23.56	34.71	0.69	7.61	55-36-100-700	29.75	100.46	0.68	16.99
40-36-150-500	17.29	32.65	0.62	8.89	50-36-130-700	29.24	**7200.00**	1.02	5822.36	55-36-130-700	47.77	73.97	1.02	12.71
40-36-170-500	18.22	52.27	0.69	11.44	50-36-150-700	31.62	68.35	1.19	20.72	55-36-150-700	40.81	49.44	1.16	14.83
41-36-130-500	18.11	113.69	0.48	27.03	50-36-180-700	34.10	627.10	1.39	245.91	55-36-180-700	43.37	109.13	1.37	27.87
41-36-150-500	19.53	24.16	0.62	5.85	50-36-200-700	34.85	37.93	1.50	10.51	55-36-200-700	45.23	120.55	1.52	30.83
41-36-170-500	20.81	25.21	0.70	4.64	50-36-250-700	40.64	**7200.00**	1.91	761.11	55-36-250-700	55.72	108.08	2.15	34.86
42-36-130-500	19.18	42.48	0.47	6.05	51-36-100-700	24.03	99.31	0.59	25.22	56-36-100-700	35.63	54.07	0.69	5.23
42-36-150-500	22.40	26.89	0.78	7.63	51-36-130-700	30.62	42.11	1.02	9.77	56-36-130-700	52.44	57.63	1.03	13.00
42-36-170-500	21.62	80.60	0.67	62.20	51-36-150-700	30.51	171.96	1.16	56.61	56-36-150-700	41.97	98.02	1.17	20.91
43-36-130-500	20.24	25.25	0.46	4.94	51-36-180-700	37.18	1798.33	1.39	777.42	56-36-180-700	48.28	244.86	1.37	73.35
43-36-150-500	21.36	138.62	0.61	30.75	51-36-200-700	38.92	55.16	1.72	10.71	56-36-200-700	51.13	65.10	1.32	14.93
43-36-170-500	21.77	65.34	0.68	15.47	51-36-250-700	49.22	772.57	2.18	268.88	56-36-250-700	54.70	1942.71	2.12	546.20
44-36-130-500	20.28	79.92	0.47	16.84	52-36-100-700	25.76	38.97	0.68	7.48	57-36-100-700	33.38	49.92	0.68	7.60
44-36-150-500	22.88	39.81	0.61	7.70	52-36-130-700	33.24	63.88	1.02	11.91	57-36-130-700	37.13	153.09	0.87	30.21
44-36-170-500	25.29	128.38	0.68	26.54	52-36-150-700	28.20	46.27	0.89	7.81	57-36-150-700	40.73	80.27	1.00	13.53
45-36-130-500	21.41	38.40	0.46	7.64	52-36-180-700	37.01	208.32	1.22	73.72	57-36-180-700	47.38	62.52	1.54	13.76
45-36-150-500	22.82	69.02	0.53	13.67	52-36-200-700	43.32	91.11	1.53	18.60	57-36-200-700	52.76	57.93	1.72	10.89
45-36-170-500	25.23	35.68	0.60	6.51	52-36-250-700	52.17	224.81	2.19	9.48	57-36-250-700	62.00	63.98	2.12	19.03
46-36-130-500	26.07	393.58	0.61	69.61	53-36-100-700	26.39	150.27	0.58	58.73	58-36-100-700	48.58	99.66	1.19	27.76
46-36-150-500	24.76	232.55	0.61	62.85	53-36-130-700	31.03	51.06	0.88	12.69	58-36-130-700	31.85	49.34	0.58	9.71
46-36-170-500	28.27	37.83	0.67	6.55	53-36-150-700	38.67	55.33	1.15	11.14	58-36-150-700	53.58	51.38	1.50	11.63
47-36-130-500	34.85	69.29	0.69	11.31	53-36-180-700	38.71	1023.86	1.39	314.93	58-36-180-700	42.30	164.57	0.98	34.26
47-36-150-500	26.57	76.43	0.60	42.31	53-36-220-700	42.52	105.90	1.44	43.48	58-36-200-700	62.03	80.03	2.12	30.48
47-36-170-500	30.42	78.83	0.69	13.71	53-36-250-700	49.47	65.19	1.90	18.85	58-36-250-700	46.64	2271.38	1.14	632.19
48-36-130-500	28.28	91.27	0.59	15.70	54-36-100-700	29.48	137.10	0.68	31.67	59-36-100-700	46.65	511.11	0.78	99.67
48-36-150-500	33.14	118.50	0.72	20.66	54-36-130-700	33.05	45.46	1.01	9.91	59-36-130-700	50.34	1147.60	1.28	201.27
48-36-170-500	29.44	163.78	0.59	26.56	54-36-150-700	38.63	81.05	1.16	24.17	59-36-150-700	60.56	79.93	1.12	14.84
49-36-130-500	28.81	37.78	0.52	5.06	54-36-180-700	40.40	107.63	1.56	35.34	59-36-180-700	52.15	256.82	1.36	56.59
49-36-150-500	40.63	121.98	0.70	19.01	54-36-200-700	41.92	45.06	1.31	14.89	59-36-200-700	55.35	73.87	1.31	14.88
49-36-170-500	34.51	72.37	0.69	14.07	54-36-250-700	57.90	1418.64	2.14	469.17	59-36-250-700	63.43	82.01	1.86	24.60

method took less than 2272 seconds. Note that both of the above mentioned instances that could not be solved by the arc-demand BP method, were indeed solved by the node-demand BP method in 2594.82 and 761.11 seconds. As observed in the results presented in the previous tables, the proposed BP method was also faster than the BP method of Cruz et al. (2020) in all instances in the sets $[40, 49]$ and $[50, 59]$.

Table 9 presents the average computational times of solving the root node only ($Av\,T_{Root}$), the average computational time of solving the integer problem ($Av\,T_{BP}$) and the number of instances solved ($Count$) for the arc-demand and node-demand formulations. From this table we observe that the computational times resulting from using the node-demand formulation are better in each set of instances. Additionally, we were able to solve two unsolved instances of the work of Cruz et al. (2020).

Tab. 9: Average computational times and count of instances solved to optimality using the BP methods based on the arc-demand and node-demand formulations.

Instance	*Arc-demand*			*Node-demand*		
	$Av\,T_{Root}$	$Av\,T_{BP}$	*Count*	$Av\,T_{Root}$	$Av\,T_{BP}$	*Count*
[10,14]	0.12	0.38	15	0.01	0.09	15
[15,19]	0.21	0.54	15	0.01	0.04	15
[20,29]	2.70	10.20	30	0.11	3.20	30
[30,39]	13.14	173.83	30	0.42	41.19	30
[40,49]	24.65	85.35	30	0.61	19.43	30
[50,54]	36.41	496.03	28	1.28	198.75	30
[55,59]	47.79	278.65	30	1.29	68.82	30
(Vasco and Morabito, 2016)	33.67	53.35	30	0.27	0.43	30

In summary, from the results shown in this section regarding the addressed compact models and BP methods, it can be observed that the proposed node-demand representation yields superior results in computational efficiency for realistic-sized instances using both the compact model in a general-purpose ILP solver (CPLEX) and the tailored exact solution method based on column generation. By using CPLEX with the node-demand formulation, we were able to solve all instances of Vasco and Morabito (2016), while these same instances could not be even mounted in the case of the arc-demand formulation. Furthermore, we were able to solve a larger proportion of instances from the 200 instances of Cruz et al. (2020) using CPLEX as well. Regarding the BP method, we obtained better results in both classes of instances, Vasco and Morabito (2016) and Cruz et al. (2020), as we were able to solve all of them faster with the new formulation. In addition, we were able to solve to optimality the two instances that could not be solved in Cruz et al. (2020).

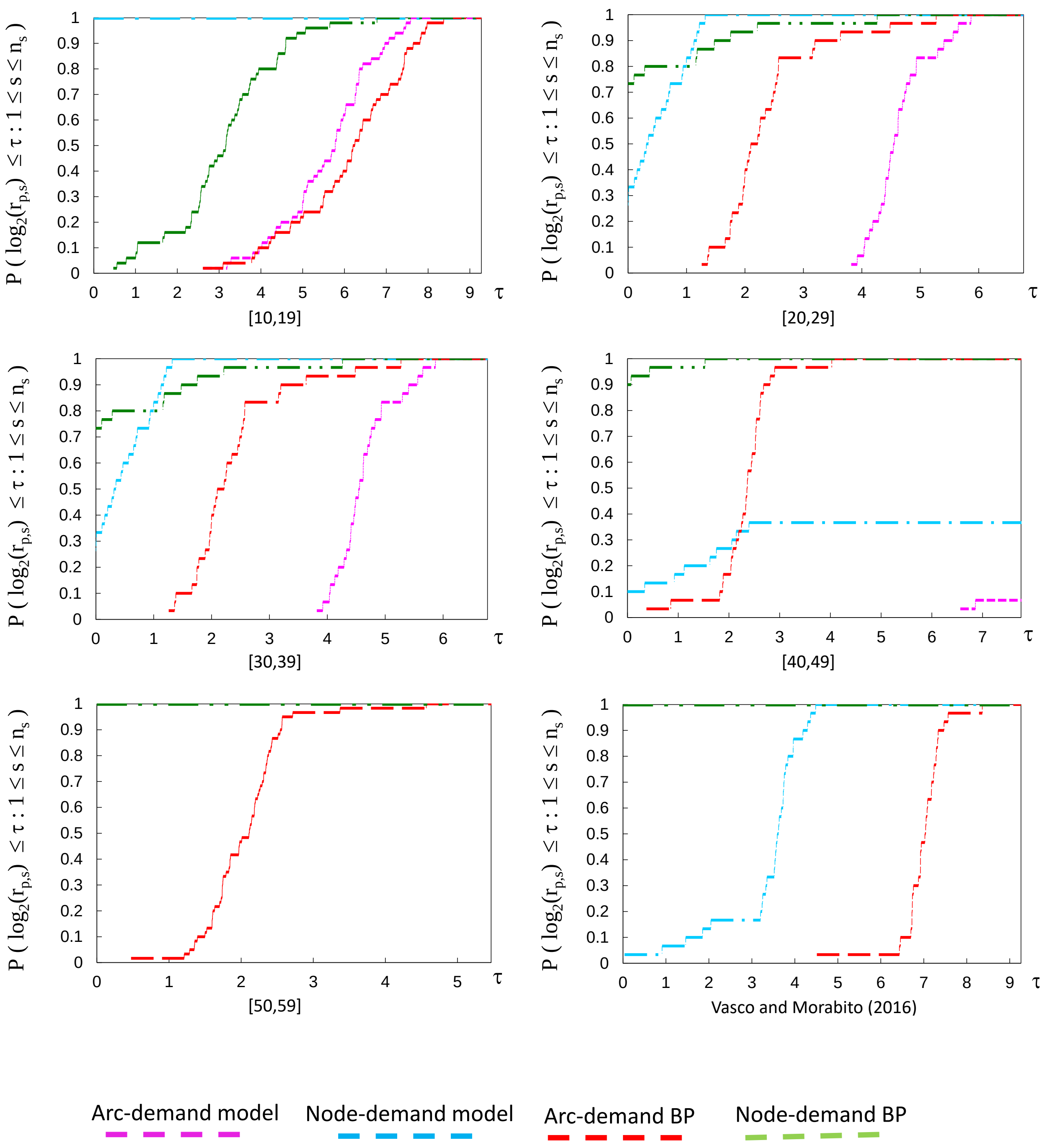
Figure 6 shows the performance profiles (Dolan and Moré, 2002) based on computational times for each set of instances, considering the four solution approaches presented in this work, i.e., the two compact models solved by CPLEX and the two BP methods. The value $P(\tau)$ for a given method corresponds to the fraction of instances for which that method provides solutions with a computational time within a factor of 2^{τ} of the best computational time. When $\tau = 0$, the value $P(\tau)$ indicates the proportion of instances for which a given method performed the best, i.e., was the fastest; when $\tau \to \infty$, the $P(\tau)$ indicates the proportion of instances that were solved by a given method. In sets $[10, 19]$, $[20, 29]$ and $[30, 39]$, all four methods were able to solve all instances. The compact node-demand model is the most efficient in solving the set of smallest instances, i.e., $[10, 19]$. For sets $[20, 29]$ and $[30, 39]$, the node-based BP method is the fastest in most instances, although it takes longer than the corresponding compact model to prove optimality for a few instances in these sets. Then, for the sets $[40, 49]$ and $[50, 59]$, the node-demand BP method clearly superior to any other approach, as it is the fastest in more than 90% of instances and is the first to prove optimality for all instances. Finally, the performance profiles for instances from Vasco and Morabito (2016) reinforces the main conclusion of this work: the proposed node-demand representation, either in compact or column generation based models, is more effective than the arc-demand representation on the benchmark instances, which are based on realistic settings with large-scale networks.

7 Conclusions

In this article, we propose a mixed-integer programming formulation for the Vehicle Allocation Problem (VAP). This formulation relies on a new network representation for the problem, in which nodes correspond to the requests for transportation services. This new representation is particularly suited for instances with a small number of requests with respect to the size of the logistics network - a structure that is characteristic of many real-world instances. Based on this new model, we propose a Dantzig-Wolfe reformulation which is solved with a branch-and-price algorithm. The implementation of the solution method resorts to effective techniques such as a stabilized column generation procedure, an efficient algorithm to solve the subproblems, and a hierarchical branching scheme.

To evaluate the performance of the proposed approaches we ran a batch of computational experiments using benchmark instances from the literature, including the 30 large-scale instances of Vasco and Morabito (2016) and the 200 instances of Cruz et al. (2020). The results showed

Fig. 6: Performance profiles of the four approaches considered in the computational experiments with instances grouped according to the defined sets.



that the proposed node-demand formulation performed better when solved with a commercial MIP solver than the existing model of Vasco and Morabito (2016). Indeed, it was able to obtain better solutions including some for some instances for which the former formulation could not even be mounted due to memory limitations. The proposed BP method also proved efficient, outperforming the BP method of Cruz et al. (2019). Additionally, it solved to optimality all benchmark instances within 2 hours of computation, including two unsolved instances from Cruz et al. (2019).

Some interesting perspectives for future research include exploring other decomposition schemes, devising learning branching procedures and evaluating the possibility of incorporating heuristic procedures within the proposed branch-and-price method to further speed up the computational times. From a modelling perspective, the proposed approach can also be extended to cases that incorporate fleet hiring decisions.

8 Acknowledgments

This study was partially financed by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) [Finance code 001], the National Council for Scientific and Technological Development (CNPq) [grant number 304601/2017-9]; and by the Sao Paulo Research Foundation (FAPESP) [grant numbers 16/23366-9 and 16/01860-1].

References

Alvarez, A. and Munari, P. (2017). An exact hybrid method for the vehicle routing problem with time windows and multiple deliverymen. *Computers & Operations Research*, 83:1–12.

Arunapuram, S., Mathur, K., and Solow, D. (2003). Vehicle routing and scheduling with full truckloads. *Transportation Science*, 37(2):170–182.

Cheung, R. K. and Chen, C.-Y. (1998). A two-stage stochastic network model and solution methods for the dynamic empty container allocation problem. *Transportation Science*, 32(2):142–162.

Cheung, R. K. and Powell, W. B. (1996). An algorithm for multistage dynamic networks with random arc capacities, with an application to dynamic fleet management. *Operations Research*, 44(6):951–963.

Cruz, C., Munari, P., and Morabito, R. (2019). Bounds for the vehicle allocation problem. *DYNA*, 86(208):329–335.

Cruz, C. A., Munari, P., and Morabito, R. (2020). A branch-and-price method for the vehicle allocation problem. *Computers & Industrial Engineering*, 149:106745.

Dejax, P. J. and Crainic, T. G. (1987). Survey paper–a review of empty flows and fleet management models in freight transportation. *Transportation Science*, 21(4):227–248.

Desrosiers, J., Laporte, G., Sauve, M., Soumis, F., and Taillefer, S. (1988). Vehicle routing with full loads. *Computers & Operations Research*, 15(3):219 – 226.

Desrosiers, J., Soumis, F., and Desrochers, M. (1984). Routing with time windows by column generation. *Networks*, 14(4):545–565.

Dolan, E. D. and Moré, J. J. (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213.

Frank, M. and Wolfe, P. (1956). An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110.

Frantzeskakis, L. F. and Powell, W. B. (1990). A successive linear approximation procedure for stochastic, dynamic vehicle allocation problems. *Transportation Science*, 24(1):40–57.

Ghiani, G., Laporte, G., and Musmanno, R. (2004). *Introduction to logistics systems planning and control*. J. Wiley, Hoboken, NJ, USA.

Gondzio, J., González-Brevis, P., and Munari, P. (2013). New developments in the primal-dual column generation technique. *European Journal of Operational Research*, 224(1):41 – 51.

Gondzio, J., González-Brevis, P., and Munari, P. (2016). Large-scale optimization with the primal-dual column generation method. *Mathematical Programming Computation*, 8(1):47–82.

Gorman, M. F. (2015). *Empty Railcar Distribution*, pages 177–189. Springer US, Boston, MA.

Gronalt, M., Hartl, R. F., and Reimann, M. (2003). New savings based algorithms for time constrained pickup and delivery of full truckloads. *European Journal of Operational Research*, 151(3):520 – 535.

Hall, R. W. (1999). Stochastic freight flow patterns: implications for fleet optimization. *Transportation Research Part A: Policy and Practice*, 33(6):449 – 465.

Heydari, R. and Melachrinoudis, E. (2017). A path-based capacitated network flow model for empty railcar distribution. *Annals of Operations Research*, 253:773–798.

Holmberg, K., Joborn, M., and Lundgren, J. T. (1998). Improved empty freight car distribution. *Transportation Science*, 32(2):163–173.

Hungerländer, P. and Steininger, S. (2019). Fleet sizing and empty freight car allocation. In Fortz, B. and Labbé, M., editors, *Operations Research Proceedings 2018*, pages 285–290, Cham. Springer International Publishing.

Joborn, M., Crainic, T. G., Gendreau, M., Holmberg, K., and Lundgren, J. T. (2004). Economies of scale in empty freight car distribution in scheduled railways. *Transportation Science*, 38(2):121–134.

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

Meng, Q. and Wang, S. (2011). Liner shipping service network design with empty container repositioning. *Transportation Research Part E: Logistics and Transportation Review*, 47(5):695–708.

Munari, P. and Gondzio, J. (2013). Using the primal-dual interior point algorithm within the branch-price-and-cut method. *Computers & Operations Research*, 40(8):2026–2036.

Munari, P. and Gondzio, J. (2015). Column generation and branch-and-price with interior point methods. *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*, 3(1):41 – 51.

Narisetty, A. K., Richard, J.-P. P., Ramcharan, D., Murphy, D., Minks, G., and Fuller, J. (2008). An optimization model for empty freight car assignment at union pacific railroad. *INFORMS Journal on Applied Analytics*, 38(2):89–102.

Powell, W. B. (1986). A stochastic model of the dynamic vehicle allocation problem. *Transportation Science*, 20(2):117–129.

Powell, W. B. and Carvalho, T. A. (1998a). Dynamic control of logistics queueing networks for large-scale fleet management. *Transportation Science*, 32(2):90–109.

Powell, W. B. and Carvalho, T. A. (1998b). Real-time optimization of containers and flatcars for intermodal operations. *Transportation Science*, 32(2):110–126.

Shintani, K., Imai, A., Nishimura, E., and Papadimitriou, S. (2007). The container shipping network design problem with empty container repositioning. *Transportation Research Part E: Logistics and Transportation Review*, 43(1):39–59.

Song, D.-P. and Dong, J.-X. (2015). *Empty Container Repositioning*, pages 163–208. Springer International Publishing, Cham.

Vanderbeck, F. (2005). *Implementing Mixed Integer Column Generation*, pages 331–358. Springer US, Boston, MA.

Vanderbeck, F. and Wolsey, L. A. (2010). Reformulation and decomposition of integer programs. In *50 Years of Integer Programming 1958-2008*, pages 431–502. Springer.

Vasco, R. A. and Morabito, R. (2016). The dynamic vehicle allocation problem with application in trucking companies in brazil. *Computers & Operations Research*, 76:118 – 133.

Appendix A Detailed results of computational experiments

This appendix shows the detailed results of the computational experiments described in Section 6, using CPLEX to solve the node-demand and arc-demand models and the branch-and-price (BP) methods based on these models. Recall that a time limit of 7200 seconds was imposed on each run. The tables in this appendix follow a similar structure as in Section 6. For each instance and solution approach, they may show the following columns:

- *Instance* is the name of the instance.
- $LP\ OV$ is the optimal value of the LP relaxation of the compact model.
- $IP\ OV$ is the optimal value of the compact model.
- LP is the time taken by CPLEX to solve the LP relaxation of the compact model.
- IP is the time taken by CPLEX to solve the compact model.
- T_{Root} (sec) is the time taken by the BP method to solve the root node only.
- UB is the best upper bound reached by the BP method.
- ILP_{OV} is the best lower bound reached by the BP method (the optimal value if the method finishes before the time limit).
- *Gap* is the relative optimality gap between the values of UB and ILP_{OV} reached the BP method.

All times are displayed in seconds. A sign "*" indicates that CPLEX could not solve or even mount the model due to lack of computer memory.

Tab. 10: Results from solving the compact model of the arc-based and node-based formulations in instances with terminals ranging from 10 to 14.

Instance	*LP OV*	*IP OV*	*Arc-demand*		*Node-demand*	
			LP	*ILP*	*LP*	*ILP*
10-10-20-20	1940	1937	0.13	0.52	0.03	0.00
10-10-20-25	1874	1857	0.06	0.12	0.02	0.01
10-10-20-30	507	501	0.05	0.10	0.01	0.00
10-10-20-35	2474.5	2464	0.05	0.16	0.02	0.01
10-10-20-50	422.5	421	0.07	0.15	0.10	0.02
11-10-20-20	1601	1584	0.06	0.50	0.01	0.00
11-10-20-25	661.5	659	0.07	0.13	0.40	0.00
11-10-20-30	1606.5	1596	0.06	0.16	0.01	0.00
11-10-20-35	2240.5	2238	0.06	0.29	0.02	0.01
11-10-20-50	2704.5	2701	0.08	0.26	0.03	0.03
12-10-20-20	907	904	0.08	0.59	0.07	0.00
12-10-20-25	396.5	396	0.08	0.21	0.02	0.00
12-10-20-30	337	335	0.08	0.17	0.19	0.00
12-10-20-35	677.5	669	0.08	0.18	0.02	0.01
12-10-20-50	811.5	811	0.08	0.16	0.02	0.02
13-10-20-20	1259.5	1259	0.10	0.18	0.02	0.00
13-10-20-25	671.5	669	0.09	0.26	0.02	0.00
13-10-20-30	581	580	0.09	0.17	0.02	0.00
13-10-20-35	267.5	266	0.09	0.16	0.01	0.00
13-10-20-50	680	678	0.09	0.33	0.02	0.02
14-10-20-20	636.5	633	0.11	0.21	0.03	0.00
14-10-20-25	1295	1292	0.12	0.36	0.04	0.00
14-10-20-30	579	574	0.11	0.31	0.04	0.01
14-10-20-35	2058.5	2057	0.11	0.21	0.01	0.00
14-10-20-50	356	355	0.11	0.65	0.09	0.02

Tab. 11: Results from solving the compact model of the arc-based and node-based formulations in instances with terminals ranging from 15 to 19.

Instance	*LP OV*	*IP OV*	*Arc-demand*		*Node-demand*	
			LP	*ILP*	*LP*	*ILP*
15-10-20-20	564	560	0.13	0.26	0.02	0.00
15-10-20-25	1202.5	1202	0.13	0.27	0.03	0.00
15-10-20-30	1454.5	1451	0.12	0.26	0.01	0.00
15-10-20-35	1679.5	1679	0.12	0.23	0.01	0.01
15-10-20-50	624	623	0.13	0.36	0.03	0.02
16-10-20-20	738	736	0.14	0.46	0.02	0.00
16-10-20-25	1481.5	1479	0.15	0.29	0.02	0.00
16-10-20-30	1800	1788	0.14	0.28	0.02	0.00
16-10-20-35	2040.5	2037	0.14	0.27	0.01	0.00
16-10-20-50	1564.5	1562	0.15	0.33	0.03	0.02
17-10-20-20	632.5	632	0.16	0.30	0.01	0.00
17-10-20-25	1613.5	1611	0.16	0.31	0.01	0.00
17-10-20-30	424.5	424	0.16	0.28	0.01	0.00
17-10-20-35	2349	2347	0.17	0.32	0.02	0.01
17-10-20-50	2210	2202	0.16	0.37	0.03	0.02
18-10-20-20	1147.5	1143	0.17	0.29	0.02	0.00
18-10-20-25	294.5	291	0.17	0.41	0.01	0.00
18-10-20-30	1836	1835	0.17	0.29	0.02	0.01
18-10-20-35	992	991	0.18	0.78	0.01	0.00
18-10-20-50	550.5	550	0.20	0.35	0.04	0.03
19-10-20-20	906.5	905	0.19	0.33	0.01	0.00
19-10-20-25	535.5	535	0.20	0.36	0.12	0.00
19-10-20-30	1405.5	1402	0.22	0.45	0.01	0.00
19-10-20-35	692.5	691	0.20	0.38	0.01	0.00
19-10-20-50	2599	2591	0.21	0.61	0.04	0.02

Tab. 12: Results from solving the compact model of the arc-based and node-based formulations in instances with terminals ranging from 20 to 29.

Instance	*LP OV*	*IP OV*	*Arc-demand*		*Node-demand*	
			LP	*ILP*	*LP*	*ILP*
20-20-100-200	14473	14471	3.49	16.23	2.64	1.56
20-20-130-200	31331	31327	4.33	19.13	2.26	1.26
20-20-150-200	34622.5	34618	5.45	22.10	2.97	1.73
21-20-100-200	2585.5	2585	5.36	19.09	2.48	1.58
21-20-130-200	4056.5	4056	5.01	21.65	2.27	1.30
21-20-150-200	27333.5	27322	5.51	23.34	2.96	1.75
22-20-100-200	4902.5	4902	5.82	23.76	2.56	1.46
22-20-130-200	12304	12301	5.71	25.56	2.89	1.88
22-20-150-200	10972.5	10971	7.14	34.33	4.20	2.68
23-20-100-200	14133.5	14132	4.65	19.48	1.84	1.11
23-20-130-200	21749.5	21745	6.44	27.30	2.90	1.62
23-20-150-200	23366	23358	8.00	31.67	3.53	2.15
24-20-100-200	10157	10155	5.30	24.04	2.29	1.33
24-20-130-200	3486.5	3486	11.08	42.33	3.51	2.56
24-20-150-200	18158	18156	12.68	52.91	3.87	2.38
25-20-100-200	9336.5	9335	5.20	23.05	2.53	1.35
25-20-130-200	33906	33905	8.86	38.92	4.29	2.30
25-20-150-200	32517	32508	9.41	37.72	2.97	2.08
26-20-100-200	9816	9815	5.49	22.13	2.50	1.37
26-20-130-200	16895	16892	7.66	33.17	2.93	1.59
26-20-150-200	3731.5	3731	19.33	67.78	4.36	2.39
27-20-100-200	21095.5	21090	7.05	32.24	2.38	1.66
27-20-130-200	8457.5	8456	11.71	55.14	3.48	2.34
27-20-150-200	26709	26704	9.95	44.40	2.89	1.81
28-20-100-200	3641.5	3641	10.32	42.01	2.89	1.89
28-20-130-200	25359	25355	8.64	36.32	2.65	1.44
28-20-150-200	11225.5	11222	11.36	57.47	4.07	2.41
29-20-100-200	5995	5994	7.09	31.58	2.15	1.28
29-20-130-200	8979.5	8978	11.48	49.86	2.77	1.88
29-20-150-200	7978.5	7977	10.10	44.10	3.22	1.88

Tab. 13: Results from solving the compact model of the arc-based and node-based formulations in instances with terminals ranging from 30 to 39.

Instance	*LP OV*	*IP OV*	*Arc-demand*		*Node-demand*	
			LP	*ILP*	*LP*	*ILP*
30-30-200-300	14513.5	14512	100.44	392.92	9.54	7.26
30-30-230-300	47060.5	47054	104.13	435.42	9.18	6.84
30-30-250-300	69007	68998	153.42	750.87	16.14	13.72
31-30-200-300	51227.5	51221	98.37	436.45	10.46	7.82
31-30-230-300	79462.5	79441	135.00	716.02	13.15	10.75
31-30-250-300	79072	79063	139.71	685.50	14.95	12.64
32-30-200-300	46440	46435	132.54	497.74	9.83	7.67
32-30-230-300	23500	23497	130.23	621.84	11.45	9.94
32-30-250-300	6280	6279	154.61	730.32	16.02	14.49
33-30-200-300	55926	55911	110.64	456.08	10.48	9.10
33-30-230-300	48200.5	48194	268.12	933.43	14.46	12.97
33-30-250-300	33573	33569	101.29	417.62	9.35	7.85
34-30-200-300	4623.5	4623	131.44	639.94	11.93	10.56
34-30-230-300	62736.5	62724	164.23	903.13	16.45	14.56
34-30-250-300	16987	*	*	*	14.17	12.80
35-30-200-300	29115	29110	115.12	536.07	9.46	7.17
35-30-230-300	76030	*	*	*	12.96	10.95
35-30-250-300	47615.5	*	*	*	12.87	10.39
36-30-200-300	40294.5	40288	116.12	615.03	11.07	8.72
36-30-230-300	52908	*	*	*	15.01	12.44
36-30-250-300	62943.5	*	*	*	20.48	18.74
37-30-200-300	57430	57422	144.08	826.57	11.46	9.75
37-30-230-300	7770.5	*	*	*	11.01	9.46
37-30-250-300	41585	*	*	*	12.85	10.59
38-30-200-300	28809.5	*	*	*	9.07	7.21
38-30-230-300	73705.5	*	*	*	13.96	11.67
38-30-250-300	24711	*	*	*	14.76	12.50
39-30-200-300	4989.5	*	*	*	11.01	8.57
39-30-230-300	67965	*	*	*	16.77	13.52
39-30-250-300	63911.5	*	*	*	12.32	10.49

Tab. 14: Results from solving the compact model of the arc-based and node-based formulations in instances with terminals ranging from 40 to 49.

Instance	*LP OV*	*IP OV*	*Arc-demand*		*Node-demand*	
			LP	*ILP*	*LP*	*ILP*
40-36-130-500	48491	48485	233.74	1354.03	17.27	14.75
40-36-150-500	*	*	*	*	23.76	19.29
40-36-170-500	*	*	*	*	*	*
41-36-130-500	4350	4349	409.28	2428.86	30.59	25.71
41-36-150-500	63709.5	63691	*	*	32.25	30.74
41-36-170-500	*	*	*	*	*	*
42-36-130-500	15601.5	15598	*	*	29.42	26.75
42-36-150-500	8179	8178	*	*	35.39	31.74
42-36-170-500	*	*	*	*	*	*
43-36-130-500	35992	35988	*	*	19.23	15.24
43-36-150-500	53283.5	53269	*	*	25.60	22.74
43-36-170-500	*	*	*	*	*	*
44-36-130-500	55765	55748	*	*	32.94	31.96
44-36-150-500	*	*	*	*	*	*
44-36-170-500	*	*	*	*	*	*
45-36-130-500	35025	35017	*	*	32.31	25.74
45-36-150-500	*	*	*	*	*	*
45-36-170-500	*	*	*	*	*	*
46-36-130-500	4479.5	4479	*	*	29.08	24.13
46-36-150-500		*		*	*	*
46-36-170-500	*	*	*	*	*	*
47-36-130-500	*	*	*	*	*	*
47-36-150-500	*	*	*	*	*	*
47-36-170-500	*	*	*	*	*	*
48-36-130-500	*	*	*	*	*	*
48-36-150-500	*	*	*	*	*	*
48-36-170-500	*	*	*	*	*	*
49-36-130-500	*	*	*	*	*	*
49-36-150-500	*	*	*	*	*	*
49-36-170-500	*	*	*	*	*	*

Tab. 15: Results from solving the compact model of the arc-based and node-based formulations in instances from Vasco and Morabito (2016).

Instance	*LP OV*	*IP OV*	*Arc-demand*		*Node-demand*	
			LP	*ILP*	*LP*	*ILP*
53-36-130-300-p1	17437.8	17437.6	*	*	28.81	4.15
53-36-130-300-p2	19260	19260	*	*	5.05	3.19
53-36-130-300-p3	16634.5	16633.8	*	*	4.84	2.34
53-36-130-300-p4	19560	19560	*	*	29.11	4.55
53-36-130-300-p5	18169.2	18169.2	*	*	2.75	2.01
53-36-130-300-p6	19969.4	19969.4	*	*	12.30	5.79
53-36-130-300-p7	19213.8	19213.8	*	*	3.42	2.81
53-36-130-300-p8	18475.9	18472.6	*	*	14.68	4.01
53-36-130-300-p9	15371.4	15371.4	*	*	3.85	3.51
53-36-130-300-p10	18344.8	18344.8	*	*	2.64	3.75
53-36-130-300-p11	16799.9	16799.6	*	*	3.39	2.09
53-36-130-300-p12	22008.4	22008.4	*	*	15.54	4.50
53-36-130-300-p13	19628.6	19628.2	*	*	6.87	3.37
53-36-130-300-p14	19616.6	19616.6	*	*	4.34	2.90
53-36-130-300-p15	20673.2	20673.2	*	*	4.04	4.48
53-36-130-300-p16	17796.2	17796.2	*	*	5.05	3.30
53-36-130-300-p17	17345.2	17345.2	*	*	2.24	2.20
53-36-130-300-p18	17850.7	17849.2	*	*	5.99	3.22
53-36-130-300-p19	18190.6	18190.6	*	*	2.23	2.00
53-36-130-300-p20	20754.4	20754.4	*	*	5.51	2.78
53-36-130-300-p21	16953.2	16953.2	*	*	16.13	7.36
53-36-130-300-p22	18699.2	18699.2	*	*	16.86	5.56
53-36-130-300-p23	21525.6	21525.6	*	*	13.97	4.41
53-36-130-300-p24	18266.2	18266.2	*	*	2.60	2.31
53-36-130-300-p25	17064.8	17064.8	*	*	6.48	3.43
53-36-130-300-p26	20324.4	20324.4	*	*	3.50	2.63
53-36-130-300-p27	20003	20003	*	*	2.69	2.67
53-36-130-300-p28	17956	17956	*	*	3.82	2.79
53-36-130-300-p29	19074.6	19074.6	*	*	2.21	2.29
53-36-130-300-p30	16464.6	16464.6	*	*	3.63	2.69

Tab. 16: Results from solving the arc-based and node-based formulations via Branch-and-Price in instances with terminals ranging from 10 to 14.

Instance	*Arc-demand*					*Node-demand*				
	T_{Root}	UB	ILP_{OV}	*Gap*	T_{BP}	T_{Root}	UB	ILP_{OV}	*Gap*	T_{BP}
10-10-20-20	0.09	1937	1937	0.0%	0.18	0.006	1937	1937	0.0%	0.06
10-10-20-25	0.06	1857	1857	0.0%	0.15	0.005	1857	1857	0.0%	0.06
10-10-20-30	0.06	501	501	0.0%	0.44	0.005	501	501	0.0%	0.09
10-10-20-35	0.07	2464	2464	0.0%	0.18	0.005	2464	2464	0.0%	0.06
10-10-20-50	0.07	421	421	0.0%	0.15	0.007	421	421	0.0%	0.05
11-10-20-20	0.09	1584	1584	0.0%	0.19	0.005	1584	1584	0.0%	0.06
11-10-20-25	0.09	659	659	0.0%	0.18	0.005	659	659	0.0%	0.03
11-10-20-30	0.06	1596	1596	0.0%	0.18	0.005	1596	1596	0.0%	0.02
11-10-20-35	0.07	2238	2238	0.0%	0.20	0.006	2238	2238	0.0%	0.07
11-10-20-50	0.37	2701	2701	0.0%	0.23	0.007	2701	2701	0.0%	0.06
12-10-20-20	0.16	904	904	0.0%	0.76	0.005	904	904	0.0%	0.10
12-10-20-25	0.09	396	396	0.0%	0.25	0.005	396	396	0.0%	0.06
12-10-20-30	0.11	335	335	0.0%	0.65	0.005	335	335	0.0%	0.53
12-10-20-35	0.13	669	669	0.0%	1.79	0.006	669	669	0.0%	0.27
12-10-20-50	0.10	811	811	0.0%	0.25	0.007	811	811	0.0%	0.04
13-10-20-20	0.12	1259	1259	0.0%	0.27	0.006	1259	1259	0.0%	0.03
13-10-20-25	0.12	669	669	0.0%	0.27	0.007	669	669	0.0%	0.07
13-10-20-30	0.18	580	580	0.0%	0.29	0.006	580	580	0.0%	0.06
13-10-20-35	0.10	266	266	0.0%	0.80	0.005	266	266	0.0%	0.10
13-10-20-50	0.11	678	678	0.0%	0.42	0.007	678	678	0.0%	0.08
14-10-20-20	0.16	633	633	0.0%	0.56	0.007	633	633	0.0%	0.08
14-10-20-25	0.11	1292	1292	0.0%	0.35	0.005	1292	1292	0.0%	0.03
14-10-20-30	0.13	574	574	0.0%	0.30	0.005	574	574	0.0%	0.04
14-10-20-35	0.13	2057	2057	0.0%	0.29	0.006	2057	2057	0.0%	0.03
14-10-20-50	0.13	355	355	0.0%	0.31	0.007	355	355	0.0%	0.07

Tab. 17: Results from solving the arc-based and node-based formulations via Branch-and-Price in instances with terminals ranging from 15 to 19.

Instance	*Arc-demand*					*Node-demand*				
	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}
15-10-20-20	0.14	560	560	0.0%	0.37	0.005	560	560	0.0%	0.04
15-10-20-25	0.32	1202	1202	0.0%	0.57	0.006	1202	1202	0.0%	0.07
15-10-20-30	0.11	1451	1451	0.0%	0.32	0.005	1451	1451	0.0%	0.02
15-10-20-35	0.15	1679	1679	0.0%	0.30	0.006	1679	1679	0.0%	0.03
15-10-20-50	0.15	623	623	0.0%	0.41	0.007	623	623	0.0%	0.03
16-10-20-20	0.32	736	736	0.0%	0.52	0.005	736	736	0.0%	0.08
16-10-20-25	0.16	1479	1479	0.0%	0.38	0.005	1479	1479	0.0%	0.03
16-10-20-30	0.16	1788	1788	0.0%	0.37	0.005	1788	1788	0.0%	0.06
16-10-20-35	0.14	2037	2037	0.0%	0.42	0.004	2037	2037	0.0%	0.05
16-10-20-50	0.16	1562	1562	0.0%	0.40	0.006	1562	1562	0.0%	0.04
17-10-20-20	0.24	632	632	0.0%	0.47	0.005	632	632	0.0%	0.03
17-10-20-25	0.18	1611	1611	0.0%	0.87	0.004	1611	1611	0.0%	0.02
17-10-20-30	0.18	424	424	0.0%	0.48	0.007	424	424	0.0%	0.03
17-10-20-35	0.24	2346.99	2346.99	0.0%	0.43	0.006	2347	2347	0.0%	0.05
17-10-20-50	0.17	2202	2202	0.0%	0.63	0.006	2202	2202	0.0%	0.09
18-10-20-20	0.31	1143	1143	0.0%	0.91	0.006	1143	1143	0.0%	0.09
18-10-20-25	0.23	291	291	0.0%	0.59	0.004	291	291	0.0%	0.02
18-10-20-30	0.25	1835	1835	0.0%	0.81	0.005	1835	1835	0.0%	0.05
18-10-20-35	0.18	991	991	0.0%	0.45	0.005	991	991	0.0%	0.04
18-10-20-50	0.22	550	550	0.0%	0.51	0.007	550	550	0.0%	0.04
19-10-20-20	0.25	905	905	0.0%	0.50	0.005	905	905	0.0%	0.03
19-10-20-25	0.28	535	535	0.0%	0.62	0.005	535	535	0.0%	0.03
19-10-20-30	0.18	1402	1402	0.0%	0.83	0.005	1402	1402	0.0%	0.04
19-10-20-35	0.22	691	691	0.0%	0.49	0.006	691	691	0.0%	0.06
19-10-20-50	0.23	2591	2591	0.0%	0.86	0.006	2591	2591	0.0%	0.06

Tab. 18: Results from solving the arc-based and node-based formulations via Branch-and-Price in instances with terminals ranging from 20 to 29.

Instance	*Arc-demand*					*Node-demand*				
	T_{Root}	UB	ILP_{OV}	*Gap*	T_{BP}	T_{Root}	UB	ILP_{OV}	*Gap*	T_{BP}
20-20-100-200	1.48	14471	14471	0.0%	5.36	0.09	14471	14471	0.0%	1.15
20-20-130-200	1.51	31327	31327	0.0%	7.51	0.11	31327	31327	0.0%	4.26
20-20-150-200	1.70	34618	34618	0.0%	2.63	0.12	34618	34618	0.0%	1.10
21-20-100-200	1.75	2585	2585	0.0%	3.09	0.08	2585	2585	0.0%	0.98
21-20-130-200	2.05	4056	4056	0.0%	3.51	0.13	4056	4056	0.0%	1.02
21-20-150-200	1.88	27322	27322	0.0%	3.56	0.12	27322	27322	0.0%	1.06
22-20-100-200	1.96	4902	4902	0.0%	5.71	0.10	4902	4902	0.0%	1.36
22-20-130-200	1.99	12301	12301	0.0%	4.04	0.11	12301	12301	0.0%	0.94
22-20-150-200	2.12	10971	10971	0.0%	5.47	0.12	10971	10971	0.0%	1.40
23-20-100-200	1.90	14132	14132	0.0%	2.99	0.09	14132	14132	0.0%	0.75
23-20-130-200	1.94	21745	21745	0.0%	3.02	0.09	21745	21745	0.0%	1.17
23-20-150-200	2.20	23358	23358	0.0%	5.03	0.12	23358	23358	0.0%	1.93
24-20-100-200	2.37	10155	10155	0.0%	6.28	0.08	10155	10155	0.0%	1.32
24-20-130-200	3.51	3486	3486	0.0%	13.02	0.14	3486	3486	0.0%	5.79
24-20-150-200	3.26	18156	18156	0.0%	9.43	0.15	18156	18156	0.0%	2.36
25-20-100-200	2.33	9335	9335	0.0%	5.93	0.07	9335	9335	0.0%	1.09
25-20-130-200	2.64	33905	33905	0.0%	4.07	0.12	33905	33905	0.0%	0.99
25-20-150-200	2.91	32508	32508	0.0%	6.05	0.14	32508	32508	0.0%	1.80
26-20-100-200	2.64	9815	9815	0.0%	4.16	0.08	9815	9815	0.0%	0.73
26-20-130-200	2.76	16892	16892	0.0%	3.93	0.10	16892	16892	0.0%	0.70
26-20-150-200	3.94	3731	3731	0.0%	21.92	0.14	3731	3731	0.0%	5.35
27-20-100-200	3.31	21090	21090	0.0%	4.54	0.08	21090	21090	0.0%	0.76
27-20-130-200	3.95	8456	8456	0.0%	29.05	0.14	8456	8456	0.0%	6.51
27-20-150-200	3.29	26704	26704	0.0%	6.69	0.12	26704	26704	0.0%	1.94
28-20-100-200	3.23	3641	3641	0.0%	42.24	0.08	3641	3641	0.0%	8.75
28-20-130-200	3.32	25355	25355	0.0%	4.63	0.10	25355	25355	0.0%	1.19
28-20-150-200	3.73	11222	11222	0.0%	5.49	0.12	11222	11222	0.0%	1.15
29-20-100-200	3.69	5994	5994	0.0%	7.48	0.08	5994	5994	0.0%	1.56
29-20-130-200	3.60	8978	8978	0.0%	72.53	0.12	8978	8978	0.0%	36.02
29-20-150-200	3.91	7977	7977	0.0%	6.73	0.11	7977	7977	0.0%	0.76

Tab. 19: Results from solving the arc-based and node-based formulations via Branch-and-Price in instances with terminals ranging from 30 to 39.

Instance	*Arc-demand*					*Node-demand*				
	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}
30-30-200-300	8.23	14512	14512	0.0%	16.41	0.36	14512	14512	0.0%	3.96
30-30-230-300	9.41	47054	47054	0.0%	21.33	0.46	47054	47054	0.0%	6.41
30-30-250-300	10.03	68998	68998	0.0%	108.11	0.51	68998	68998	0.0%	71.58
31-30-200-300	8.46	51221	51221	0.0%	11.79	0.30	51221	51221	0.0%	2.08
31-30-230-300	9.68	79441	79441	0.0%	1539.31	0.41	79441	79441	0.0%	382.25
31-30-250-300	10.34	79063	79063	0.0%	13.13	0.49	79063	79063	0.0%	4.01
32-30-200-300	9.11	46435	46435	0.0%	39.08	0.35	46435	46435	0.0%	9.51
32-30-230-300	10.83	23497	23497	0.0%	18.04	0.40	23497	23497	0.0%	2.51
32-30-250-300	17.02	6279	6279	0.0%	394.39	0.57	6279	6279	0.0%	176.37
33-30-200-300	9.86	55911	55911	0.0%	254.35	0.35	55911	55911	0.0%	21.98
33-30-230-300	11.13	48194	48194	0.0%	32.99	0.40	48194	48194	0.0%	6.21
33-30-250-300	11.59	33569	33569	0.0%	65.80	0.41	33569	33569	0.0%	16.85
34-30-200-300	14.44	4623	4623	0.0%	63.03	0.45	4623	4623	0.0%	14.86
34-30-230-300	12.19	62724	62724	0.0%	56.14	0.45	62724	62724	0.0%	11.04
34-30-250-300	12.77	16985	16985	0.0%	21.54	0.44	16985	16985	0.0%	4.00
35-30-200-300	11.06	29110	29110	0.0%	170.60	0.34	29110	29110	0.0%	34.44
35-30-230-300	13.41	76015	76015	0.0%	45.84	0.40	76015	76015	0.0%	6.39
35-30-250-300	13.90	47613	47613	0.0%	61.65	0.44	47613	47613	0.0%	13.50
36-30-200-300	13.64	40288	40288	0.0%	29.92	0.39	40288	40288	0.0%	4.82
36-30-230-300	14.57	52902	52902	0.0%	28.51	0.44	52902	52902	0.0%	6.79
36-30-250-300	14.50	62932	62932	0.0%	21.88	0.47	62932	62932	0.0%	2.76
37-30-200-300	13.07	57422	57422	0.0%	16.96	0.29	57422	57422	0.0%	6.33
37-30-230-300	16.65	7769	7769	0.0%	1250.38	0.39	7769	7769	0.0%	273.13
37-30-250-300	16.07	41580	41580	0.0%	48.15	0.41	41580	41580	0.0%	7.76
38-30-200-300	14.08	28805	28805	0.0%	465.01	0.34	28805	28805	0.0%	75.33
38-30-230-300	15.72	73698	73698	0.0%	25.18	0.44	73698	73698	0.0%	3.72
38-30-250-300	18.31	24709	24709	0.0%	26.90	0.48	24709	24709	0.0%	4.05
39-30-200-300	18.71	4989	4989	0.0%	69.13	0.39	4989	4989	0.0%	9.22
39-30-230-300	17.87	67958	67958	0.0%	238.48	0.43	67958	67958	0.0%	44.26
39-30-250-300	17.57	63894	63894	0.0%	60.95	0.46	63894	63894	0.0%	9.59

Tab. 20: Results from solving the arc-based and node-based formulations via Branch-and-Price in instances with terminals ranging from 40 to 49.

Instance	*Arc-demand*					*Node-demand*				
	T_{Root}	UB	ILP_{OV}	*Gap*	T_{BP}	T_{Root}	UB	ILP_{OV}	*Gap*	T_{BP}
40-36-130-500	15.23	48485	48485	0.0%	47.90	0.46	48485	48485	0.0%	11.66
40-36-150-500	17.29	28510	28510	0.0%	32.65	0.62	28510	28510	0.0%	8.89
40-36-170-500	18.22	76106	76106	0.0%	52.27	0.69	76106	76106	0.0%	11.44
41-36-130-500	18.11	4349	4349	0.0%	113.69	0.48	4349	4349	0.0%	27.03
41-36-150-500	19.53	63691	63691	0.0%	24.16	0.62	63691	63691	0.0%	5.85
41-36-170-500	20.81	54550	54550	0.0%	25.21	0.70	54550	54550	0.0%	4.64
42-36-130-500	19.18	15598	15598	0.0%	42.48	0.47	15598	15598	0.0%	6.05
42-36-150-500	22.40	8178	8178	0.0%	26.89	0.78	8178	8178	0.0%	7.63
42-36-170-500	21.62	36376	36376	0.0%	80.60	0.67	36376	36376	0.0%	62.20
43-36-130-500	20.24	35988	35988	0.0%	25.25	0.46	35988	35988	0.0%	4.94
43-36-150-500	21.36	53269	53269	0.0%	138.62	0.61	53269	53269	0.0%	30.75
43-36-170-500	21.77	37484	37484	0.0%	65.34	0.68	37484	37484	0.0%	15.47
44-36-130-500	20.28	55748	55748	0.0%	79.92	0.47	55748	55748	0.0%	16.84
44-36-150-500	22.88	70064	70064	0.0%	39.81	0.61	70064	70064	0.0%	7.70
44-36-170-500	25.29	37247	37247	0.0%	128.38	0.68	37247	37247	0.0%	26.54
45-36-130-500	21.41	35017	35017	0.0%	38.40	0.46	35017	35017	0.0%	7.64
45-36-150-500	22.82	35095	35095	0.0%	69.02	0.53	35095	35095	0.0%	13.67
45-36-170-500	25.23	67070	67070	0.0%	35.68	0.60	67070	67070	0.0%	6.51
46-36-130-500	26.07	4479	4479	0.0%	393.58	0.61	4479	4479	0.0%	69.61
46-36-150-500	24.76	25481	25481	0.0%	232.55	0.61	25481	25481	0.0%	62.85
46-36-170-500	28.27	45355	45355	0.0%	37.83	0.67	45355	45355	0.0%	6.55
47-36-130-500	34.85	4825	4825	0.0%	69.29	0.69	4825	4825	0.0%	11.31
47-36-150-500	26.57	54139	54139	0.0%	76.43	0.60	54139	54139	0.0%	42.31
47-36-170-500	30.42	46154	46154	0.0%	78.83	0.69	46154	46154	0.0%	13.71
48-36-130-500	28.28	26069	26069	0.0%	91.27	0.59	26069	26069	0.0%	15.70
48-36-150-500	33.14	4848	4848	0.0%	118.50	0.72	4848	4848	0.0%	20.66
48-36-170-500	29.44	68476	68476	0.0%	163.78	0.59	68476	68476	0.0%	26.56
49-36-130-500	28.81	38305	38305	0.0%	37.78	0.52	38305	38305	0.0%	5.06
49-36-150-500	40.63	4944	4944	0.0%	121.98	0.70	4944	4944	0.0%	19.01
49-36-170-500	34.51	4676	4676	0.0%	72.37	0.69	4676	4676	0.0%	14.07

Tab. 21: Results from solving the arc-based and node-based formulations via Branch-and-Price in instances with terminals ranging from 50 to 54.

Instance	*Arc-demand*					*Node-demand*				
	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}
50-36-100-700	23.56	50961	50961	0.0%	34.71	0.69	50961	50961	0.0%	7.61
50-36-130-700	29.24	57051.5	57048	**0.0001%**	**7200.00**	1.02	570489	57048	0.0%	5822.36
50-36-150-700	31.62	22656	22656	0.0%	68.35	1.19	22656	22656	0.0%	20.72
50-36-180-700	34.10	62122	62122	0.0%	627.10	1.39	62122	62122	0.0%	245.91
50-36-200-700	34.85	28843	28843	0.0%	37.93	1.50	28843	28843	0.0%	10.51
50-36-250-700	40.64	111448	111444	**0.0001%**	**7200.00**	1.91	111444	111444	0.0%	761.11
51-36-100-700	24.03	17676	17676	0.0%	99.31	0.59	17676	17676	0.0%	25.22
51-36-130-700	30.62	62845	62845	0.0%	42.11	1.02	62845	62845	0.0%	9.77
51-36-150-700	30.51	61608	61608	0.0%	171.96	1.16	61608	61608	0.0%	56.61
51-36-180-700	37.18	45116	45116	0.0%	1798.33	1.39	45116	45116	0.0%	777.42
51-36-200-700	38.92	69124	69124	0.0%	55.16	1.72	69124	69124	0.0%	10.71
51-36-250-700	49.22	29392	29392	0.0%	772.57	2.18	29392	29392	0.0%	268.88
52-36-100-700	25.76	47273	47273	0.0%	38.97	0.68	47273	47273	0.0%	7.48
52-36-130-700	33.24	60081	60081	0.0%	63.88	1.02	60081	60081	0.0%	11.91
52-36-150-700	28.20	75988	75988	0.0%	46.27	0.89	75988	75988	0.0%	7.81
52-36-180-700	37.01	54693	54693	0.0%	208.32	1.22	54693	54693	0.0%	73.72
52-36-200-700	43.32	33908	33908	0.0%	91.11	1.53	33908	33908	0.0%	18.60
52-36-250-700	52.17	8242	8242	0.0%	224.81	2.19	8242	8242	0.0%	9.48
53-36-100-700	26.39	18727	18727	0.0%	150.27	0.58	18727	18727	0.0%	58.73
53-36-130-700	31.03	36388	36388	0.0%	51.06	0.88	36388	36388	0.0%	12.69
53-36-150-700	38.67	63060	63060	0.0%	55.33	1.15	63060	63060	0.0%	11.14
53-36-180-700	38.71	28785	28785	0.0%	1023.86	1.39	28785	28785	0.0%	314.93
53-36-220-700	42.52	101729	101729	0.0%	105.90	1.44	101729	101729	0.0%	43.48
53-36-250-700	49.47	104922	104922	0.0%	65.19	1.90	104922	104922	0.0%	18.85
54-36-100-700	29.48	23551	23551	0.0%	137.10	0.68	23551	23551	0.0%	31.67
54-36-130-700	33.05	39535	39535	0.0%	45.46	1.01	39535	39535	0.0%	9.91
54-36-150-700	38.63	4722	4722	0.0%	81.05	1.16	4722	4722	0.0%	24.17
54-36-180-700	40.40	36084	36084	0.0%	107.63	1.56	36084	36084	0.0%	35.34
54-36-200-700	41.92	54198	54198	0.0%	45.06	1.31	54198	54198	0.0%	14.89
54-36-250-700	57.90	8531	8531	0.0%	1418.64	2.14	8531	8531	0.0%	469.17

Tab. 22: Results from solving the arc-based and node-based formulations via Branch-and-Price in instances with terminals ranging from 55 to 59.

Instance	*Arc-demand*					*Node-demand*				
	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}
55-36-100-700	29.75	28195	28195	0.0%	100.46	0.68	28195	28195	0.0%	16.99
55-36-130-700	47.77	4506	4506	0.0%	73.97	1.02	4506	4506	0.0%	12.71
55-36-150-700	40.81	7853	7853	0.0%	49.44	1.16	7853	7853	0.0%	14.83
55-36-180-700	43.37	40857	40857	0.0%	109.13	1.37	40857	40857	0.0%	27.87
55-36-200-700	45.23	29966	29966	0.0%	120.55	1.52	29966	29966	0.0%	30.83
55-36-250-700	55.72	54658	54658	0.0%	108.08	2.15	54658	54658	0.0%	34.86
56-36-100-700	35.63	3481	3481	0.0%	54.07	0.69	3481	3481	0.0%	5.23
56-36-130-700	52.44	4433	4433	0.0%	57.63	1.03	4433	4433	0.0%	13.00
56-36-150-700	41.97	11529	11529	0.0%	98.02	1.17	11529	11529	0.0%	20.91
56-36-180-700	48.28	44014	44014	0.0%	244.86	1.37	44014	44014	0.0%	73.35
56-36-200-700	51.13	71438	71438	0.0%	65.10	1.32	71438	71438	0.0%	14.93
56-36-250-700	54.70	85911	85911	0.0%	1942.71	2.12	85911	85911	0.0%	546.20
57-36-100-700	33.38	12646	12646	0.0%	49.92	0.68	12646	12646	0.0%	7.60
57-36-130-700	37.13	59518	59518	0.0%	153.09	0.87	59518	59518	0.0%	30.21
57-36-150-700	40.73	51978	51978	0.0%	80.27	1.00	51978	51978	0.0%	13.53
57-36-180-700	47.38	62345	62345	0.0%	62.52	1.54	62345	62345	0.0%	13.76
57-36-200-700	52.76	91807	91807	0.0%	57.93	1.72	91807	91807	0.0%	10.89
57-36-250-700	62.00	47464	47464	0.0%	63.98	2.12	47464	47464	0.0%	19.03
58-36-100-700	48.58	31050	31050	0.0%	99.66	1.19	31050	31050	0.0%	27.76
58-36-130-700	31.85	18790	18790	0.0%	49.34	0.58	18790	18790	0.0%	9.71
58-36-150-700	53.58	64722	64722	0.0%	51.38	1.50	64722	64722	0.0%	11.63
58-36-180-700	42.30	49225	49225	0.0%	164.57	0.98	49225	49225	0.0%	34.26
58-36-200-700	62.03	24632	24632	0.0%	80.03	2.12	24632	24632	0.0%	30.48
58-36-250-700	46.64	116424	116424	0.0%	2271.38	1.14	116424	116424	0.0%	632.19
59-36-100-700	46.65	3566	3566	0.0%	511.11	0.78	3566	3566	0.0%	99.67
59-36-130-700	50.34	4956	4956	0.0%	1147.60	1.28	4956	4956	0.0%	201.27
59-36-150-700	60.56	5374	5374	0.0%	79.93	1.12	5374	5374	0.0%	14.84
59-36-180-700	52.15	83504	83504	0.0%	256.82	1.36	83504	83504	0.0%	56.59
59-36-200-700	55.35	46808	46808	0.0%	73.87	1.31	46808	46808	0.0%	14.88
59-36-250-700	63.43	96410	96410	0.0%	82.01	1.86	96410	96410	0.0%	24.60

Tab. 23: Results from solving the arc-based and node-based formulations via Branch-and-Price in instances from Vasco and Morabito (2016).

Instance	*Arc-demand*					*Node-demand*				
	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}	T_{Root}	UB	ILP_{OV}	Gap	T_{BP}
53-36-130-300-p1	32.98	17437.6	17437.6	0.0%	46.89	0.53	17437.6	17437.6	0.0%	0.45
53-36-130-300-p2	29.10	19260	19260	0.0%	29.10	0.24	19260	19260	0.0%	0.24
53-36-130-300-p3	31.03	16633.8	16633.8	0.0%	409.09	0.41	16634.2	16633.8	0.0%	1.24
53-36-130-300-p4	36.83	19560	19560	0.0%	36.83	0.34	19560	19560	0.0%	0.34
53-36-130-300-p5	29.08	18169.2	18169.2	0.0%	29.08	0.20	18169.2	18169.2	0.0%	0.20
53-36-130-300-p6	33.00	19969.4	19969.4	0.0%	33.00	0.28	19969.4	19969.4	0.0%	0.28
53-36-130-300-p7	31.51	19213.8	19213.8	0.0%	31.51	0.30	19213.8	19213.8	0.0%	0.30
53-36-130-300-p8	29.23	18472.6	18472.6	0.0%	127.94	0.17	18472.6	18472.6	0.0%	1.46
53-36-130-300-p9	36.74	15371.4	15371.4	0.0%	36.74	0.30	15371.4	15371.4	0.0%	0.30
53-36-130-300-p10	25.20	18344.8	18344.8	0.0%	25.20	0.19	18344.8	18344.8	0.0%	0.19
53-36-130-300-p11	36.01	16799.6	16799.6	0.0%	46.50	0.17	16799.7	16799.6	0.0%	2.04
53-36-130-300-p12	34.73	22008.4	22008.4	0.0%	34.73	0.33	22008.4	22008.4	0.0%	0.33
53-36-130-300-p13	37.51	19628.2	19628.2	0.0%	80.58	0.15	19628.2	19628.2	0.0%	0.93
53-36-130-300-p14	31.77	19616.6	19616.6	0.0%	31.77	0.22	19616.6	19616.6	0.0%	0.22
53-36-130-300-p15	38.57	20673.2	20673.2	0.0%	38.57	0.25	20673.2	20673.2	0.0%	0.25
53-36-130-300-p16	36.01	17796.2	17796.2	0.0%	36.01	0.23	17796.2	17796.2	0.0%	0.23
53-36-130-300-p17	33.52	17345.2	17345.2	0.0%	33.52	0.18	17345.2	17345.2	0.0%	0.18
53-36-130-300-p18	36.62	17849.2	17849.2	0.0%	82.99	0.51	17849.2	17849.2	0.0%	0.78
53-36-130-300-p19	37.23	18190.6	18190.6	0.0%	37.23	0.21	18190.6	18190.6	0.0%	0.21
53-36-130-300-p20	36.26	20754.4	20754.4	0.0%	36.26	0.24	20754.4	20754.4	0.0%	0.24
53-36-130-300-p21	44.80	16953.2	16953.2	0.0%	44.80	0.33	16953.2	16953.2	0.0%	0.33
53-36-130-300-p22	37.64	18699.2	18699.2	0.0%	37.64	0.36	18699.2	18699.2	0.0%	0.36
53-36-130-300-p23	34.35	21525.6	21525.6	0.0%	34.35	0.28	21525.6	21525.6	0.0%	0.28
53-36-130-300-p24	27.34	18266.2	18266.2	0.0%	27.34	0.17	18266.2	18266.2	0.0%	0.17
53-36-130-300-p25	33.85	17064.8	17064.8	0.0%	33.85	0.28	17064.8	17064.8	0.0%	0.28
53-36-130-300-p26	30.16	20324.4	20324.4	0.0%	30.16	0.23	20324.4	20324.4	0.0%	0.23
53-36-130-300-p27	27.10	20003	20003	0.0%	27.10	0.21	20003	20003	0.0%	0.21
53-36-130-300-p28	37.92	17956	17956	0.0%	37.92	0.24	17956	17956	0.0%	0.24
53-36-130-300-p29	30.23	19074.6	19074.6	0.0%	30.23	0.22	19074.6	19074.6	0.0%	0.22
53-36-130-300-p30	33.68	16464.6	16464.6	0.0%	33.68	0.22	16464.6	16464.6	0.0%	0.22