CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

The Two-Period Vehicle Routing Problem with Crossdocking and Transfers

Fester Kroep
Paul Buijs
Leandro C. Coelho
Kees Jan Roodbergen

August 2017

CIRRELT-2017-53

Document de travail également publié par la Faculté des sciences de l'administration de l'Université Laval, sous le numéro FSA-2017-010.

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

The Two-Period Vehicle Routing Problem with Crossdocking and Transfers

Fester Kroep[1], Paul Buijs[1], Leandro C. Coelho[2,*], Kees Jan Roodbergen[1]

[1] Faculty of Economics and Business, University of Groningen, The Netherlands

[2] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Operations and Decision Systems, 2325 de la Terrasse, Université Laval, Québec, Canada G1V 0A6

Abstract. This paper introduces the Two-Period Vehicle Routing Problem with Crossdocking and Transfers (2P-VRPCDT), which extends several problems in the class of one-to-one pickup and delivery, as well as the vehicle routing problem with crossdocking (VRPCD). The 2P-VRPCDT involves transportation requests between unique origin-destination pairs. All goods are picked up in the first period and returned to one of several crossdocks. Before being delivered to their destination in the second period, goods can be transferred to another, more conveniently located crossdock. We develop a mixed integer program (MIP) and propose several sets of valid inequalities, which decreases the average runtime by 82%. With our MIP we can optimally solve all instances with 2 crossdocks and 12 requests. On a set of benchmark instances for the VRPCD we prove optimality for 148 out of 243 instances, matching best known solutions obtained heuristically in the literature. Our results show that scheduling routes for two crossdocks jointly rather than independently yields an average cost reduction of 12.5%. By incorporating transfers between crossdocks, cost savings increase to 14.6% and almost 13% fewer vehicles are needed.

Keywords. Vehicle routing, pickup and delivery, crossdocking, transfers, valid inequalities.

Acknowledgements. This project was partly funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant 2014-05764 nd the Netherlands Organisation for Scientific Research (NWO) under grant 438-15-525. This support is greatly acknowledged. We would like to thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Peregrine high performance computing cluster.

---

* Corresponding author: Leandro.Coelho@cirrelt.ca

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

1 Introduction

We introduce the Two-Period Vehicle Routing Problem with Crossdocking and Transfers (2P-VRPCDT), where goods are transported from a specific supplier location (origin) to its corresponding customer location (destination). Each shipment between such an origin-destination pair is called a *request*, and a number of requests has to be satisfied. All goods are picked up from origins in period 1 and shipped to a crossdock. At the crossdock, goods are unloaded, handled, and redistributed among outgoing vehicles. Before goods are delivered to their destination in period 2, they can be transferred to a more conveniently located crossdock by means of roundtrips between crossdocks (Figure 1). An example timeline of the 2P-VRPCDT is depicted in Figure 2. Pickup, delivery, and transfer routes are executed by unlimited, homogeneous fleets of vehicles available at the crossdocks. All origins and destinations must be visited exactly once, i.e., no split pickups or deliveries are allowed.

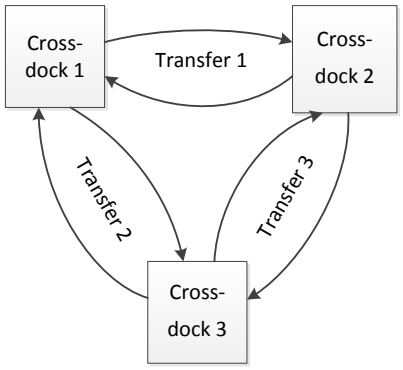


Figure 1: Example of transfers between three crossdocks.

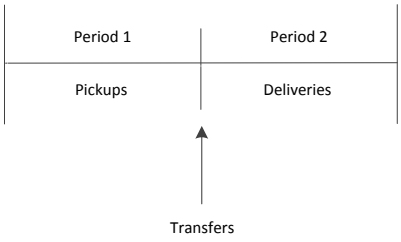


Figure 2: Timeline displaying the events during the two periods.

The two distinct periods for pickups and deliveries generally create more opportunities for consolidation and the time restrictions specified in a request often allow, or even require it (Buijs et al. [2016]). Although many transportation firms operate more than one crossdock, the literature on multiple crossdock routing problems is scarce. Whereas the multi-depot vehicle routing problem (MDVRP) has received quite some attention

(Braekers et al. [2016]), only a few papers study problems involving multiple crossdocks (e.g., Chen et al. [2006] and Miao et al. [2012]), none of which is closely related to our problem. When multiple crossdocks are available, transfers between them can potentially further enhance efficiency. They allow for shorter pickup and delivery routes and enable a high fill rate of vehicles. On the other hand, transfers add to the total distance, and the trade-off between these effects determines the usefulness of transfers.

In this paper, we develop a mixed integer program (MIP) for the 2P-VRPCDT and propose several sets of valid inequalities to strengthen its formulation. We assess the effect of these inequalities by comparing runtimes and solution bounds for formulations with and without valid inequalities. We compare solutions for three different problem variants. The first one is the problem with independently scheduled routes, where origin-destination pairs are assigned to the crossdock located closest to the origin before constructing routes, which is common in practice. The second variant is the problem with jointly scheduled routes without considering transfers, where origin-destination pairs are optimally assigned to a crossdock, making it more flexible. Finally, we consider the problem with jointly scheduled routes with transfers between crossdocks (the 2P-VRPCDT), which is the most flexible problem variant. We show that cost savings can be achieved by jointly scheduling routes and incorporating transfers between crossdocks, and that fewer vehicles are needed to visit all origins and destinations. We evaluate the effects of changes in parameters, such as the number of requests and vehicle capacity.

Pickup and delivery problems (PDP) have been extensively studied in the operations research literature (see Berbeglia et al. [2007] and Parragh et al. [2008]). A classification proposed by Berbeglia et al. [2007] divides PDPs into three categories. Following this classification, our 2P-VRPCDT is a new variant in the class of one-to-one PDPs (see Cordeau et al. [2008]), requiring a number of requests between distinct origin-destination pairs to be satisfied. Each node is either an origin, a destination, or a crossdock.

The VRPCD is a variant of the one-to-one PDP, studied by Wen et al. [2009] and Grangier et al. [2017]. Instead of shipping goods directly between origin-destination pairs as in the PDP, all shipments in the VRPCD are redirected to a crossdock for consolidation. This makes the VRPCD related to our problem, the main differences being that the VRPCD only considers one period and one crossdock, which makes synchronization at the crossdock an important aspect. This is not the case in our problem, as synchronization is simply enforced by the fact that pickups are performed in period 1, and deliveries in period 2. To synchronize operations at the crossdock, Wen et al. [2009] take into account whether or not goods have to switch vehicles, with corresponding unloading and reloading time per pallet. Lee et al. [2006] adopt a simpler approach to consider synchronization in their VRPCD, enforcing simultaneous arrivals of vehicles at the crossdocks, where vehicles arriving earlier have to wait. Their problem is not one-to-one though, making it fundamentally different from ours. Nikolopoulou et al. [2017] and Santos et al. [2013] study a generalization of the VRPCD, called PDP with Crossdocking (PDPCD). They relax the assumption that all pickups have to be consolidated at a crossdock before being delivered, by allowing direct shipments as well. Again, only one crossdock and one period are considered, whereas we study multiple crossdocks and two periods. A variant investigated by Qu and Bard [2012] and Masson et al. [2013] is the PDP with Transshipments (PDPT), which allows goods to be exchanged between vehicles at any designated transfer point in

the network, not necessarily a crossdock. As opposed to transfers between crossdocks, this problem requires two vehicles to be at the transfer point at the same time in order to exchange goods, rather than having the load wait for a vehicle at a crossdock. Although allowing direct shipments (PDPCD) and transshipments (PDPT) can yield substantial cost savings, they require the goods in the vehicle to be easily accessible during transportation. This may not always be possible, or bring additional handling costs (Veenstra et al. [2017a] and Veenstra et al. [2017b]). Our 2P-VRPCDT builds on the VRPCD, adding two periods, multiple crossdocks and transfers between those crossdocks.

Since the VRP and PDP are NP-hard problems, most papers studying variants of these problems present heuristics. To the best of our knowledge, only Santos et al. [2011] propose an exact method for the PDPCD. Using a a branch-and-price algorithm, they optimally solve instances with up to 10 requests. As the instance size increases, the optimality gap increases to 14.18% for instances with 50 requests. For the PDPT, Rais et al. [2014] propose an MIP, capable of optimally solving instances with up to 7 requests. Several heuristics for larger instances with time windows have been proposed for both these variants. Using iterated local search, Morais et al. [2014] study instances up to 500 requests for the VRPCD with time windows. Their results are improved by a metaheuristic based on large neighborhood search proposed by Grangier et al. [2017]. Using a greedy randomized adaptive search procedure with adaptive large neighborhood search, Qu and Bard [2012] give solutions for instances up to 50 requests of the PDPT with time windows. We develop an MIP including several sets of valid inequalities, capable of solving instances with 2 crossdocks and 12 requests to optimality. Moreover, we prove optimality for 148 best known heuristic solutions on instances for the related vehicle routing problem with crossdocking (VRPCD) from Nikolopoulou et al. [2017].

The remainder of this paper is organized as follows. In Section 2, we provide a formal description of the 2P-VRPCDT and formulate it as an MIP. The results of extensive computational experiments on new and benchmark instances are presented in Section 3 and the conclusions are given in Section 4.

2 Problem formulation

We study the 2P-VRPCDT with m crossdocks and n requests defined on a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{A}$ is the set of arcs. The node set $\mathcal{V}$ is partitioned into subsets $\mathcal{V}_+ = \{1, \ldots, n\}$ consisting of n origin nodes, $\mathcal{V}_- = \{n+1, \ldots, 2n\}$ with n destination nodes, and $\mathcal{V}_d = \{2n+1, \ldots, 2n+m\}$ representing m crossdocks. For ease of notation, let $\mathcal{V}_c$ denote the set of all origins and destinations, i.e., $\mathcal{V}_c = \mathcal{V}_+ \cup \mathcal{V}_-$ such that $\mathcal{V} = \mathcal{V}_d \cup \mathcal{V}_c$. A transportation request consists of a shipment from an origin node i to its corresponding destination node $i + n$.

For pickups, deliveries, and transfers, each crossdock has available an unlimited set $\mathcal{K}$ of homogeneous vehicles with capacity Q. Every origin node i has an associated load equal to q_i. For every destination node i, we have $q_i = -q_{i-n}$. Let c_{ij} denote the travel cost from node i to j. Transfers always occur as roundtrips between two crossdocks d and e, with corresponding costs $c_{de} + c_{ed}$. The travel time from node i to node j is denoted t_{ij} and the service time at node $i \in \mathcal{V}$ is given by s_i, where $s_i = 0, \forall i \in \mathcal{V}_d$. Finally, the

maximum driving time for a single route, equal to the sum of the travel and service times, is given by T. The objective of the 2P-VRPCDT is to minimize the total routing and transfer costs, here measured as time.

We formulate our MIP as follows. Let the binary variable x_{ij}^d be equal to 1 if a vehicle originating from crossdock d travels from i to j and zero otherwise. The binary variable y_i^d equals 1 if and only if node i is assigned to crossdock d. Integer variable z_{ij} is equal to the remaining load on the vehicle when traversing arc (i, j), i.e., after visiting i and before visiting j. Let v_i^d be an integer variable denoting the time at which a vehicle originating from crossdock d arrives at node i. Binary variable r_{ide}^k is equal to 1 if and only if load q_i originating from customer $i \in \mathcal{V}_+$ is assigned to a transfer from crossdock d to crossdock e on vehicle vehicle $k \in \mathcal{K}$. Binary variable $\tilde{r}_{de}^k$ indicates whether a transfer between crossdocks d and e with vehicle k is used in a solution. The 2P-VRPCDT can then be formulated as follows:

Minimize

$$\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{V}}\sum_{d\in\mathcal{V}_d} c_{ij}x_{ij}^d + \sum_{d\in\mathcal{V}_d}\sum_{e\in\mathcal{V}_d}\sum_{k\in\mathcal{K}} c_{de}\tilde{r}_{de}^k, \tag{1}$$

subject to

$$\sum_{d\in\mathcal{V}_d} y_i^d = 1, \qquad i \in \mathcal{V}_c \tag{2}$$

$$\sum_{j\in\mathcal{V}} x_{ij}^d + \sum_{j\in\mathcal{V}} x_{ji}^d = 2y_i^d, \qquad i \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{3}$$

$$\sum_{j\in\mathcal{V}_c} x_{dj}^d + \sum_{j\in\mathcal{V}_c} x_{jd}^d \geq 2y_d^d, \qquad d \in \mathcal{V}_d \tag{4}$$

$$\sum_{j\in\mathcal{V}} x_{ij}^d = \sum_{j\in\mathcal{V}} x_{ji}^d, \qquad i \in \mathcal{V}, d \in \mathcal{V}_d \tag{5}$$

$$y_i^d \leq y_d^d, \qquad i \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{6}$$

$$y_d^d \leq \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{V}} x_{ij}^d, \qquad d \in \mathcal{V}_d \tag{7}$$

$$x_{ie}^d = 0, \qquad i \in \mathcal{V}_c, d, e \in \mathcal{V}_d, d \neq e \tag{8}$$

$$x_{ei}^d = 0, \qquad d, e \in \mathcal{V}_d, i \in \mathcal{V}_c, d \neq e \tag{9}$$

$$\sum_{i\in\mathcal{V}_+}\sum_{d\in\mathcal{V}_d} z_{id} = \sum_{i\in\mathcal{V}_+} q_i, \tag{10}$$

$$\sum_{d\in\mathcal{V}_d}\sum_{i\in\mathcal{V}_-} z_{di} = \sum_{i\in\mathcal{V}_+} q_i, \tag{11}$$

$$\sum_{j\in\mathcal{V}} z_{ij} - \sum_{j\in\mathcal{V}} z_{ji} = q_i, \qquad i \in \mathcal{V}_c \tag{12}$$

$$z_{ij} \leq (Q - q_j)\sum_{d\in\mathcal{V}_d} x_{ij}^d, \qquad i, j \in \mathcal{V}_+, i \neq j \tag{13}$$

$$z_{ij} \leq (Q + q_i)\sum_{d\in\mathcal{V}_d} x_{ij}^d, \qquad i, j \in \mathcal{V}_-, i \neq j \tag{14}$$

$$z_{id} \leq Q x^d_{id}, \qquad i \in \mathcal{V}_+, d \in \mathcal{V}_d \tag{15}$$
$$z_{di} \leq Q x^d_{di}, \qquad d \in \mathcal{V}_d, i \in \mathcal{V}_- \tag{16}$$
$$x^d_{ij} = 0, \qquad i \in \mathcal{V}_+, j \in \mathcal{V}_-, d \in \mathcal{V}_d \tag{17}$$
$$x^d_{ji} = 0, \qquad j \in \mathcal{V}_-, i \in \mathcal{V}_+, d \in \mathcal{V}_d \tag{18}$$
$$z_{ij} = 0, \qquad i \in \mathcal{V}_+, j \in \mathcal{V}_- \tag{19}$$
$$z_{ji} = 0, \qquad j \in \mathcal{V}_-, i \in \mathcal{V}_+ \tag{20}$$
$$v^d_d = 0, \qquad d \in \mathcal{V}_d \tag{21}$$
$$v^d_j \geq (v^d_i + s_i + t_{ij}) x^d_{ij}, \qquad j \in \mathcal{V}_c, i \in \mathcal{V}, d \in \mathcal{V}_d \tag{22}$$
$$v^d_i + (s_i + t_{id}) x^d_{id} \leq T y^d_i, \qquad i \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{23}$$
$$y^d_i \cdot y^e_{i+n} - \sum_{k \in \mathcal{K}} r^k_{ide} = 0, \qquad i \in \mathcal{V}_+, d, e \in \mathcal{V}_d, d \neq e \tag{24}$$
$$\tilde{r}^k_{de} = \tilde{r}^k_{ed}, \qquad d, e \in \mathcal{V}_d, k \in \mathcal{K} \tag{25}$$
$$\sum_{i \in \mathcal{V}_+} q_i r^k_{ide} \leq Q \tilde{r}^k_{de}, \qquad d, e \in \mathcal{V}_d, k \in \mathcal{K} \tag{26}$$
$$x^d_{ij} \in \{0, 1\}, \qquad i, j \in \mathcal{V}, d \in \mathcal{V}_d \tag{27}$$
$$y^d_i \in \{0, 1\}, \qquad i \in \mathcal{V}, d \in \mathcal{V}_d \tag{28}$$
$$z_{ij} \geq 0, \qquad i, j \in \mathcal{V} \tag{29}$$
$$v^d_i \geq 0, \qquad i \in \mathcal{V}, d \in \mathcal{V}_d \tag{30}$$
$$r^k_{ide} \in \{0, 1\}, \qquad i \in \mathcal{V}_+, d, e, \in \mathcal{V}_d, k \in \mathcal{K} \tag{31}$$
$$\tilde{r}^k_{de} \in \{0, 1\}, \qquad d, e \in \mathcal{V}_d, k \in \mathcal{K}. \tag{32}$$

The objective (1) is to minimize the total routing costs, consisting of two terms: the pickup and delivery costs and the transfer costs. Constraints (2)–(16) are derived from formulations F1 and F3 of Lahyani et al.. Since they consider a vehicle routing problem without pickups and deliveries, the constraints have been adjusted to fit our problem with origin-destination pairs. Constraints (2) ensure that all customers are visited exactly once. Constraints (3) and (4) are the linking constraints and constraints (5) define flow conservation. Constraints (6) and (7) imply that customers can only be served by vehicles that are used and constraints (8) and (9) guarantee that vehicles leave from and return to the same crossdock. Constraints (10) and (11) enforce the correct load to return to the crossdocks in period 1 and to leave the crossdocks in period 2, respectively, and constraints (12) ensure that the change in vehicle load at a node is consistent. Constraints (13) and (14) link these loads to arcs and provide upper bounds for the load on each arc for pickups and deliveries, respectively. Constraints (15) and (16) restrict the load on arcs leaving and entering the crossdock so that the vehicle capacity is never exceeded. Simultaneously, these constraints guarantee that if there is no arc from or to the crossdock, there is no load on this arc either. Constraints (17) and (18) eliminate arcs between origins and destinations and constraints (19) and (20) ensure that there is no load on those connections. Constraints (21)–(23) are used to enforce a limit on the route duration, where constraints (21) state that all routes start from a crossdock at time zero. Constraints (22) guarantee that arrival times at all customers are consistent and

constraints (23) ensure that the maximum driving time is not exceeded. Constraints (24) assign a transfer to every request that is delivered from a different crossdock than the one the goods from the origin were shipped to. Constraints (25) enforce roundtrips between crossdocks, whereas constraints (26) ensure that the vehicle capacity in both directions is not exceeded. Finally, constraints (27)–(32) define the nature of the decision variables.

2.1 Linearizing the MIP

Constraints (22) are currently not linear, as it contains the product of the integer variable v_i^d and binary variable x_{ij}^d. In order to linearize this, we define an additional integer variable w_{ij}^d, and replace constraints (22) by the following:

$$w_{ij}^d \leq T x_{ij}^d, \qquad i \in \mathcal{V}, j \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{33}$$
$$w_{ij}^d \leq v_i^d, \qquad i \in \mathcal{V}, j \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{34}$$
$$w_{ij}^d \geq v_i^d - (1 - x_{ij}^d)T, \qquad i \in \mathcal{V}, j \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{35}$$
$$w_{ij}^d \geq 0, \qquad i \in \mathcal{V}, j \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{36}$$
$$v_j^d \geq w_{ij}^d + (s_i + t_{ij})x_{ij}^d, \quad i \in \mathcal{V}, j \in \mathcal{V}_c, d \in \mathcal{V}_d. \tag{37}$$

Constraints (33)–(36) ensure that w_{ij}^d contains the product of v_i^d and x_{ij}^d, where T is used as a *Big M*. Constraints (37) are then the linear version of constraints (22).

Since constraints (24) contain the product of the two binary variables y_i^d and y_{i+n}^e, they are also nonlinear. We linearize this by defining the binary variable p_i^d, and replacing constraints (24) with the following:

$$p_i^{de} \leq y_i^d, \qquad i \in \mathcal{V}_+, d, e \in \mathcal{V}_d, d \neq e \tag{38}$$
$$p_i^{de} \leq y_{i+n}^e, \qquad i \in \mathcal{V}_+, d, e \in \mathcal{V}_d, d \neq e \tag{39}$$
$$p_i^{de} \geq y_i^d + y_{i+n}^e - 1, \quad i \in \mathcal{V}_+, d, e \in \mathcal{V}_d, d \neq e \tag{40}$$
$$p_i^{de} - \sum_{k \in \mathcal{K}} r_{ide}^k = 0, \qquad i \in \mathcal{V}_+, d, e \in \mathcal{V}_d, d \neq e. \tag{41}$$

Constraints (38)–(40) ensure that p_i^{de} contains the product of y_i^d and y_{i+n}^{de} and constraints (41) are the linear version of constraints (24).

2.2 Improved constraints

Although the nonlinear constraints (22) can be straightforwardly linearized as shown in Section 2.1, we present here an alternative linear formulation. In Section 3, we will show that this alternative formulation is more effective than the straightforward linearization. This gives

$$v_j^d \geq v_i^d + (s_i + t_{ij})x_{ij}^d - T(1 - x_{ij}^d), \qquad j \in \mathcal{V}_c, i \in \mathcal{V}, d \in \mathcal{V}_d. \tag{42}$$

For nonlinear constraints (24), it is also better to rewrite than to linearize, yielding

$$y_i^d + y_{i+n}^e - \sum_{k \in \mathcal{K}} r_{ide}^k \leq 1, \qquad i \in \mathcal{V}_+, d, e \in \mathcal{V}_d, d \neq e. \tag{43}$$

Although constraints (43) correctly assign transfers if goods are delivered from different crossdocks, they do not exclude the assignment of transfers when they are delivered from the same crossdock. Thus, we introduce constraints (44) that forbid the use of transfers for goods that are handled by vehicles from the same crossdock in both periods.

$$2\sum_{k\in\mathcal{K}} r^k_{ide} \leq y^d_i + y^e_{i+n}, \qquad i \in \mathcal{V}_+, d, e, \in \mathcal{V}_d, d \neq e. \tag{44}$$

2.3 Valid inequalities

In order to speed up computations, we have created several sets of valid inequalities, given below. In Section 3, we will show that they significantly improve the solution bounds and decrease runtimes.

$$\sum_{j\in\mathcal{V}}\sum_{d\in\mathcal{V}_d} x^d_{ij} = 1, \qquad i \in \mathcal{V}_c \tag{45}$$

$$\sum_{j\in\mathcal{V}}\sum_{d\in\mathcal{V}_d} x^d_{ji} = 1, \qquad i \in \mathcal{V}_c \tag{46}$$

$$z_{di} = 0, \qquad d \in \mathcal{V}_d, i \in \mathcal{V}_+ \tag{47}$$

$$z_{id} = 0, \qquad i \in \mathcal{V}_-, d \in \mathcal{V}_d \tag{48}$$

$$z_{ij} \geq q_i \sum_{d\in\mathcal{V}_d} x^d_{ij}, \qquad i, j \in \mathcal{V}_+ \tag{49}$$

$$z_{ij} \geq -q_j \sum_{d\in\mathcal{V}_d} x^d_{ij}, \qquad i, j \in \mathcal{V}_- \tag{50}$$

$$x^d_{ef} = 0, \qquad e, f, d \in \mathcal{V}_d \tag{51}$$

$$x^d_{ii} = 0, \qquad i \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{52}$$

$$z_{de} = 0, \qquad d, e \in \mathcal{V}_d \tag{53}$$

$$z_{ii} = 0, \qquad i \in \mathcal{V}_c \tag{54}$$

$$x^d_{id} \leq y^d_i, \qquad i \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{55}$$

$$x^d_{di} \leq y^d_i, \qquad d \in \mathcal{V}_d, i \in \mathcal{V}_c \tag{56}$$

$$x^d_{ij} + y^d_i + \sum_{e\in\mathcal{V}_d, e\neq d} y^e_j \leq 2, \qquad i, j \in \mathcal{V}_c, d \in \mathcal{V}_d, i \neq j \tag{57}$$

$$\tilde{r}^k_{dd} = 0, \qquad d \in \mathcal{V}_d, k \in \mathcal{K} \tag{58}$$

$$y^e_d = 0, \qquad d, e \in \mathcal{V}_d, d \neq e \tag{59}$$

$$v^d_i \leq y^d_i(T - t_{id} - s_i), \qquad i \in \mathcal{V}_c, d \in \mathcal{V}_d \tag{60}$$

$$\sum_{d\in\mathcal{V}_d}\sum_{e\in\mathcal{V}_d} \tilde{r}^k_{de} \leq \sum_{d\in\mathcal{V}_d}\sum_{e\in\mathcal{V}_d} \tilde{r}^{k-1}_{de}, \qquad k \in \mathcal{K} \setminus \{1\} \tag{61}$$

$$\sum_{d\in\mathcal{V}_d}\sum_{e\in\mathcal{V}_d} r^{dek}_i \leq \sum_{d\in\mathcal{V}_d}\sum_{e\in\mathcal{V}_d}\sum_{j\in\mathcal{V}_+, j<i} r^{de,k-1}_j, \quad i \in \mathcal{V}_+, k \in \mathcal{K} \setminus \{1\}. \tag{62}$$

Constraints (45)–(57) are derived from the valid inequalities proposed by Lahyani et al. and have been adapted to fit our pickup and delivery setting. Constraints (45) and (46)

extend flow conservation constraints (3)–(5) by stating that the total flow coming into and going out of each node is equal to one. Constraints (47) and (48) ensure that a vehicle leaves the crossdock empty in period 1 and returns to the crossdock empty in period 2, respectively, and constraints (49) and (50) impose lower bounds for the load on each arc for pickups and deliveries. Constraints (51) and (52) prohibit any arcs between two crossdocks and from a node to itself, respectively, while constraints (53) and (54) forbid any loads on those respective connections. Constraints (55) and (56) ensure that there can only be arcs between a customer and a crossdock it is assigned to. Constraints (57) ensure that there are no arcs between customers that are assigned to different crossdocks. Constraints (58) forbid transfers between a crossdock and iteself. Due to constraints (59), crossdocks cannot be assigned to a different crossdock than itself. Constraints (60) impose an upper bound on the arrival time and constraints (61) and (62) are symmetry breaking constraints based on Coelho and Laporte [2014]. These ensure that lowest indexed vehicles are used first, and lowest indexed customers are assigned to low indexed vehicles.

2.4 Problem without transfers

To establish the effect of transfers between crossdocks, we also need to obtain solutions that do not allow for them. Using some minor modifications, the model previously defined can be restricted to disallow transfers. In order to this, we add the following additional constraints to enforce pickups and deliveries to be handled by the same crossdock:

$$y_i^d = y_{i+n}^d, \qquad i \in \mathcal{V}_+, d \in \mathcal{V}_d. \tag{63}$$

Moreover, all constraints involving transfers, i.e., (24)–(26), (31), (32), (58), (61), and (62) are removed from the model, as well as the transfer costs (second term) in the objective (1). Thus, the MIP without transfers is given by:

Minimize

$$\sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} \sum_{d \in \mathcal{V}_d} c_{ij} x_{ij}^d, \tag{64}$$

subject to: (2)–(21), (23), (27)–(30), (42), (45)–(57), (59), (60), and (63).

3 Computational experiments

In this section, we evaluate the effect of valid inequalities, compare solutions for three different problem variants, and apply our model to VRPCD instances from Nikolopoulou et al. [2017]. For most of these experiments we use our own instances, which include all characteristics of the problem. Known instances from related literature such as Nikolopoulou et al. [2017] contain only one crossdock and one period and thus have a too limited scope. Transforming VRP or PDP instances to 2P-VRPCDT instances would change the instances to such a large extent that it would essentially generate completely new instances as well. We do include a comparison to the VRPCD instances created by Nikolopoulou et al. [2017] in Section 3.5.

Our MIP has been implemented into IBM CPLEX in C++ and compiled with GCC 5.1.0. Up to eight threads can be used of an Intel Xeon E5 2680v3 CPU at 2.5GHz. We have used a time limit of three hours and a maximum of 48Gb memory for all instances. The results obtained in Section 3.3 use at most eight threads of an Intel Core i5-3470 CPU at 3.2GHz, again with a time limit of three hours and maximum memory of 48Gb.

3.1 Experimental setup

In order to create new instances, we based our approach on that of Nikolopoulou et al. [2017]. The transportation loads q_i, $i \in \mathcal{V}_+$ are drawn uniformly from $[1, 30]$ and all nodes $i \in \mathcal{V}_c$ have a fixed service time equal to two time units. Travel costs and time between two nodes are equal to their Euclidean distance, thus the graph $\mathcal{G}$ is symmetric. Vehicle capacities are $Q \in \{70, 100, 120\}$ and the maximum driving time is given by $T = 250$, or about half a working day in minutes. This allows for pickups and deliveries to be effectively performed on the same day, and the concept of periods in our model then refer to an early and a late shift.

Every instance consists of a number $n \in \{12, 14, 16, 18\}$ of one-to-one requests between origin-destination pairs. This gives instances of 24, 28, 32, and 36 nodes, which are distributed on a $[0, 100]^2$ grid. Here we study settings with two crossdocks, which is the most compact form to demonstrate effects of having multiple crossdocks. A setting with three crossdocks is presented in Section 3.4. We use three different location pairs on the $[0, 100]^2$ grid for the two crossdocks, denoted D1, D2, and D3, with increasing distance between them. For all three location pairs, one of the crossdocks is located in the eastern part of the grid, $x_{2n+1} = 20$, while the other one is located in the western part, $x_{2n+2} = 80$. For the y-coordinates, we have $y_{2n+1} \in \{50, 70, 90\}$ and $y_{2n+2} = 100 - y_{2n+1}$. We consider four classes of clustering for origins and destinations, namely: random (R), semi-clustered (SC), clustered (C), and fully clustered (FC). For random instances, all nodes are uniformly distributed on the $[0, 100]^2$ grid. For semi-clustered instances, the y-coordinates $y_i, i \in \mathcal{V}_c$ are still drawn uniformly from $[0, 100]$. For x-coordinates however, we have the following:

- $x_i \sim \mathcal{U}[0, 60]$, $i \in \{1, 2, \ldots, 0.5n\}$,
- $x_i \sim \mathcal{U}[40, 100]$, $i \in \{0.5n + 1, 0.5n + 2, \ldots n\}$,
- $x_i \sim \mathcal{U}[40, 100]$, $i \in \{n + 1, n + 2, \ldots 1.5n\}$,
- $x_i \sim \mathcal{U}[0, 60]$, $i \in \{1.5n + 1, 1.5n + 2, \ldots, 2n\}$.

Thus, half the origins are located mainly in the western part of the grid, while their corresponding destinations are mainly in the eastern part and there is some overlap in the middle. For the other half of the origins and destinations, it is the other way around. For the clustered instances, we have the following:

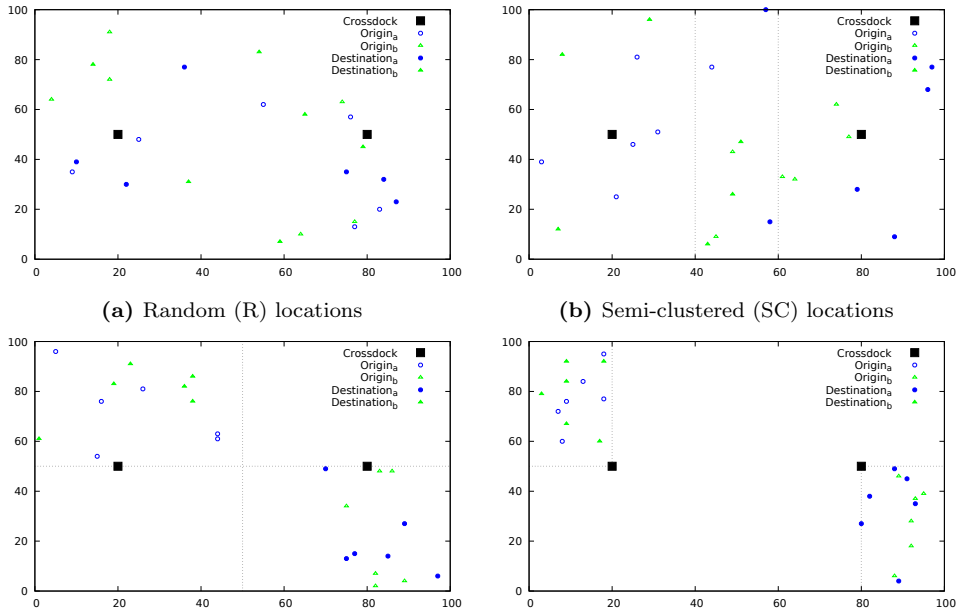
- $x_i \sim \mathcal{U}[0, 50]$, $y_i \sim \mathcal{U}[51, 100]$, $i \in \{1, 2, \ldots, 0.5n\}$,
- $x_i \sim \mathcal{U}[51, 100]$, $y_i \sim \mathcal{U}[0, 50]$, $i \in \{0.5n + 1, 0.5n + 2, \ldots n\}$,

- $x_i \sim \mathcal{U}[51, 100]$, $y_i \sim \mathcal{U}[0, 50]$, $i \in \{n+1, n+2, \ldots 1.5n\}$,
- $x_i \sim \mathcal{U}[0, 50]$, $y_i \sim \mathcal{U}[51, 100]$, $i \in \{1.5n+1, 1.5n+2, \ldots, 2n\}$.

Half the origins are located in the north-west, while their corresponding destinations are located in the south-east. For the other half of origins and destinations, it is the other way around. For the fully clustered instances, the origin and destination locations depend on the crossdock locations. Recall that the x- and y-coordinates of the crossdocks for an instance with two crossdocks are denoted (x_{2n+1}, y_{2n+1}) and (x_{2n+2}, y_{2n+2}), respectively. Then we have the following:

- $x_i \sim \mathcal{U}[0, x_{2n+1}]$, $y_i \sim \mathcal{U}[y_{2n+1}, 100]$, $i \in \{1, 2, \ldots, 0.5n\}$,
- $x_i \sim \mathcal{U}[x_{2n+2}, 100]$, $y_i \sim \mathcal{U}[0, y_{2n+2}]$, $i \in \{0.5n+1, 0.5n+2, \ldots n\}$,
- $x_i \sim \mathcal{U}[x_{2n+2}, 100]$, $y_i \sim \mathcal{U}[0, y_{2n+2}$, $i \in \{n+1, n+2, \ldots 1.5n\}$,
- $x_i \sim \mathcal{U}[0, x_{2n+1}]$, $y_i \sim \mathcal{U}[y_{2n+1}, 100]$, $i \in \{1.5n+1, 1.5n+2, \ldots, 2n\}$.

Thus, half the origins are located north-west from the western crossdock, while their corresponding destinations are located south-east of the eastern crossdock. For the other half of origins and destinations, it is the other way around. Figure 3 shows an example of each clustering class.



(a) Random (R) locations

(b) Semi-clustered (SC) locations

(c) Clustered (C) locations

(d) Fully clustered (FC) locations

Figure 3: Example instance for each of the four clustering classes.

Following the above, we have generated three different instances for each combination of number of requests and clustering class, labeled *a-c*. The dispersion of nodes in the graph is kept fixed for variations in vehicle capacity Q, in order to assess its effect. Hence, we have $4 \cdot 4 \cdot 3 \cdot 3$ different node distributions, each tested for 3 different vehicle capacities, yielding a total of 432 instances.

3.2 Performance of valid inequalities

In order to assess the added value of our valid inequalities, we distinguish between the following three different sets of constraints:

- VI0: no valid inequalities, constraints (2)–(21), (23), (25)–(41)
- VI1: all valid inequalities, constraints (2)–(21), (23), (25)–(41), (45)–(62)
- VI2: all valid inequalities and improved constraints: (2)–(21), (23), (25)–(32), (42)–(62)

We have tested a number of other valid inequalities that did not lead to improvements in our preliminary tests. Hence, all valid inequalities mentioned here have shown to improve the performance of the solution process.

For all instances, all three constraint sets obtained feasible solutions well within three hours. Table 1 shows the total number of instances for which the different constraint sets obtained optimal solutions within the time limit. As expected, formulation VI0 was the least successful, while VI2 solved the most instances to optimality, indicating that our valid inequalities do indeed have a positive effect. Constraint set VI1 obtained about 25% more optimal solutions than VI0, while constraint set VI2 obtained about 43% more optimal solutions.

Table 1: Number of optimal solutions obtained by each constraint set.

Constraint set	Number of optimal solutions
VI0	197
VI1	246
VI2	281

A total of 197 instances have been solved to optimality by all three constraint sets. The average runtimes for these instances are given in Table 2 for each constraint set. The results indicate that valid inequalities have a strong, positive impact on runtimes. Constraint set VI1 has an average runtime of 772 seconds, which is about 62% faster than without valid inequalities (VI0). The rewritten constraints (VI2) make the model another 53% faster, giving a total speedup of more than 82%. If we distinguish between solutions using transfers and solutions without them, we see a similar result. The average time saving from VI0 to VI1 for instances with transfers is about 65%, compared to 60% for instances without transfers. The time saving from VI1 to VI2 for instances with transfers is about 52%, compared to 54% for instances without transfers. Hence, the time saving for instances where transfers are used and for instances where they are not used seems to be more or less equal.

In addition to the time savings as shown in Table 2, there are many instances that cannot be solved to optimality by one or more constraint sets, because either the time limit or the memory limit is reached. The obtained average lower bounds, upper bounds, gaps, and runtimes for these instances are presented in Table 3. The pattern between constraint sets is the same. Constraint set VI2 performs best, while VI0 does worst. On

Table 2: Average runtime in seconds for 197 instances solved optimally by all three constraint sets.

	VI0	VI1	VI2
Instances where transfers are used	2,040	817	376
Instances where transfers are not used	2,029	718	348
All instances	2,035	772	363

average, the lower bounds are increasing and the upper bounds decreasing in the number of valid inequalities. The average gap decreases by about 42% from VI0 to VI1, and another 27% from VI1 to VI2. In total, this means that the average gap using VI2 is more than 58% smaller compared to VI0. Moreover, VI2 obtains these results about 14% faster than VI1 and about 30% faster than VI0. Thus, VI2 gives better results in a smaller amount of time.

Table 3: Average bounds for 235 instances not solved to optimality by all three constraint sets.

	LB	UB	Gap (%)	Runtime (s)
VI0	860.66	982.80	12.90	10,475
VI1	902.87	974.04	7.46	9,016
VI2	919.33	972.18	5.47	7,343

To better understand what kind of instances are harder to solve, Table 4 shows the number of instances solved to optimality by all three constraint sets for different instance classes. The size of the instances has the biggest influence. For instances with 24 nodes (12 requests), 101 out of 108 instances have been solved to optimality for all constraint sets. For 14 requests, this number has already decreased to 74 instances, and only 5 instances with 18 requests can be solved optimally by all constraint sets. Other than instance size, the crossdock locations seem to have a large influence on the difficulty to solve instances. If crossdocks are located centrally (D1), 77 instances can be solved to optimality, while only 50 of the instances where crossdocks are farthest apart (D3) can be solved to optimality. The effect of different types of clustering and changes in vehicle capacity appears to be smaller.

Table 4: Number of instances solved optimally by all constraint sets for different parameter settings.

Instance size	# Optimal	Clustering	# Optimal	Crossdocks	# Optimal	Capacity	# Optimal
24	101	R	53	D1	77	70	59
28	74	SC	55	D2	70	100	71
32	17	C	48	D3	50	120	67
36	5	FC	41				

3.3 Effects of transfers and joint scheduling

Here we study the effects on efficiency from jointly scheduling routes and from the transfers between crossdocks. All instances have been solved using all relevant valid inequalities and rewritten constraints (VI2), for the following three problem variants:

- Problem IN: independently scheduled routes. All origin-destination pairs are assigned to a crossdock based on the shortest distance between origin and crossdock. Consequently, routes are constructed independently for each crossdock. Hence, after assignment of origin-destination pairs, a two-period VRPCD is solved for each crossdock.

- Problem J0: jointly scheduled routes without transfers. Origin-destination pairs are not assigned to crossdocks before scheduling routes, rather routing and crossdock assignments are jointly optimized. Transfers are not allowed.

- Problem J1: jointly scheduled routes with transfers. This adds transfers to the jointly scheduled routes without transfers, which gives the 2P-VRPCDT.

Table 5 shows the number of optimal solutions obtained per problem variant. For problem IN, 429 out of 432 instances are solved optimally. For problem J0, about 50% of the optimal solutions is obtained, with a total of 208. For problem J1, more than 25% additional optimal solutions are found and all 108 instances with 24 nodes (12 requests) are solved optimally. For the largest instances with 18 requests, problem J0 and problem J1 both obtain only 15 optimal solutions. Problem IN is easier to solve, since origin-destination pairs are assigned to crossdocks first. After that, a much smaller problem has to be solved for each crossdock. The reason that problem J1 obtains more optimal solutions than problem J0 is most likely that pruning of the search tree can be done more efficiently. Most of the additional solutions obtained by problem J1 use transfers, such that solutions that do not use transfers have higher costs and can quickly be eliminated, decreasing the runtime.

Table 5: Number of optimal solutions for each problem variant.

	Number of nodes				
Problem variant	24	28	32	36	Total
Problem IN	108	108	108	105	429
Problem J0	85	71	37	15	208
Problem J1	108	87	51	15	261

Out of the 432 instances, 195 have been solved optimally for all three problem variants. Table 6 shows results for each problem variant: the average number of vehicles used, average costs, and average runtime, respectively. On average, the problem IN solutions use almost one vehicle more per instance than the problem J0 and the problem J1 solutions. Moreover, the average costs for problem J0 solutions are about 12.5% lower than those for problem IN solutions, while problem J1 solutions reduce average costs by another 2.4%, giving a total cost saving of about 14.6%.

In order to compare differences between different types of clustering and vehicle capacities, we define the following indicators:

- ΔC_{IN-J0} = % cost difference between the problem with independently scheduled routes and the problem with jointly scheduled routes without transfers

Table 6: Comparison between different problem variants for 195 optimally solved instances.

Problem variant	# Vehicles	Costs	Runtime (s)
Problem IN	6.9	990.6	2.4
Problem J0	6.0	866.7	1916.0
Problem J1	6.0	845.6	1315.9

- ΔC_{J0-J1} = % cost difference between the problem with jointly scheduled routes without transfers and the problem with jointly scheduled routes with transfers
- ΔC_{IN-J1} = % cost difference between the problem with independently scheduled routes and the problem with jointly scheduled routes with transfers
- ΔV_{IN-J0} = % difference in vehicles used between the problem with independently scheduled routes and the problem with jointly scheduled routes without transfers
- ΔV_{IN-J1} = % difference in vehicles used between the problem with independently scheduled routes and the problem with jointly scheduled routes with transfers
- Δ_{joint} = % of the instances where the costs for the problem with jointly scheduled routes without transfers are lower than the costs for the problem with independently scheduled routes
- Δ_{trans} = % of the instances where the costs for the problem with jointly scheduled routes with transfers are lower than the costs for the problem with jointly scheduled routes without transfers, i.e., transfers are used.

Note that any feasible solution for problem IN is feasible for problem J0, and any feasible solution for problem J0 is feasible for problem J1. Hence, the optimal costs for problem J1 are at most as high as the optimal costs for problem J0, which are in turn at most as high as the optimal costs for problem IN. From Table 7, it follows that the costs for problem J0 are lower than the costs for problem IN (Δ_{joint}) in almost 95% of the instances. For the other 5% instances, the objectives are equal. Moreover, the costs for J1 are lower than those of problem J0 (Δ_{trans}) in about 34% of the instances and equal for the other 66% instances. This 34% is likely underestimated though, since we have obtained fewer FC and C solutions, which use a lot more transfers than SC and R instances, namely 100% and 67.57%, as opposed to 7.35% and 18.18%, respectively. The cost difference between problem IN and problem J0 (ΔC_{IN-J0}) is the largest for SC instances, where costs are reduced by almost 16% and almost 20% fewer vehicles are needed (ΔV_{IN-J0}). This is no coincidence, since the clusters of these instances overlap, making jointly scheduling routes more beneficial and reducing the likelihood of transfers. For the FC instances, the number of vehicles used among different solutions is similar. This is most likely because these instances are so strongly clustered that little can be gained from including origins and destinations from different clusters in one route. Hence, routes are similar for the three different problem variants. For all other clustering types, problem J0 and problem J1 use fewer vehicles than problem IN, because routes can be made more efficient by including origins and destination from different geographical areas into one route.

Table 7: Savings between different types of clustering (in %).

Type	# Instances	ΔC_{IN-J0}	ΔC_{J0-J1}	ΔC_{IN-J1}	ΔV_{IN-J0}	ΔV_{IN-J1}	Δ_{joint}	Δ_{trans}
FC	24	1.54	16.06	17.36	1.59	0.00	79.17	100.00
C	37	8.04	3.24	11.02	10.13	11.45	94.59	67.57
SC	68	15.96	0.18	16.12	19.47	19.07	100.00	7.35
R	66	13.62	0.42	13.99	9.78	10.39	93.93	18.18
All	195	12.51	2.43	14.64	12.64	12.72	94.36	33.85

The fraction of instances where transfers are used (Δ_{trans}) seems a bit small for SC and R instances, which can be explained by the instance size. Most of our solutions are obtained for small instances of 24 and 28 nodes. Especially for R and SC instances, there is often simply no need for transfers yet, because only a few origin-destination pairs are far apart. Table 8 shows the percentage of R and SC instances using transfers for small (24 & 28 nodes) and large (32 & 36 nodes) instances. For the latter, we have only 33 results, but these indicate a strong increase in transfers with instance size, using more than four times as many transfers for both clustering classes. Hence, transfers have the potential of further increasing cost savings on larger instances.

Table 8: Fraction of instances using transfers in the SC and R class (in %).

Instance size	SC	R
Small (24, 28)	4.00	10.20
Large (32, 36)	18.75	41.18

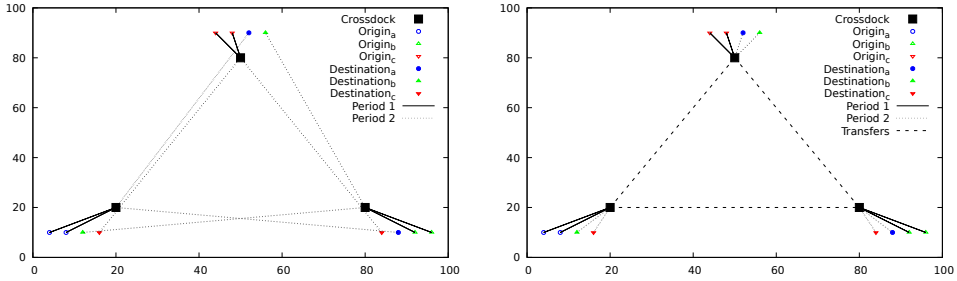
Table 9 shows that an increase in vehicle capacity from 70 to 120 increases the cost difference between problem IN and problem J0 (ΔC_{IN-J0}) from 11.39% to 16.37%, and the difference in vehicles needed (ΔV_{IN-J0}) from 10.88% to 20.92%. On the other hand, it leads to a decrease in cost difference between problem J0 and problem J1 (ΔC_{J0-J1}), from 1.66% to 0.51%. This is logical, because locations far from a crossdock can be included in routes more easily if the vehicle capacity increases, reducing the need for transfers. Note that the maximum route duration is usually not an issue, such that the vehicle capacity is the limiting factor. Even with the largest vehicle capacity though, transfers are used in almost 15% of the instances (Δ_{trans}). Note that results from Table 9 mostly include small R and SC instances, since these are easiest to solve by all problem variants. Therefore, savings are underestimated again.

Table 9: Comparison between vehicle capacities for 41 instances with identical node locations.

Capacity	ΔC_{IN-J0}	ΔC_{J0-J1}	ΔC_{IN-J1}	ΔV_{IN-J0}	ΔV_{IN-J1}	Δ_{joint}	Δ_{trans}
70	11.39	1.66	12.85	10.88	10.27	97.56	34.15
100	14.23	0.52	14.68	13.73	14.12	97.56	19.51
120	16.37	0.51	16.80	20.92	21.34	97.56	14.63

3.4 More than two crossdocks

Transportation networks can consist of more than two crossdocks. Figure 4 gives an example of an instance where it is clear that transfers reduce costs. The loads to be transported are all equal to the vehicle capacity. For problem IN, six vehicles have to drive long distances back empty after making deliveries. However, for problem J1 only three transfers are needed to exchange these goods between crossdocks, since they can take goods in both directions. The routes to destinations are then very short, leading to a cost reduction of 33.09%, from 1,076 to 720.



(a) Problem IN **(b)** Problem J1

Figure 4: Example instance with three crossdocks.

3.5 Comparison with VRPCD

Currently, there are no relevant benchmarks available from literature that are suitable for determining the solution quality of our 2P-VRPCDT. In the VRPCD studied by Nikolopoulou et al. [2017], vehicles taking care of deliveries can only leave after the goods to be delivered are available at the crossdock, i.e., all vehicles that picked up those goods have to be back at the crossdock and unloaded. We extended our MIP to include this aspect. However, while this type of synchronization is manageable with a heuristic, it significantly increases the difficulty to solve the MIP and it appeared to be inhibitive for finding good solutions. Therefore, we obtained solutions without taking synchronization at the crossdock into account and checked whether or not these solutions are still feasible if synchronization is considered. Using this method, we either obtain an optimal solution, or no feasible solution at all. A summary of the results can be found in Table 10. For 148 of the 243 instances with 12 requests, we match the best known heuristic solution from Nikolopoulou et al. [2017] and hence prove its optimality. For the other 95 solutions, we do not obtain a feasible solution. All solutions are obtained within 12 seconds, with an average runtime of 2.3 seconds.

Table 10: Results for the Nikolopoulou instances (time in seconds).

	# Instances	T_{ave}	T_{max}
Total	243	2.3	12
Optimal	148	1.9	5
Infeasible	95	2.8	12

4 Conclusion

In this paper, we introduced the 2P-VRPCDT and formulated it as an MIP. We proposed several sets of valid inequalities, yielding 43% more optimal solutions than without them within the given time limit. For instances that could be solved optimally without valid inequalities, they decrease the runtime by more than 82%. Using all valid inequalities, our model is capable of solving all 108 instances with 2 crossdocks and 12 requests optimally. For larger instances, this number decreases to 15 out of 108 solved instances for instances with 18 requests. In addition, we have proven optimality for 148 of 243 benchmark instances for the VRPCD.

By jointly scheduling routes rather than independently, the average cost savings are 12.5%. Transfers further increase these savings to a total of 14.6%. Moreover, the number of vehicles needed is reduced by almost 13%. For almost 95% of the instances, jointly scheduling routes reduces costs, and transfers reduce costs in almost 34% of the instances. Moreover, our results indicate that both jointly scheduling routes and allowing for transfers reduce costs even more for larger instances.

Since the 2P-VRPCDT is an NP-hard problem, only small instances with a limited number of requests can be solved optimally using our MIP. An interesting area for further research is to develop a heuristic, allowing larger instances to be solved. If heuristics are considered, another possibility would be to allow for direct-shipments as well, without an intermediate stop at a crossdock.

References

G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte. Static pickup and delivery problems: a classification scheme and survey. *TOP*, 15(1):1–31, 2007.

K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99:300–313, 2016.

P. Buijs, J.A.L. Alvarez, M. Veenstra, and K.J. Roodbergen. Improved collaborative transport planning at dutch logistics service provider Fritom. *Interfaces*, 46(2):119–132, 2016.

P. Chen, Y. Guo, A. Lim, and B. Rodrigues. Multiple crossdocks with inventory and time windows. *Computers & Operations Research*, 33(1):43–63, 2006.

L.C. Coelho and G. Laporte. Improved solutions for inventory-routing problems through valid inequalities and input ordering. *International Journal of Production Economics*, 155:391–397, 2014.

J.-F. Cordeau, G. Laporte, and S. Røpke. Recent models and algorithms for one-to-one pickup and delivery problems. In B.L. Golden, S. Raghavan, and E.A. Wasil, editors, *The Vehicle Routing Problem*, pages 327–357. Springer, 2008.

P. Grangier, M. Gendreau, F. Lehuédé, and L.-M. Rousseau. A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking. *Computers & Operations Research*, 84:116–126, 2017.

R. Lahyani, L.C. Coelho, and J. Renaud. Alternative formulations and improved bounds for the multi-depot fleet size and mix vehicle routing problem. Technical report CIRRELT-2015-36, Montréal 2015.

Y.H Lee, J.W. Jung, and K.M. Lee. Vehicle routing scheduling for cross-docking in the supply chain. *Computers & Industrial Engineering*, 51(2):247–256, 2006.

R. Masson, F. Lehuédé, and O. Péton. An adaptive large neighborhood search for the pickup and delivery problem with transfers. *Transportation Science*, 47(3):344–355, 2013.

Z. Miao, K. Fu, and F. Yang. A hybrid genetic algorithm for the multiple crossdocks problem. *Mathematical Problems in Engineering*, 2012.

V.W.C. Morais, G.R. Mateus, and T.F. Noronha. Iterated local search heuristics for the vehicle routing problem with cross-docking. *Expert Systems with Applications*, 41(16): 7495–7506, 2014.

A.I. Nikolopoulou, P.P. Repoussis, C.D. Tarantilis, and E.E. Zachariadis. Moving products between location pairs: Cross-docking versus direct-shipping. *European Journal of Operational Research*, 256(3):803–819, 2017.

S.N. Parragh, K.F. Doerner, and R.F. Hartl. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(1):21–51, 2008.

Y. Qu and J.F. Bard. A grasp with adaptive large neighborhood search for pickup and delivery problems with transshipment. *Computers & Operations Research*, 39(10):2439–2456, 2012.

A. Rais, F. Alvelos, and M.S. Carvalho. New mixed integer-programming model for the pickup-and-delivery problem with transshipment. *European Journal of Operational Research*, 235(3):530–539, 2014.

F.A. Santos, G.R. Mateus, and A.S. da Cunha. A branch-and-price algorithm for a vehicle routing problem with cross-docking. *Electronic Notes in Discrete Mathematics*, 37:249–254, 2011.

F.A. Santos, G.R. Mateus, and A.S. da Cunha. The pickup and delivery problem with cross-docking. *Computers & Operations Research*, 40(4):1085–1093, 2013.

M. Veenstra, M. Cherkesly, G. Desaulniers, and G. Laporte. The pickup and delivery problem with time windows and handling operations. *Computers & Operations Research*, 77:127–140, 2017a.

M. Veenstra, K.J. Roodbergen, I.F.A. Vis, and L.C. Coelho. The pickup and delivery traveling salesman problem with handling costs. *European Journal of Operational Research*, 257(1):118–132, 2017b.

M. Wen, J. Larsen, J. Clausen, and J.-F. Cordeau. Vehicle routing with cross-docking. *The Journal of the Operational Research Society*, 60(12):1708–1718, 2009.