University of Groningen

An Inventory-Routing Problem with Pickups and Deliveries Arising in the Replenishment of Automated Teller Machines

van Anholt, Roel G.; Coelho, Leandro C.; Laporte, Gilbert; Vis, Iris F. A.

Published in:
Transportation Science

DOI:
10.1287/trsc.2015.0637

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2016

Link to publication in University of Groningen/UMCG research database

Citation for published version (APA):
van Anholt, R. G., Coelho, L. C., Laporte, G., & Vis, I. F. A. (2016). An Inventory-Routing Problem with Pickups and Deliveries Arising in the Replenishment of Automated Teller Machines. *Transportation Science*, *50*(3), 1077-1091. https://doi.org/10.1287/trsc.2015.0637

Copyright
Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license. More information can be found on the University of Groningen website: https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment.

Take-down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): http://www.rug.nl/research/portal. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

TRANSPORTATION SCIENCE
Vol. 50, No. 3, August 2016, pp. 1077–1091
ISSN 0041-1655 (print) | ISSN 1526-5447 (online)

http://dx.doi.org/10.1287/trsc.2015.0637
© 2016 INFORMS

An Inventory-Routing Problem with Pickups and Deliveries Arising in the Replenishment of Automated Teller Machines

Roel G. van Anholt
Faculty of Economics and Business Administration, VU University Amsterdam, 1081 HV Amsterdam, Netherlands,
r.g.van.anholt@vu.nl

Leandro C. Coelho
CIRRELT and Faculté des sciences de l'administration, Université Laval, Québec, Québec G1V 0A6, Canada,
leandro.coelho@cirrelt.ca

Gilbert Laporte
CIRRELT and Canada Research Chair in Distribution Management, HEC Montréal, Montréal, Québec H3T 2A7, Canada,
gilbert.laporte@cirrelt.ca

Iris F. A. Vis
Faculty of Economics and Business, University of Groningen, 9747 AE Groningen, Netherlands,
i.f.a.vis@rug.nl

The purpose of this paper is to introduce, model, and solve a rich multiperiod inventory-routing problem with pickups and deliveries motivated by the replenishment of automated teller machines in the Netherlands. Commodities can be brought to and from the depot, as well as being exchanged among customers to efficiently manage their inventory shortages and surpluses. A single customer can both provide and receive commodities at different periods, since its demand changes dynamically throughout the planning horizon and can be either positive or negative. In the case study, new technology provides these machines with the additional functionality of receiving deposits and reissuing banknotes to subsequent customers. We first formulate the problem as a very large-scale mixed-integer linear programming model. Given the size and complexity of the problem, we first decompose it into several more manageable subproblems by means of a clustering procedure, and we further simplify the subproblems by fixing some variables. The resulting subproblems are strengthened through the generation of valid inequalities and solved by branch and cut. We assess the performance of the proposed solution methodology through extensive computational experiments using real data. The results show that we are able to obtain good lower and upper bounds for this new and challenging practical problem.

Keywords: inventory-routing; vehicle routing; inventory management; pickup and delivery; branch and cut; clustering; exact algorithm; recirculation automated teller machines
History: Received: November 2013; revision received: March 2015; accepted: May 2015. Published online in *Articles in Advance* February 24, 2016.

1. Introduction

The purpose of this paper is to introduce, model, and solve an inventory-routing problem with pickups and deliveries (IRPPD) arising in the replenishment of automated teller machines (ATMs). Our study is motivated by the problem faced by a transporter responsible for such operations in the Netherlands. As in other cash-intensive economies, the Dutch credit institutions are gradually replacing regular ATMs by recirculation ATMs (RATMs), which are capable of accepting and dispensing banknotes, as well as checking their quality and authenticity. RATMs therefore provide customers the capability of both depositing and withdrawing cash. These machines provide tangible benefits to banks and customers, and are becoming the new standard in many markets. By the end of 2013, a total of 433,780 RATMs were in operation worldwide, corresponding to 17% of all ATMs (RBR 2014). The number of RATMs increased by 41% between 2012 and 2014, whereas the total number of ATMs grew by only 14% over the same period. RBR (2014) expects that the fast growth of the number and share of RATMs will continue to intensify. RATMs allow retailers to make deposits and enable customers to later withdraw the same cash. In this sense, RATMs are considerably more self-sufficient than the regular machines, which can only dispense cash. RATMs only require a visit to prevent the inventory level from reaching zero or from attaining the holding capacity of the machines when the supply and demand are out of sync. When an RATM is empty, a customer can only use it to deposit cash, and when it is full only withdrawals are allowed.

In the problem considered here, the RATMs are replenished or partly emptied by using a fleet of rented armored trucks, which are very expensive. These trucks deliver cash from a depot to some machines, collect cash from some others to bring it back to the depot, or transfer cash between machines. The last operation reduces the routing cost and sometimes allows smaller or fewer trucks to be used. An important feature of the problem is the presence of inventory holding costs. Indeed, cash lying in a machine generates an implicit holding cost partly because it is insured and also because it incurs lost interest income.

To keep RATMs fully operational, that is when customers can use the RATM to both deposit and withdraw cash, one must solve an IRPPD. The IRPPD combines the features of two well-known classes of the vehicle routing problem: the inventory-routing problem (IRP) and the pickup and delivery problem (PDP), which we now briefly review.

The IRP belongs to the broader field of vendor-managed inventory systems in which a supplier coordinates the inventory management of a number of locations (Coelho, Cordeau, and Laporte 2014). In IRPs, the supplier must simultaneously decide when to visit its inventory locations, how much to deliver to each of them, and how to combine the deliveries into vehicle routes. There exist numerous variants of this problem (Andersson et al. 2010; Coelho, Cordeau, and Laporte 2014) and the related literature is rapidly expanding. The first exact algorithm for the single-vehicle IRP was based on branch and cut (Archetti et al. 2007). It could solve instances with up to 50 customers and three periods, and 30 customers and six periods. Archetti et al. (2012) later presented a hybrid tabu search matheuristic algorithm capable of dealing with larger instances and yielding solutions with very low optimality gaps. Coelho, Cordeau, and Laporte (2012) presented an adaptive large neighborhood search heuristic for the multivehicle IRP, and Coelho and Laporte (2013a) were the first to solve the multivehicle IRP exactly. Recently, multivehicle and multicommodity IRPs were also solved to optimality by Coelho and Laporte (2013b). All of these algorithms are based on branch and cut. For a recent survey of models and algorithms, see Coelho, Cordeau, and Laporte (2014). A related problem appears in maritime transportation, in which ships have to visit several ports to continuously deliver and pickup merchandise and commodities. Applications include the distribution of cement (Christiansen et al. 2011), chemical products (Dauzère-Pérès et al. 2007), and liquefied gases (Rakke et al. 2011), among others. For reviews of maritime transportation, see Christiansen, Fagerholt, and Ronen (2004) and Christiansen et al. (2013).

The PDP concerns the collection and distribution of one or several commodities to and from a set of locations. Berbeglia et al. (2007) classify PDPs and distinguish between three different problem structures: *many-to-many* (M-M), *one-to-many-to-one* (1-M-1), and *one-to-one* (1-1). The M-M structure means that each commodity may have multiple origins and multiple destinations, and that each location may be the origin or destination of multiple commodities. In 1-M-1 problems, some commodities are picked up at the depot and transported to some locations, whereas other commodities are picked up at these locations and transported to the depot. The 1-1 structure refers to a context in which each commodity has a single origin and a single destination, like in dial-a-ride problems (Cordeau and Laporte 2007).

Two important classes of PDPs with an M-M structure are the Swapping Problem (SP) and the One-Commodity Pickup and Delivery Traveling Salesman Problem (1-PDTSP). In the SP, introduced by Anily and Hassin (1992), each vertex of a graph provides a commodity and requests a commodity, possibly the same one. The problem is to design a least cost vehicle route to satisfy all requests. This problem is NP-hard on general graphs, but polynomial on some special structures (Anily, Gendreau, and Laporte 2011). Erdoğan, Laporte, and Cordeau (2010) have developed heuristics and a branch-and-cut algorithm for the multivehicle case. In the 1-PDTSP, each vertex either provides or requests a given amount of a single commodity, and a single vehicle route must be designed to transfer the right amounts of the commodity among vertices. The problem was introduced by Hernández-Pérez and Salazar-González (2003) and was solved by branch and cut (Hernández-Pérez and Salazar-González 2003, 2004a, 2007) and by heuristics (Hernández-Pérez and Salazar-González 2004b; Zhao et al. 2009). The 1-PDTSP arises in the rebalancing operations in shared bicycle systems (Benchimol et al. 2011; Chemla, Meunier, and Wolfler Calvo 2013; Contardo, Morency, and Rousseau 2012; Erdoğan et al. 2012; Erdoğan, Laporte, and Wolfler Calvo 2014; Raviv, Tzur, and Forma 2013). Some algorithms (Contardo, Morency, and Rousseau 2012; Dell'Amico et al. 2014; Raviv, Tzur, and Forma 2013; Shu et al. 2013) are capable of handling the multivehicle case. The problems encountered in RATM replenishment and bicycle repositioning are similar in the sense that they both consist of moving a commodity between locations to keep its level within given limits. The main differences lie in inventory holding costs, which are larger in the case of RATMs, and also in the planning horizon. Indeed, in the case of RATMs, planning is typically done over several days whereas in the case of bicycles, repositioning operations occur on a daily basis, sometimes only once during the night and often several times during the same day.

The PDP with a 1-M-1 structure deals with two types of commodities. Delivery commodities are transported from a single depot to multiple nodes, and pickup commodities are transported from multiple nodes back to the depot. A typical application is the recycling of products such as beer bottles, pallets, and containers (Berbeglia et al. 2007). A distinction is made between combined and single demands. Combined demands occur when locations may require both a pickup and a delivery, whereas single demand refers to problems where each location has either a pickup or delivery demand. The single demand problem was introduced by Mosheiov (1994). Heuristics were proposed by Gendreau, Laporte, and Vigo (1999) and by Subramanian and Battarra (2013), whereas Baldacci, Hadjiconstantinou, and Mingozzi (2003) and Hernández-Pérez and Salazar-González (2007) solved the problem exactly by branch and cut. The combined demand case was introduced by Min (1989). It was solved heuristically by Bianchessi and Righini (2007); Subramanian et al. (2010); Zachariadis, Tarantilis, and Kiranoudis (2010); Vidal et al. (2013), and exactly by branch-and-price algorithms by Angelelli and Mansini (2002); Dell'Amico, Righini, and Salani (2006); Subramanian and Battarra (2013). We are not aware of any contributions on the multicommodity 1-M-1 PDP.

Our aim is to model and solve the IRPPD arising in the replenishment of RATMs. We view it as a novel logistics application arising in the banking service sector. As will be seen, the Dutch application that motivates this study gives rise to a very large-scale mathematical model, which cannot be solved directly. To tackle it, we first decompose it into several subproblems through the application of a clustering procedure, and we further simplify the subproblems by fixing some variables. The resulting subproblems are then strengthened through the generation of valid inequalities and solved by branch and cut. The paper makes four main contributions. First, it introduces pickups and deliveries within an IRP context. Second, it combines two PDP structures: the 1-M-1 structure, which accounts for commodity movements from the depot to RATMs to the depot, and the M-M structure, which refers to commodity transfers among RATMs. Hence, our PDP could appropriately be designated as a 1-M-M-1 problem. Note that we tackle a rather general case of this problem, in that there are several vehicles and side constraints in addition to the standard IRP and PDP features. Third, to cope with realistic sized instances, we propose a practical decomposition methodology combining clustering, variable fixing, and branch and cut. Our fourth contribution is to apply our algorithm to data derived from a real-world case arising in the Netherlands.

The remainder of this paper is organized as follows. We formally describe the problem in mathematical terms in §2, where we provide a mixed-integer formulation. The algorithm is described in §3. This is followed by the results of an extensive computational study in §4 and by conclusions in §5.

2. Mathematical Programming Formulation

The IRPPD is defined on a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$. The set $\mathcal{V}$ of vertices is partitioned into $\{\mathcal{D}, \mathcal{R}\}$, where $\mathcal{D}$ is a set of depots and $\mathcal{R}$ is the set of RATM locations. The set $\mathcal{A} = \{(i, j)\colon i, j \in \mathcal{V}, i \neq j\}$ is the arc set. Each RATM incurs unit inventory holding costs α_i per period ($i \in \mathcal{R}$), and has an inventory holding capacity C_i. A handling cost β is incurred at each depot, and is proportional to the quantity picked up and delivered at the depot. The length of the planning horizon is p, with discrete time periods $t \in \mathcal{T} = \{1, \ldots, p\}$. A set of rented armored vehicles $k \in \mathcal{K} = \{1, \ldots, K\}$, each with capacity Q_k and average speed s_k, is given. A renting cost γ_k per period is incurred if vehicle k is used. Each vehicle is able to perform one route per period, from one depot to a subset of RATMs, each requiring r units of time to be served, and back to the same depot. The shift of each vehicle is limited to S minutes, after which δ monetary units per minute of overtime are incurred. A travel time c_{ij} in minutes is associated with arc $(i, j) \in \mathcal{A}$. We assume each depot has sufficient inventory and capacity to perform all pickups and deliveries during the planning horizon. The inventories are not allowed to exceed the holding capacity nor are they allowed to become negative. At the beginning of the planning horizon, the decision maker knows the current inventory level I_i^0 of the RATMs and receives information on the net demand d_i^t of each RATM i for each period t. Negative demands mean that the RATM provides a commodity, whereas positive demands mean that the RATM receives some quantity of the commodity. We assume that the quantities q_i^t received by RATM i in period t can be used to satisfy its net demand in that period. The quantities picked up at the depots and at the RATMs may be delivered to any RATM to satisfy their demands. The objective of the problem is to minimize the total cost while satisfying the net demand for each RATM in each period.

The variables used in the formulation are as follows. Five families of binary variables are used: directed routing variables x_{ij}^{kt} are equal to 1 if and only if arc (i, j) is used on the route of vehicle k in period t; visiting variables y_i^{kt} are equal to 1 if and only if RATM i is visited by vehicle k in period t; vehicle usage variables v^{kt} are equal to 1 if and only if vehicle k is used in period t; delivery variables w_i^{kt} are equal to 1 if and only if a delivery is made to RATM i by vehicle k in period t; and pickup variables z_i^{kt} are 1 if and only if a pickup is performed at RATM i by vehicle k in period t. Integer variables I_i^t represent the inventory level at RATM $i \in \mathcal{R}$ at the end of period $t \in \mathcal{T}$. Variables q_i^{kt} are integers representing the product quantity delivered to RATM i using vehicle k in period t, and variables p_i^{kt} are integers representing the product quantity picked up from RATM i using vehicle k in period t. Two sets of variables represent the amount of inventory carried by vehicle k in period t into and out of the depot: H_k^t and J_k^t, respectively. The load of vehicle k after serving RATM i in period t is represented by variables u_i^{kt}, and E_k^t represents the overtime in number of extra minutes worked by vehicle k in period t beyond the regulatory shift duration S.

The multidepot IRPPD is then formulated as follows:

(MD-IRPPD)

$$\text{minimize } \left\{ \sum_{i\in\mathcal{R}}\sum_{t\in\mathcal{T}} \alpha_i I_i^t + \sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}} \beta (J_k^t + H_k^t) + \sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}} \gamma_k v^{kt} + \sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}} \delta E_k^t \right\} \quad (1)$$

subject to

$$I_i^t = I_i^{t-1} + \sum_{k\in\mathcal{K}} q_i^{kt} - \sum_{k\in\mathcal{K}} p_i^{kt} + d_i^t, \quad i\in\mathcal{R}\ t\in\mathcal{T}, \quad (2)$$

$$0 \le I_i^t \le C_i, \quad i\in\mathcal{R}\ t\in\mathcal{T}, \quad (3)$$

$$\sum_{j\in\mathcal{V}} x_{ij}^{kt} - \sum_{j\in\mathcal{V}} x_{ji}^{kt} = 0, \quad i\in\mathcal{V}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (4)$$

$$\sum_{j\in\mathcal{V}} x_{ij}^{kt} = y_i^{kt}, \quad i\in\mathcal{V}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (5)$$

$$\sum_{i\in\mathcal{D}}\sum_{j\in\mathcal{R}} x_{ij}^{kt} \le 1, \quad k\in\mathcal{K}\ t\in\mathcal{T}, \quad (6)$$

$$y_i^{kt} \le v^{kt}, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (7)$$

$$v^{kt} \le \sum_{i\in\mathcal{D}}\sum_{j\in\mathcal{R}} x_{ij}^{kt}, \quad k\in\mathcal{K}\ t\in\mathcal{T}, \quad (8)$$

$$\sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}} x_{ij}^{kt} \le \sum_{i\in\mathcal{S}} y_i^{kt} - y_m^{kt}, \quad \mathcal{S}\subseteq\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}\ m\in\mathcal{S}, \quad (9)$$

$$w_i^{kt} \le y_i^{kt}, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (10)$$

$$z_i^{kt} \le y_i^{kt}, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (11)$$

$$q_i^{kt} \le w_i^{kt}(C_i - I_i^t), \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (12)$$

$$p_i^{kt} \le z_i^{kt} I_i^t, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (13)$$

$$w_i^{kt} + z_i^{kt} \le 1, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (14)$$

$$s_k \sum_{(i,j)\in\mathcal{A}} c_{ij} x_{ij}^{kt} + r\sum_{i\in\mathcal{R}} y_i^{kt} \le S + E_k^t, \quad k\in\mathcal{K}\ t\in\mathcal{T}, \quad (15)$$

$$u_j^{kt} \ge (u_i^{kt} + p_j^{kt} - q_j^{kt}) x_{ij}^{kt}, \quad i,j\in\mathcal{V}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (16)$$

$$0 \le u_i^{kt} \le Q_k, \quad i\in\mathcal{V}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (17)$$

$$J_k^t = \sum_{i\in\mathcal{D}} u_i^{kt} y_i^{kt}, \quad k\in\mathcal{K}\ t\in\mathcal{T}, \quad (18)$$

$$H_k^t = \sum_{i\in\mathcal{R}}\sum_{j\in\mathcal{D}} x_{ij}^{kt} u_i^{kt}, \quad k\in\mathcal{K}\ t\in\mathcal{T}, \quad (19)$$

$$q_i^{kt}, p_i^{kt}, J_k^t, H_k^t \ge 0, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (20)$$

$$x_{ij}^{kt} \in \{0,1\}, \quad (i,j)\in\mathcal{A}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (21)$$

$$v^{kt}, y_i^{kt} \in \{0,1\}, \quad i\in\mathcal{V}\ k\in\mathcal{K}\ t\in\mathcal{T}, \quad (22)$$

$$w_i^{kt}, z_i^{kt} \in \{0,1\}, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}. \quad (23)$$

The objective function (1) minimizes the total cost of inventory holding, inventory handling at the depot, and vehicle renting. Constraints (2) state the inventory conservation condition over successive periods: they define the inventory in period t as the inventory held in period $t-1$, plus the quantity delivered, minus the quantity picked up, plus the net demand of the location. Constraints (3) define the bounds on the inventory held by each RATM throughout all periods. Constraints (4)–(9) guarantee that proper vehicle routes are created. More specifically, constraints (4) are flow conservation constraints, constraints (5) relate the incoming flow of a vertex to the fact that it is visited or not. Constraints (6) state that a vehicle can leave at most one depot in any period. Constraints (7) mean that a vertex cannot be visited if no vehicle is assigned to it. Similarly, constraints (8) relate the use of a vehicle during a given period to the flow variables associated with that vehicle and that period. Constraints (9) prevent the formation of subtours over a set of RATMs. These are the subtour elimination constraints proposed by Gendreau, Laporte, and Semet (1997) for the covering tour problem. Constraints (10) link the delivery binary variables w_i^{kt} to the visiting binary variables y_i^{kt}. They allow a delivery decision to be made only if a visit is performed to the RATM. Constraints (11) are similar to (10) and apply to the pickup decisions. Constraints (12) and (13) allow a quantity to be delivered or picked up at an RATM only if the corresponding binary decision variable is equal to 1. Constraints (14) ensure that a visit to an RATM is made either for a pickup or for a delivery operation. Constraints (15) guarantee that the shift duration is respected and that overtime is properly accounted for. Constraints (16) ensure that the load within the vehicle is consistent along its route. Constraints (17) set bounds on the load inside the vehicle after serving RATM i. Constraints (18) set the load of the vehicle when leaving the depot, whereas constraints (19) compute its load when returning to the depot at the end of the route. Finally, constraints (20)–(23) define nonnegativity and binary conditions on the variables.

3. Solution Algorithm

In the RATM application that motivates this study, 6,377 machines must be replenished by 32 vehicles, each based at a different depot, over a six-day planning horizon. This means that the (MD-IRPPD) model just described will contain more than 7.8 billion binary variables and about the same number of nonlinear constraints, in addition to $O(2^{|\mathcal{R}|})$ subtour elimination constraints. Solving such a model or even its continuous relaxation is beyond the reach of any available mathematical programming solver. We have therefore opted for a practical three-step solution strategy. We first apply a clustering procedure to decompose the initial problem into 32 subproblems of about 200 RATMs each, one for each depot (each having a single vehicle). This reduces the number of variables of each subproblem to a more manageable size, around 250,000, but still too many to reach optimality. For comparison purposes, the largest IRP instance that can be solved exactly, with a single vehicle and a single depot, contains 200 customers (Coelho and Laporte 2014), but the IRP does not contain many of the features presented in this paper. Likewise, the largest M-M PDP instance solved to optimality contains 200 vertices and a single vehicle (Hernández-Pérez and Salazar-González 2007). The IRPPD combines features of these two problems and is significantly more complicated than either of these. For these reasons, it is unrealistic to solve this problem exactly for the large sizes we are considering. We therefore simplify the subproblems further by fixing some variables. This yields instances that can be solved exactly or to near optimality by branch and cut. These three phases are described in §§3.1–3.3, respectively. Algorithm 1 provides a sketch of our solution procedure.

Algorithm 1 (Overall solution algorithm)

```
1: Clustering procedure: solve a transportation
     problem to obtain as many single-depot
     instances as the number of depots (see §3.1)
2: for each single-depot instance do
3:    Apply the variable fixing procedure (see §3.2)
        to reduce the instance size
4:    Solve the IRPPD model by branch and cut
        with some fixed variables and valid
        inequalities (see §3.3)
5: end for.
```

3.1. Clustering Procedure

Partitioning the customer base into clusters not only makes sense from a computational point of view but also from an empirical standpoint because this is how the problem is solved in practice. We solve a transportation problem to assign RATMs to depots while controlling the size of the clusters. The assignment cost in each cluster is the sum of the radial average travel times, interpreted here as distances. Eilon, Watson-Gandy, and Christofides (1971) show that radial distances are often a good proxy for routing costs. Let π_{ij} be a binary variable equal to 1 if and only if RATM i is assigned to depot j. The transportation problem is then

$$\text{minimize} \quad \sum_{i\in\mathcal{R}}\sum_{j\in\mathcal{D}}(c_{ij}+c_{ji})\pi_{ij}/2 \tag{24}$$

$$\text{subject to} \quad \sum_{j\in\mathcal{D}}\pi_{ij}=1 \quad i\in\mathcal{R}, \tag{25}$$

$$\sum_{i\in\mathcal{R}}\pi_{ij}\le A \quad j\in\mathcal{D}, \tag{26}$$

$$\pi_{ij}\in\{0,1\} \quad i\in\mathcal{R}\ j\in\mathcal{D}. \tag{27}$$

The objective function (24) minimizes the total sum of the distances between assigned RATMs and depots. Constraints (25) ensure that each RATM is assigned to exactly one depot, and constraints (26) ensure that each depot is assigned to at most A RATMs, with A chosen so as to balance the number of RATMs per depot. Constraints (27) define the domain of the variables.

After the clustering procedure has been applied, the problem is simplified and the (MD-IRPPD) formulation can be reduced to one without the depot index, i.e., $\mathcal{D}=\{0\}$. We provide this formulation for the sake of completeness and because we will later use it to generate valid inequalities. Reusing the notation employed in §2, we now state the single-depot formulation

(IRPPD)

$$\text{minimize} \quad \left\{\sum_{i\in\mathcal{R}}\sum_{t\in\mathcal{T}}\alpha_i I_i^t+\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}}\beta(J_k^t+H_k^t)+\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}}\gamma_k y_0^{kt}+\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}}\delta E_k^t\right\} \tag{28}$$

subject to

$$I_i^t=I_i^{t-1}+\sum_{k\in\mathcal{K}}q_i^{kt}-\sum_{k\in\mathcal{K}}p_i^{kt}+d_i^t, \quad i\in\mathcal{R}\ t\in\mathcal{T}, \tag{29}$$

$$0\le I_i^t\le C_i, \quad i\in\mathcal{R}\ t\in\mathcal{T}, \tag{30}$$

$$\sum_{j\in\mathcal{V}}x_{ij}^{kt}-\sum_{j\in\mathcal{V}}x_{ji}^{kt}=0, \quad i\in\mathcal{V}\ k\in\mathcal{K}\ t\in\mathcal{T}, \tag{31}$$

$$\sum_{j\in\mathcal{V}}x_{ij}^{kt}=y_i^{kt}, \quad i\in\mathcal{V}\ k\in\mathcal{K}\ t\in\mathcal{T}, \tag{32}$$

$$\sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}}x_{ij}^{kt}\le\sum_{i\in\mathcal{S}}y_i^{kt}-y_m^{kt}, \quad \mathcal{S}\subseteq\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}\ m\in\mathcal{S}, \tag{33}$$

$$w_i^{kt}\le y_i^{kt}, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \tag{34}$$

$$z_i^{kt}\le y_i^{kt}, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \tag{35}$$

$$q_i^{kt}\le w_i^{kt}(C_i-I_i^t), \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \tag{36}$$

$$p_i^{kt}\le z_i^{kt}I_i^t, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \tag{37}$$

$$w_i^{kt}+z_i^{kt}\le 1, \quad i\in\mathcal{R}\ k\in\mathcal{K}\ t\in\mathcal{T}, \tag{38}$$

$$s_k\sum_{(i,j)\in\mathcal{A}}c_{ij}x_{ij}^{kt}+r\sum_{i\in\mathcal{R}}y_i^{kt}\le S+E_k^t, \quad k\in\mathcal{K}\ t\in\mathcal{T}, \tag{39}$$

$$u_j^{kt}\ge(u_i^{kt}+p_j^{kt}-q_j^{kt})x_{ij}^{kt}, \quad i,j\in\mathcal{V}\ k\in\mathcal{K}\ t\in\mathcal{T}, \tag{40}$$

$$0 \le u_i^{kt} \le Q_k, \quad i \in \mathcal{V} \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{41}$$

$$J_k^t = u_0^{kt}, \quad k \in \mathcal{K} \; t \in \mathcal{T}, \tag{42}$$

$$H_k^t = \sum_{i \in \mathcal{R}} x_{i0}^{kt} u_i^{kt}, \quad k \in \mathcal{K} \quad t \in \mathcal{T}, \tag{43}$$

$$q_i^{kt}, p_i^{kt}, J_k^t, H_k^t \ge 0, \quad i \in \mathcal{R} \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{44}$$

$$x_{ij}^{kt} \in \{0,1\}, \quad (i,j) \in \mathcal{A} \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{45}$$

$$y_i^{kt} \in \{0,1\}, \quad i \in \mathcal{V} \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{46}$$

$$w_i^{kt}, z_i^{kt} \in \{0,1\}, \quad i \in \mathcal{R} \; k \in \mathcal{K} \; t \in \mathcal{T}. \tag{47}$$

The description of the variables, objective function, and constraints is similar to that of the multidepot model. Three comments are relevant with respect to the IRPPD formulation. The first regards constraints (40), which resemble the Miller-Tucker-Zemlin (1960) subtour elimination constraints. These constraints do not eliminate subtours in this context, because the load within the vehicle is not monotonically increasing or decreasing. We therefore impose Dantzig-Fulkerson-Johnson (1954) subtour elimination constraints (33) whose number is $O(2^{|\mathcal{R}|})$. Second, the load consistency constraints (40) have been used in a number of other pickup and delivery problems; see, e.g., Desaulniers et al. (2002); Gribkovskaia et al. (2007); Hoff et al. (2009). When it is known in advance whether an RATM requires a pickup or a delivery, they can be lifted (Gribkovskaia et al. 2007). However, this is not the case here. The third comment refers to the fact that the formulation is nonlinear because of constraints (36), (37), (40), and (43). However, these can be linearized as follows. Constraints (36) and (37) can be rewritten in a slightly weaker form as

$$q_i^{kt} \le w_i^{kt} C_i, \quad i \in \mathcal{R} \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{48}$$

$$p_i^{kt} \le z_i^{kt} C_i, \quad i \in \mathcal{R} \; k \in \mathcal{K} \; t \in \mathcal{T}. \tag{49}$$

On their own, these constraints would allow quantities to be picked up or delivered to violate the inventory bounds, but together with constraints (29) and (30), they are feasible linear representations of constraints (36) and (37). Constraints (40) can be linearized as

$$u_j^{kt} \ge u_i^{kt} + p_j^{kt} - q_j^{kt} - (1 - x_{ij}^{kt}) Q_k, \quad i, j \in \mathcal{V} \; k \in \mathcal{K} \; t \in \mathcal{T}. \tag{50}$$

Finally, constraints (43) can be rewritten in a linear form as

$$H_k^t \ge u_i^{kt} - (1 - x_{i0}^{kt}) Q_k, \quad i \in \mathcal{R} \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{51}$$

$$H_k^t \le Q_k, \quad k \in \mathcal{K} \; t \in \mathcal{T}. \tag{52}$$

The IRPPD formulation can be strengthened through the generation of the following valid inequalities:

$$x_{ij}^{kt} \le y_i^{kt}, \quad i, j \in \mathcal{R} \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{53}$$

$$y_i^{kt} \le y_0^{kt}, \quad i \in \mathcal{R} \; k \in \mathcal{K} \; t \in \mathcal{T}. \tag{54}$$

Constraints (53) are referred to as logical inequalities. They tighten the relation between routing and visiting variables. Constraints (54) include the supplier in the route of vehicle k if any RATM is visited by that vehicle in that period.

3.2. Variable Fixing Procedure

To solve realistic instances, we have designed a variable fixing procedure, which is applied prior to executing the branch-and-cut algorithm, in the spirit of granular search (Johnson and McGeoch 1997; Toth and Vigo 2003). This heuristic determines for each RATM $i \in \mathcal{R}$ and each period $t \in \mathcal{T}$ whether it must be visited, must not be visited, or must perhaps be visited. Specifically, for each period t we distinguish five subsets: (1) a *must pickup* set $\mathcal{U}^t$, (2) a *must deliver* set $\mathcal{W}^t$, (3) a *perhaps pickup* set $\mathcal{X}^t$, (4) a *perhaps deliver* set $\mathcal{Y}^t$, and (5) a *not visit* set $\mathcal{Z}^t$. These five sets define a partition of $\mathcal{R}$. When RATMs are added to one of the subsets $\mathcal{U}^t$, $\mathcal{W}^t$, or $\mathcal{Z}^t$, the size of the formulation is considerably reduced and the branch-and-cut algorithm benefits from this reduction, because some of the decisions related to pickups, deliveries, and visits for some RATMs are already made. When RATMs are added to one of the other two subsets $\mathcal{X}^t$ or $\mathcal{Y}^t$, the algorithm still determines for each period t whether a visit is required, but the benefit of the variable fixing procedure is that it limits the choice to either a pickup or a delivery. It is implemented as follows:

$$\sum_{k \in \mathcal{K}} z_i^{kt} = 1, \quad i \in \mathcal{U}^t \; t \in \mathcal{T}, \tag{55}$$

$$\sum_{k \in \mathcal{K}} w_i^{kt} = 1, \quad i \in \mathcal{W}^t \; t \in \mathcal{T}, \tag{56}$$

$$\sum_{k \in \mathcal{K}} w_i^{kt} = 0, \quad i \in \mathcal{X}^t \; t \in \mathcal{T}, \tag{57}$$

$$\sum_{k \in \mathcal{K}} z_i^{kt} = 0, \quad i \in \mathcal{Y}^t \; t \in \mathcal{T}, \tag{58}$$

$$\sum_{k \in \mathcal{K}} y_i^{kt} = 0, \quad i \in \mathcal{Z}^t \; t \in \mathcal{T}. \tag{59}$$

Also, it follows directly that several other constraints fixing binary variables can be derived from (55)–(59). These are used to further reduce the size of the problem by effectively eliminating several variables from the problem

$$\sum_{k \in \mathcal{K}} y_i^{kt} = 1, \quad i \in \mathcal{U}^t \; t \in \mathcal{T}, \tag{60}$$

$$w_i^{kt} = 0, \quad i \in \mathcal{U}^t \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{61}$$

$$\sum_{k \in \mathcal{K}} y_i^{kt} = 1, \quad i \in \mathcal{W}^t \; t \in \mathcal{T}, \tag{62}$$

$$z_i^{kt} = 0, \quad i \in \mathcal{W}^t \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{63}$$

$$w_i^{kt} = 0, \quad i \in \mathcal{X}^t \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{64}$$

$$z_i^{kt} = 0, \quad i \in \mathcal{Y}^t \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{65}$$

$$w_i^{kt} = 0, \quad i \in \mathcal{Z}^t \; k \in \mathcal{K} \; t \in \mathcal{T}, \tag{66}$$

$$z_i^{kt} = 0, \quad i \in \mathcal{Z}^t \; k \in \mathcal{K} \; t \in \mathcal{T}. \tag{67}$$

The variable fixing procedure uses the parameters defined below, which allow us to test various settings. The first parameter relates to *due periods* and *tipping periods*. A due period is a period in which a visit is required to prevent the inventory from exceeding the holding capacity or from a stock out. A tipping period is a period in which a pickup is required to remain cost efficient, that is when the holding cost is disproportionately high related to the cost of visiting the RATM. The parameters are as follows:

- m: number of periods preceding a due period or a tipping period;
- f: number of RATMs that are closest to RATM i;
- g: minimum inventory level of RATM i, in a percentage of its holding capacity C_i;
- b: expected number of RATM visits per vehicle k in a period t.

In what follows, let $\bar{I}_i^{t'} = I_i^0 + \sum_{t=1}^{t'} d_i^t$ be the cumulative inventory of RATM i in period t'. This is useful to identify up to which period t' the RATM will be capable of respecting its inventory constraints without replenishment. The pseudocode of the variable fixing procedure is presented in Algorithm 2.

Algorithm 2 (Variable fixing procedure (m, f, g, b))

```
1:  $\bar{\mathcal{R}} \leftarrow \mathcal{R}$
2:  for all $i \in \mathcal{R}$ do
3:    for all $t \in \mathcal{T}$ do
4:      Add $i$ to $\mathcal{Z}^t$
5:      if $t = 1$ or $m = 0$ then
6:        if $\bar{I}_i^t > C_i$ then
7:          Add $i$ to $\mathcal{U}^t$; remove $i$ from $\mathcal{Z}^t$; remove $i$ from $\bar{\mathcal{R}}$
8:        else if $\bar{I}_i^t < 0$ then
9:          Add $i$ to $\mathcal{W}^t$; remove $i$ from $\mathcal{Z}^t$; remove $i$ from $\bar{\mathcal{R}}$
10:       end if
11:     else
12:       for $t' = t - m$ to $t' = t$; $t' > 1$ do
13:         if $\sum_{t'' \in \mathcal{T}} d_i^{t''} > 0$ and $(\bar{I}_i^{t'} \times \alpha_i) > (\gamma_k/b/(\bar{I}_i^{t'}/((\sum_{t'' \in \mathcal{T}} d_i^{t''})/p)))$ then
14:           Add $i$ to $\mathcal{X}^t$; remove $i$ from $\mathcal{Z}^t$; remove $i$ from $\bar{\mathcal{R}}$
15:         else if $\bar{I}_i^{t'} > C_i$ then
16:           Add $i$ to $\mathcal{X}^t$; remove $i$ from $\mathcal{Z}^t$; remove $i$ from $\bar{\mathcal{R}}$
17:         else if $\bar{I}_i^{t'} < 0$ then
18:           Add $i$ to $\mathcal{Y}^t$; remove $i$ from $\mathcal{Z}^t$; remove $i$ from $\bar{\mathcal{R}}$
19:         end if
20:       end for
21:     end if
22:     for all $f$ number of RATMs $j$ having the smallest $c_{ij}$ to $i$ do
23:       if $i \in \{\mathcal{U}^t, \mathcal{X}^t\}$ and $(\bar{I}_j^{t'}/C_j) < (1-g)$ then
24:         Add $j$ to $\mathcal{Y}^t$; remove $i$ from $\mathcal{Z}^t$; remove $i$ from $\bar{\mathcal{R}}$
25:       else if $i \in \{\mathcal{W}^t, \mathcal{Y}^t\}$ and $(\bar{I}_j^{t'}/C_j) > g$ then
26:         Add $j$ to $\mathcal{X}^t$; remove $i$ from $\mathcal{Z}^t$; remove $i$ from $\bar{\mathcal{R}}$
27:       end if
28:     end for
29:   end for
30: end for
31: Return $\mathcal{U}^t, \mathcal{W}^t, \mathcal{X}^t, \mathcal{Y}^t, \mathcal{Z}^t$.
```

The variable fixing procedure starts by adding each RATMs i in each period t to the not visit subset $\mathcal{Z}^t$ (lines 2–4). When $t = 1$ or $m = 0$, lines 5–10 ensure that RATMs are added to the must visit subsets $\mathcal{U}^t$ and $\mathcal{W}^t$ when the holding capacity C_i is exceeded or when the inventory $\bar{I}_i^{t'}$ becomes negative. If either condition is satisfied, the algorithm does not have the flexibility to choose between preceding or succeeding periods to visit the RATM. Therefore, the RATM must be visited.

If the condition in line 5 is not met, we check three more conditions in lines 13–18 to add RATMs i to the perhaps visit subsets $\mathcal{X}^t$ and $\mathcal{Y}^t$ in period t as well as to the subsets in m preceding periods t. Line 12 ensures that RATMs i are also added to the subsets in m number of preceding periods t. We introduce lines 13 and 14 to ensure RATM i is visited for a pickup when a tipping period occurs, i.e., when the holding cost exceeds the cost of visiting the RATM. More precisely, the following condition is verified: the inventory $\bar{I}_i^{t'}$ of RATM i in period t multiplied with the unit inventory holding cost α_i exceeds the vehicle rental cost γ_k divided by the expected number of visits per route b and divided by the expected number of days elapsed since the last visit. To this end, we calculate the expected number of days elapsed since the last visit by dividing the inventory level $\bar{I}_i^{t'}$ by the average demand $\sum_{t \in \mathcal{T}} d_i^t/t$. If the latter condition is met, then the RATM i is added to the perhaps pickup set $\mathcal{X}^t$. RATMs i are also added to the $\mathcal{X}^t$ subset in periods t when the inventory exceeds capacity $\bar{I}_i^{t'} > C_i$ (lines 15 and 16). In the case of stock outs, the RATMs are added to the perhaps deliver subset $\mathcal{Y}^t$, which is indicated in lines 17 and 18.

Most likely, some RATMs i will have been taken out of set $\mathcal{Z}^t$ and added to the other subsets $\mathcal{U}^t, \mathcal{W}^t, \mathcal{X}^t$, and $\mathcal{Y}^t$ when arriving at line 18. The final part in lines 22–28 ensures that even more RATMs j, which are located closest to RATMs i belonging to $\mathcal{U}^t, \mathcal{W}^t, \mathcal{X}^t$, or $\mathcal{Y}^t$, are added to the perhaps visit subsets. The number of RATMs j added per RATM i is determined by parameter f. It should be noted that RATMs j are only added when they meet an inventory level $\bar{I}_j^t$, which is at least as high as a given percentage g of the holding capacity C_i for a pickup, or lower than a given percentage $1-g$ for a delivery. The rationale of this final part is to stimulate the exchange of inventories between RATMs i and j to eventually reduce the amount to be picked up from, and delivered to the depot.

3.3. Branch-and-Cut Algorithm

We have implemented a branch-and-cut algorithm capable of solving the IRPPD model with the linearized constraints, the valid inequalities, and the fixed variables. All variables of the formulation are explicitly handled by the algorithm, but we cannot generate all subtour elimination constraints (33) a priori. These will be dynamically generated as cuts as they are found to be violated. The formulation is then solved by branch and cut as follows. At a generic node of the search tree, a linear program with relaxed integrality constraints is solved, a search for violated constraints is performed, and violated valid inequalities are added to the current program, which is then reoptimized. This process is reiterated until a feasible or dominated solution has been reached, or until no more cuts can be added. At this point, branching on a fractional variable occurs. Details regarding the implementation and improvements to this algorithm are provided in §4.2.

4. Computational Experiments

We will present the instances generator in §4.1, some implementation details in §4.2, the analysis of our extensive computational experiments in §4.3, and the estimated benefits of our study in §4.4. The instance set as well as detailed computational results are available at http://www.leandro-coelho.com/instances.

4.1. Instances Description

The data used in our experiments stem from a real-world case in the Netherlands. A total of 6,377 cash dispensing self-service devices, both regular and recirculation ATMs, were installed in the Netherlands in 2013 (ABN ARMO Bank 2013; ING Bank 2013; Rabobank 2013). These are replenished from 32 cash centers (depots). By setting A equal to 200 in constraints (26), we assigned the 6,377 RATMs fairly evenly to the depots, each of which serves a small region of the Netherlands (Figure 1). For our experiments, we assume that all self-service devices in the Netherlands are RATMs. This is not unrealistic given the recent strong increase in the share of RATMs in the Netherlands (ECB 2013). Also, in several other countries, such as Japan, RATMs already dominate the cash self-service device market (RBR 2014). To conduct our numerical experiments, we used real data for the locations of RATM devices, true distances and speeds, and real RATM demands. Although the Netherlands is rather densely populated, not all parts of it are equally well served by RATMs. Their coverage varies considerably over the country. Our solution methodology is rather robust and can easily deal with this diversity. Figure 2 depicts the position of the RATMs and the cash center for the Amsterdam area. There still exist some ARMs that are not RATMs. For these, we generated RATM demands. The depot locations are confidential for security reasons, so we used approximate locations.

To mimic real demands for devices that are not yet RATMs, we have carefully studied real demands from readily installed RATMs. This study demonstrates that the number of transactions per day is well estimated by a Poisson probability distribution; the number of deposits and the number of withdrawals follow completely different patterns in terms of quantity, value, and weekly seasonality pattern; a huge diversity exists among the demand intensity of RATMs, and the average value per deposit is €1,000 and the average value per withdrawal is €133.

Based on these findings, we have generated deposit and withdrawal distributions separately and added these sequentially to construct a net demand per RATM per day. For the deposits and withdrawals of each RATM, we uniformly assigned two demand intensity levels of one to five to mimic demand intensity for both deposits and withdrawals (Table 1).

By randomly drawing from a Poisson distribution with the assigned number of transactions per day and multiplying the resulting number of transactions by the average transaction amount, i.e., €1,000 or €133, we obtained daily deposit and withdrawal amounts. Finally, to cope with week seasonality, we modified these amounts with the factors described in Table 2.

The parameters of our instance are then as follows:

- $\alpha_i =$ €0.08 per €1,000 inventory per period t (based on a 3% annual interest rate).
- $C_i =$ €260,000 per RATM i. This is an estimation based on an RATM with four cassettes, each with a capacity of 2,000 notes.
- $\beta =$ €0.30 per €1,000 picked up or delivered at the depot.
- $p =$ six periods. Periods coincide with days, which together define a workweek from Monday to Saturday. In practice, order lead times are not more than one or two days and so a planning horizon of six days is sufficient.
- $\mathcal{K} =$ one vehicle.
- $Q_k =$ €7,800,000 per vehicle k. This is an estimation based on 30 full replenishments of €260,000 in a single period.

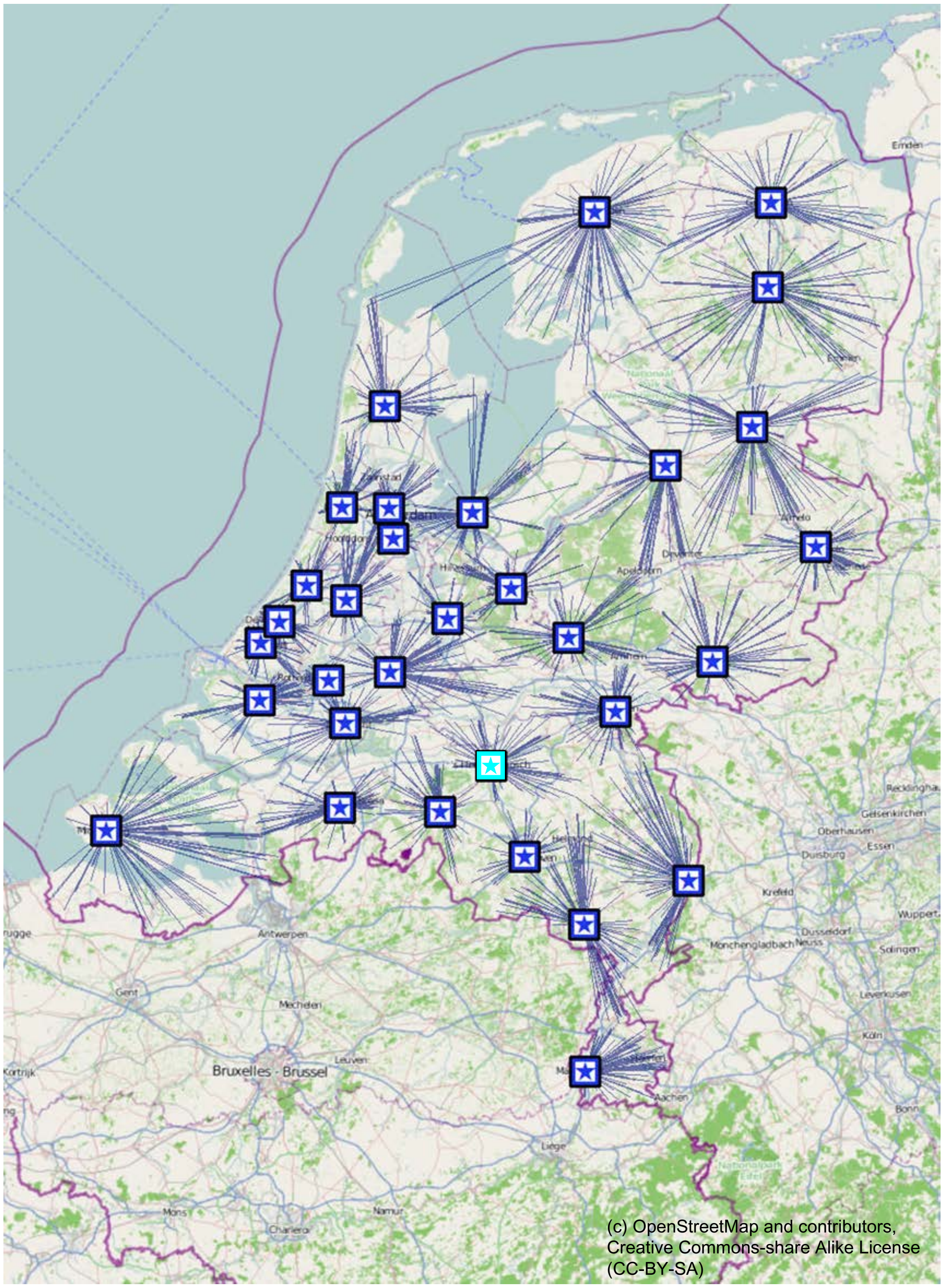
(c) OpenStreetMap and contributors,
Creative Commons-share Alike License
(CC-BY-SA)

Figure 1 (Color online) Map of the Netherlands Depicting 32 Subregions

- $\gamma_k =$ €2,000 per vehicle k per period used.
- r = 18 minutes.
- $S =$ eight hours. This is the regular daily work time in the Netherlands. We assume the vehicle is loaded prior to performing the route, so the driver has at most eight hours to perform all pickups and deliveries.
- $\delta =$ €8.00 per minute, which is approximately twice as expensive as the regular vehicle renting cost.
- $I_i^0 \in \{0, \ldots, C_i\}$. Each RATM i is assigned a random initial inventory in euros.
- $d_i^t = \{\pm -45{,}000, \ldots, \pm 45{,}000\}$ per period. Random values are drawn from a Poisson distribution for both

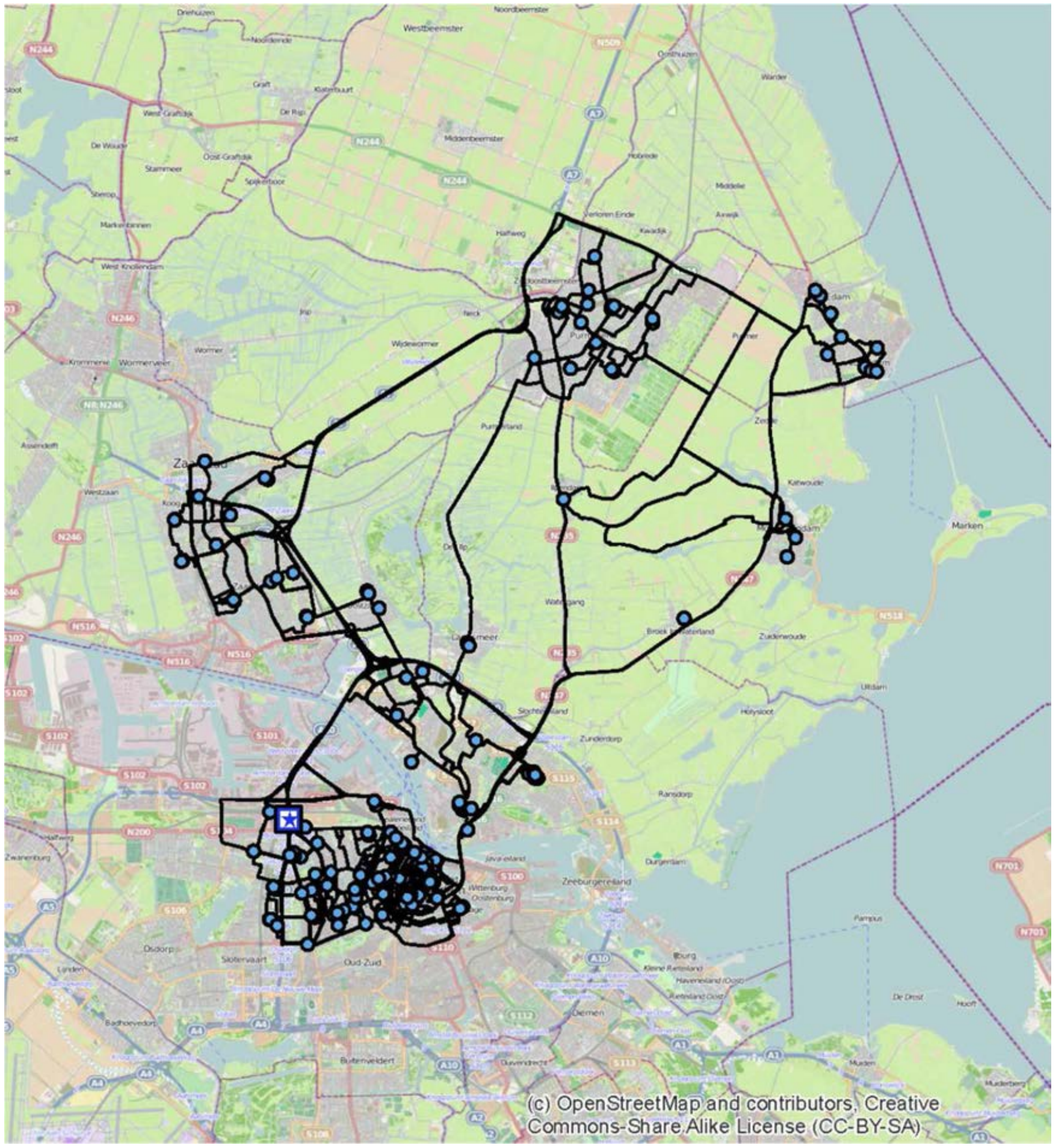
(c) OpenStreetMap and contributors, Creative Commons-Share Alike License (CC-BY-SA)

Figure 2 (Color online) Map of the Amsterdam Subregion

withdrawals and deposits, which are thereafter combined into a net demand. To simulate real demands at different locations, we have ensured that RATMs either have a net-positive, net-negative, or balanced demand without losing stochasticity. We have then generated different demands at distinct periods: in periods $t \in \{1, 2\}$ the demand tends to be net-positive, because in the Netherlands more cash is deposited on Mondays and Tuesdays. The demand in periods $t \in \{5, 6\}$ tends to be net-negative, since more cash is withdrawn on Fridays and Saturdays.

- c_{ij} = the arc set is constructed by first using real travel distances between the RATMs and the depot. A routable network data set for the Netherlands was then constructed using OpenStreetMap data (Geofabrik GmbH, OpenStreetMap Contributors 2013) and average driving speeds were used on the various road types to approximate true speeds.

4.2. Implementation Features

The algorithm just described was coded in C++ using IBM Concert Technology and solved with CPLEX 12.5.1 running on a single thread. All computations were

Table 1 Different Levels of Deposits and Withdrawals per Day

Demand intensity level	1	2	3	4	5
Deposits per day	10.0	17.5	25.0	32.5	40.0
Withdrawals per day	75.0	131.3	187.5	243.8	300.0

Table 2 Seasonal Factors for Deposits and Withdrawals in Percentage

Weekday	Mon	Tue	Wed	Thu	Fri	Sat
Deposits	120	110	100	90	80	100
Withdrawals	70	80	90	110	120	130

executed on a grid of Intel Xeon processors running at 2.66 GHz with up to 48 GB RAM installed per node, with the Scientific Linux 6.1 operating system. A time limit of six hours was imposed on the execution of each of the 32 instances.

The clustering procedure was run only once. It is conceivable that overall improvements could have been obtained by performing 1-opt or 2-opt moves between clusters, but there exist $O(|\mathcal{R}|^2)$ such potential moves and evaluating each of them would have entailed solving a large-scale integer linear program, which was impractical.

Several experiments were performed with various variable fixing parameters. These settings were gathered from cash supply chain parties in the Netherlands, and estimated when these could not be made publicly available for security reasons. The parameters for the clustering heuristic were set as follows: $m \in \{0, 1, 2\}$; $f \in \{0, 1, 2\}$; $g \in \{30, 50, 70\}$; $b \in \{15\}$.

For the instances considered in this paper, which contain up to 200 vertices, we cannot obtain optimal solutions with our branch-and-cut algorithm. To cope with this situation, we have decided to add two new layers to the branch-and-cut algorithm. In the first one, we verify at every node having a fractional solution whether inequalities (53) are violated and we then add them to the linear program. This helps improve the lower bound of the instance. The second new layer is added to handle constraints (50) since we have observed that these constraints are not tight with respect to the vehicle capacity and are generally satisfied. For this reason, we have devised a procedure to verify whether they are violated only at an integer solution. If the integer solution is found to violate constraints (50), these are added and the node is then reoptimized. Otherwise, the solution is feasible for the IRPPD. These two enhancements have significantly decreased the computational time.

Moreover, we have also observed that the separation algorithm of constraints (9) can be improved by adding a new set of variables representing the route of the vehicle in an undirected graph. By doing this, the separation algorithm runs on a graph half the size of the original one, looking for connected components and deriving maximum cuts over a much smaller network. Moreover, each subtour elimination cut derived for this new variable is equivalent to two cuts expressed in the original variables, one in each direction.

These two procedures employed to generate new cuts dynamically typically yield around 20,000 cuts only at the root node of the search tree. This number, although sizeable, is only a small fraction of the total number of cuts that could potentially be generated. By optimizing a problem with fewer constraints, we gain in speed since far fewer simplex iterations are needed. We note that a typical IRPPD instance still contains around 250,000 binary variables and 150,000 constraints after the two procedures just described have been applied, besides $O(2^{|\mathcal{R}|})$ subtour elimination constraints.

We have also observed that at the beginning of the optimization process, the problem is degenerate, i.e., several pivoting operations do not improve the value of the objective function. We have therefore taken advantage of the fact that during the optimization process, one can change the routing decisions while remaining feasible. Since routing variables x do not appear in the objective function, this solution has the same objective function value. Note that as long as the total route length is less than the shift duration, the solution remains feasible. We have also tested adding a small coefficient to the objective function to further minimize route length, and thus avoid degeneracy, but this option did not yield any significant advantage.

Finally, to obtain a clear view of the performance of our algorithms, and also to speed up the solution of each node, we have turned off the CPLEX cut generation.

4.3. Analysis of the Computational Experiments

We start our analysis by presenting the results obtained by the algorithm presented in §§3 and 4.2 on the set of 32 instances described in §4.1. Not all conceivable instances of the MD-IRPPD are feasible, but feasibility was always achieved with our data. The optimality gaps observed before variable fixing are rather large, with an average of 51%, a maximum of 62%, and a minimum of 45%. Even when 24 hours of computing time were allotted, these figures did not improve significantly: the average lower bound increased by 2% from 16,002 to 16,355, and the average gap went down from 51.31% to 50.23%. In what follows, we use the information obtained within six hours of computing time to allow for a direct comparison with the performance of the variable fixing procedure. To benchmark the quality of these solutions, we can compare to the instances of the IRP without pickup and delivery, which has recently been solved to optimality in Coelho and Laporte (2014) for comparable sizes. The problem at hand is much more complicated and clearly requires an additional effort to obtain good solutions.

We have then limited the flexibility of the algorithm by disallowing visits to RATMs that do not require a pickup or delivery to remain operational. This is achieved by setting to zero both parameters m and f of the clustering procedure. Obviously, all solutions

Table 3 **Full Results of the Branch-and-Cut Algorithm with the Clustering Procedure Set with $m = 0$ and $f = 0$**

Instance	$\lvert\mathcal{V}'\rvert$	Upper bound (without variable fixing)	Upper bound (with variable fixing)	Improvement due to variable fixing (%)	Lower bound (with variable fixing)	Optimality gap (%)
Inst-01	200	37,289	25,698	31.08	25,642	0.22
Inst-02	191	34,076	23,890	29.89	23,849	0.17
Inst-03	200	35,898	25,594	28.70	25,576	0.06
Inst-04	200	33,962	24,988	26.42	24,949	0.15
Inst-05	200	30,505	24,754	18.85	24,754	0.00
Inst-06	200	34,214	25,420	25.70	25,394	0.10
Inst-07	200	33,984	24,978	26.50	24,963	0.06
Inst-08	200	31,837	25,067	21.26	25,053	0.05
Inst-09	200	43,820	25,190	42.51	25,181	0.03
Inst-10	200	31,955	25,070	21.54	25,070	0.00
Inst-11	200	39,339	24,647	37.34	24,640	0.02
Inst-12	200	29,865	24,313	18.59	24,301	0.04
Inst-13	200	32,564	23,663	27.33	23,644	0.08
Inst-14	200	28,123	26,498	5.77	26,498	0.00
Inst-15	200	30,684	25,705	16.22	25,705	0.00
Inst-16	178	31,620	24,255	23.29	24,165	0.37
Inst-17	200	33,619	24,831	26.13	24,816	0.06
Inst-18	200	37,025	24,502	33.82	24,457	0.18
Inst-19	200	28,200	24,708	12.38	24,685	0.09
Inst-20	200	31,525	24,751	21.48	24,737	0.05
Inst-21	200	33,028	25,343	23.26	25,313	0.12
Inst-22	200	31,669	24,732	21.90	24,685	0.19
Inst-23	200	29,385	24,751	15.76	24,737	0.05
Inst-24	200	36,754	25,012	31.94	24,960	0.20
Inst-25	200	31,767	24,156	23.95	24,134	0.09
Inst-26	200	31,970	24,743	22.60	24,743	0.00
Inst-27	200	28,714	23,771	17.21	23,763	0.03
Inst-28	200	33,452	25,343	24.24	25,343	0.00
Inst-29	200	35,921	25,852	28.03	25,852	0.00
Inst-30	200	30,078	24,529	18.44	24,525	0.01
Inst-31	200	32,503	24,081	25.91	24,030	0.21
Inst-32	200	33,661	25,339	24.72	25,313	0.10
Average	199	33,094	25,114	24.11	24,859	0.09

obtained for this constrained version of the problem remain valid for the general case. In this situation, we have observed an average reduction of 24% in the upper bounds, even if the problem is more constrained. Note that this improvement was not observed even when the computing time was four times as long for the case without variable fixing. This can be explained by the fact that the branch-and-cut algorithm performs a search in a more promising area of the solution space. Under this scenario, seven instances were solved to optimality, and most of them yielded gaps below 0.1%. This is remarkable given the difficulty of the problem, which combines characteristics of the IRP and of the M-M PDP. These results are displayed in Table 3, where we present in columns "upper bound (without variable fixing)" and "upper bound (with variable fixing)" the upper bounds before and after applying the variable fixing phase, as well as the "improvement due to variable fixing (%)." Again, note that the latter solutions are valid for the general problem. The size of the vertex set $\mathcal{V}'$ is also listed: all instances but two contain 200 RATMs. These two exceptions resulted from the indivisibility of the number of RATMs (i.e., 6,377) over the 32 instances. In column "lower bound (with variable fixing)" we present the lower bound obtained when solving the problem after applying the variable fixing procedure, i.e., the constrained problem. We observe that these lower bounds are not valid for the general problem, which is less constrained. The column "optimality gap (%)" refers to the optimality gap between the upper and lower bounds computed after variable fixing. By increasing the number of periods m in which the algorithm can decide when to serve the RATMs, we obtain a problem that turns out to be very similar to the original one, and the same solutions are obtained when $m = 1$ and $m = 2$.

We have also tested different cases by allowing the algorithm to visit RATMs that do not necessarily need a visit, but that could help reallocate cash throughout the system, thus avoiding the need to return cash to the depot, which incurs a cost. Whenever we set the number f of close-by RATMs allowed to be visited to 1 or 2, but did not allow them to be visited in different periods, i.e., $m = 0$, the same solutions from the general

Table 4 **Summary of the Results When Changing the Parameter b, Controlling the Inventory Levels of Extra RATMs**

Variable fixing parameters		Upper bound (without variable fixing)	Upper bound (with variable fixing)	Improvement due to variable fixing (%)
$f = 1$, $m = 1$	$b = 30$	33,094	31,937	3.51
	$b = 50$	33,094	30,802	6.94
	$b = 70$	33,094	27,829	15.79
$f = 1$, $m = 2$	$b = 30$	33,094	31,949	3.47
	$b = 50$	33,094	31,944	3.49
	$b = 70$	33,094	30,840	6.80
$f = 2$, $m = 1$	$b = 30$	33,094	31,958	3.45
	$b = 50$	33,094	31,934	3.51
	$b = 70$	33,094	31,562	4.63
$f = 2$, $m = 2$	$b = 30$	33,094	31,952	3.46
	$b = 50$	33,094	31,941	3.49
	$b = 70$	33,094	31,926	3.54

case were obtained once again. This remained true irrespective of the value of the parameter b, i.e., the minimum inventory level of the extra RATMs.

Fixing the number f of extra RATMs and the number m of extra periods, and testing the effect of selecting RATMs based on their minimum average inventory level, i.e., $b = 30$, 50, and 70%, we have observed a clear trend: when the RATMs with higher inventory levels are allowed to be visited, better solutions are obtained. One possible explanation for this is that cash could be moved from these high inventory RATMs to those with shortages, hence decreasing both inventory costs and depot handling costs. Table 4 provides a summary of these results.

Finally, since all upper bounds obtained by different clustering procedures remain feasible for the general case, we are able to compile the best known upper bounds for each instance. These best-known upper bounds are presented in Table 5, along with the improvement with respect to the upper bounds obtained for the general case. We observe that solving the problem with our variable fixing procedure yields better upper bounds for all instances, with average improvements of 29.94% and attaining 47.17% in one case.

4.4. Estimated Benefits

Although the results of this study have not been implemented at the time of this writing, we can provide some estimates of its benefits. According to the data to which we had access, the expected business cost savings is estimated at about €10.1 million per year only in the Netherlands, when compared with current practice, mostly due to the reduction in the number of required armored trucks. Since RATMs are expected to become a market standard in the near future, our work anticipates this development by providing cash supply chain parties the means to immediately improve the replenishment operations. In addition to significant cost savings, we emphasize that our methodology yields extra potential benefits, which are harder to quantify in monetary terms. Indeed, we observe that the maximum amount of money that can be carried by an armored truck is limited by its (very expensive) insurance policy. The larger the average amount of cash

Table 5 **Best Results of the Branch-and-Cut Algorithm with the Clustering Procedure**

Instance	$\lvert\mathcal{V}'\rvert$	Upper bound (without variable fixing)	Upper bound (with variable fixing)	Improvement due to variable fixing (%)
Inst-01	200	37,289	22,538	39.56
Inst-02	191	34,076	23,890	29.89
Inst-03	200	35,898	25,594	28.70
Inst-04	200	33,962	24,988	26.42
Inst-05	200	30,505	24,754	18.85
Inst-06	200	34,214	25,420	25.70
Inst-07	200	33,984	24,978	26.50
Inst-08	200	31,837	25,067	21.26
Inst-09	200	43,820	25,190	42.51
Inst-10	200	31,955	25,070	21.55
Inst-11	200	39,339	24,647	37.35
Inst-12	200	29,865	24,313	18.59
Inst-13	200	32,564	23,663	27.33
Inst-14	200	28,123	26,498	5.77
Inst-15	200	30,684	25,705	16.22
Inst-16	178	31,620	18,607	41.15
Inst-17	200	33,619	21,677	35.52
Inst-18	200	37,025	22,235	39.95
Inst-19	200	28,200	24,708	12.38
Inst-20	200	31,525	19,204	39.08
Inst-21	200	33,028	18,861	42.89
Inst-22	200	31,669	24,732	21.90
Inst-23	200	29,385	19,238	34.53
Inst-24	200	36,754	19,417	47.17
Inst-25	200	31,767	19,271	39.34
Inst-26	200	31,970	19,721	38.31
Inst-27	200	28,714	23,771	17.21
Inst-28	200	33,452	20,028	40.13
Inst-29	200	35,921	21,381	40.48
Inst-30	200	30,078	24,529	18.45
Inst-31	200	32,503	24,081	25.91
Inst-32	200	33,661	21,010	37.58
Average	199	33,094	23,196	29.94

being transported, the more expensive the insurance becomes. Our algorithm achieves a 54% reduction in the in-transit inventory, when compared with the traditional 1-M-1 structure. Not only will the insurance premium decrease substantially, but a less expensive truck (due to reduced armor) could be deployed, CO_2 emissions can be reduced, and losses due to thefts should significantly go down.

5. Conclusions

We have introduced, modeled, and solved an inventory-routing problem with pickups and deliveries. We have been successful in solving a difficult application of the problem arising in the optimization of distribution and inventory management of cash in recirculation ATMs in the Netherlands. The problem was first modeled as a very large-scale nonlinear mixed-integer program. After applying a clustering procedure and linearizing some of the constraints, including some valid inequalities and fixing the values of several variables, we have solved the problem by branch and cut. Different settings of the variable fixing procedure were tested, and we have shown which ones are able to provide a good trade-off in terms of simplification of the problem and upper bound values. After variable fixing, we were able to solve exactly or to near optimality 32 instances involving up to 200 vertices. This size is similar to that of the largest IRP or M-M PDP instances that have been solved in the past. Our problem is obviously more difficult, because it combines these two features. Since recirculation ATMs are expected to become a market standard in the near future, our paper anticipates this development by providing cash supply chain parties the means to immediately improve the replenishment operations and yield significant cost savings.

Acknowledgments

The authors thank Calcul Québec for providing computing facilities. The authors also thank the associate editor and the anonymous referees for providing valuable comments on this paper. This work was partly supported by the Canadian Natural Sciences and Engineering Research Council [Grants 39682-10 and 2014-05764]. This support is gratefully acknowledged.

References

ABN ARMO Bank (2013) https://extra.abnamro.nl/maps/index_new.php.

Andersson H, Hoff A, Christiansen M, Hasle G, Løkketangen A (2010) Industrial aspects and literature survey: Combined inventory management and routing. *Comput. Oper. Res.* 37(9):1515–1536.

Angelelli E, Mansini R (2002) The vehicle routing problem with time windows and simultaneous pick-up and delivery. Klose A, Speranza MG, Van Wassenhove LN, eds. *Quantitative Approaches to Distribution Logistics and Supply Chain Management*, Lecture Notes Econom. Math. Systems, Vol. 519 (Springer-Verlag, Berlin Heidelberg), 249–267.

Anily S, Hassin R (1992) The swapping problem. *Networks* 22(4): 419–433.

Anily S, Gendreau M, Laporte G (2011) The preemptive swapping problem on a tree. *Networks* 58(2):83–94.

Archetti C, Bertazzi L, Hertz A, Speranza MG (2012) A hybrid heuristic for an inventory routing problem. *INFORMS J. Comput.* 24(1):101–116.

Archetti C, Bertazzi L, Laporte G, Speranza MG (2007) Branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transportation Sci.* 41(3):382–391.

Baldacci R, Hadjiconstantinou E, Mingozzi A (2003) An exact algorithm for the traveling salesman problem with deliveries and collections. *Networks* 42(1):26–41.

Benchimol M, Benchimol P, Chappert B, De La Taille A, Laroche F, Meunier F, Robinet L (2011) Balancing the stations of a self-service bike hire system. *RAIRO-Oper. Res.* 45(1):37–61.

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

Bianchessi N, Righini G (2007) Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. *Comput. Oper. Res.* 34(2):578–594.

Chemla D, Meunier F, Wolfler Calvo R (2013) Bike sharing systems: Solving the static rebalancing problem. *Discrete Optim.* 10(2): 120–146.

Christiansen M, Fagerholt K, Ronen D (2004) Ship routing and scheduling: Status and perspectives. *Transportation Sci.* 38(1):1–18.

Christiansen M, Fagerholt K, Nygreen B, Ronen D (2013) Ship routing and scheduling in the new millennium. *Eur. J. Oper. Res.* 228(3):467–483.

Christiansen M, Fagerholt K, Flatberg T, Haugen Ø, Kloster O, Lund EH (2011) Maritime inventory routing with multiple products: A case study from the cement industry. *Eur. J. Oper. Res.* 208(1):86–94.

Coelho LC, Laporte G (2013a) The exact solution of several classes of inventory-routing problems. *Comput. Oper. Res.* 40(2):558–565.

Coelho LC, Laporte G (2013b) A branch-and-cut algorithm for the multi-product multi-vehicle inventory-routing problem. *Internat. J. Production Res.* 51(23–24):7156–7169.

Coelho LC, Laporte G (2014) Improved solutions for inventory-routing problems through valid inequalities and input ordering. *Internat. J. Production Econom.* 155(1):391–397.

Coelho LC, Cordeau J-F, Laporte G (2012) The inventory-routing problem with transshipment. *Comput. Oper. Res.* 39(11):2537–2548.

Coelho LC, Cordeau J-F, Laporte G (2014) Thirty years of inventory-routing. *Transportation Sci.* 48(1):1–19.

Contardo C, Morency C, Rousseau L-M (2012) Balancing a dynamic public bike-sharing system. Technical report CIRRELT-2012-09, Montréal.

Cordeau J-F, Laporte G (2007) The dial-a-ride problem: Models and algorithms. *Ann. Oper. Res.* 153(1):29–46.

Dantzig GB, Fulkerson DR, Johnson SM (1954) Solution of a large-scale traveling-salesman problem. *Oper. Res.* 2(4):393–410.

Dauzère-Pérès S, Nordli A, Olstad A, Haugen K, Koester U, Olav MP, Teistklub G, Reistad A (2007) Omya Hustadmarmor optimizes its supply chain for delivering calcium carbonate slurry to European paper manufacturers. *Interfaces* 37(1):39–51.

Dell'Amico M, Righini G, Salani M (2006) A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Sci.* 40(2):235–247.

Dell'Amico M, Hadjicostantinou E, Iori M, Novellani S (2014) The bike sharing rebalancing problem: Mathematical formulations and benchmark instances. *Omega* 45:7–19.

Desaulniers G, Desrosiers J, Erdmann A, Solomon MM, Soumis F (2002) Vehicle routing problem with pickup and delivery. Toth P, Vigo D, eds. *The Vehicle Routing Problem*, Monographs Discrete Math. Appl., Vol. 9 (SIAM, Philadelphia), 225–242.

ECB (2013) ATM cash deposits at terminals located in the country with cards issued in the country. Technical report DA12, European Central Bank, http://sdw.ecb.europa.eu/.

Eilon S, Watson-Gandy CDT, Christofides N (1971) *Distribution Management: Mathematical Modeling and Practical Analysis* (Griffin, London).

Erdoğan G, Laporte G, Cordeau J-F (2010) A branch-and-cut algorithm for the non-preemptive capacitated swapping problem. *Discrete Appl. Math.* 158(15):1599–1614.

Erdoğan G, Laporte G, Wolfler Calvo R (2014) The static bicycle relocation problem with demand intervals. *Eur. J. Oper. Res.* 238(2):451–457.

Erdoğan G, Battarra M, Laporte G, Vigo D (2012) Metaheuristics for the traveling salesman problem with pickups, deliveries and handling costs. *Comput. Oper. Res.* 39(5):1074–1086.

Gendreau M, Laporte G, Semet F (1997) The covering tour problem. *Oper. Res.* 45(4):568–576.

Gendreau M, Laporte G, Vigo D (1999) Heuristics for the traveling salesman problem with pickup and delivery. *Comput. Oper. Res.* 26(7):699–714.

Geofabrik GmbH, OpenStreetMap Contributors (2013) Openstreetmap Netherlands shapefile. http://download.geofabrik.de/europe/netherlands-latest.shp.zip.

Gribkovskaia I, Halskau Ø, Laporte G, Vlček M (2007) General solutions to the single vehicle routing problem with pickups and deliveries. *Eur. J. Oper. Res.* 180(2):568–584.

Hernández-Pérez H, Salazar-González J-J (2003) The one-commodity pickup-and-delivery traveling salesman problem. Jünger M, Reinelt G, Rinaldi G, eds. *Combinatorial Optimization—Eureka, You Shrink!*, Lecture Notes Comput. Sci., Vol. 2570 (Springer-Verlag, Berlin Heidelberg), 89–104.

Hernández-Pérez H, Salazar-González J-J (2004a) A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Appl. Math.* 145(1):126–139.

Hernández-Pérez H, Salazar-González J-J (2004b) Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation Sci.* 38(2):245–255.

Hernández-Pérez H, Salazar-González J-J (2007) The one-commodity pickup-and-delivery traveling salesman problem: Inequalities and algorithms. *Networks* 50(4):258–272.

Hoff A, Gribkovskaia I, Laporte G, Løkketangen A (2009) Lasso solution strategies for the vehicle routing problem with pickups and deliveries. *Eur. J. Oper. Res.* 192(3):755–766.

ING Bank (2013) https://www.ing.nl/particulier/klantenservice/bankzaken-beginnen-of-beeindigen/geldautomaat-of-kantoor-zoeken/index.html.

Johnson DS, McGeoch LA (1997) The traveling salesman problem: A case study in local optimization.Aarts EHL, Lenstra JK, eds. *Local Search in Combinatorial Optimization* (Princeton University Press, Princeton, NJ), 215–310.

Miller CE, Tucker AW, Zemlin RA (1960) Integer programming formulation of traveling salesman problems. *J. Assoc. Comput. Machinery* 7(4):326–329.

Min H (1989) The multiple vehicle routing problem with simultaneous delivery and pick-up points. *Transportation Res. Part A: General* 23(5):377–386.

Mosheiov G (1994) The travelling salesman problem with pick-up and delivery. *Eur. J. Oper. Res.* 79(2):299–310.

Rabobank (2013) https://www.rabobank.nl/particulieren/servicemenu/naaruwbank/.

Rakke JG, Stålhane M, Moe CR, Christiansen M, Andersson H, Fagerholt K, Norstad I (2011) A rolling horizon heuristic for creating a liquefied natural gas annual delivery program. *Transportation Res. Part C: Emerging Tech.* 19(5):896–911.

Raviv T, Tzur M, Forma IA (2013) Static repositioning in a bike-sharing system: Models and solution approaches. *Eur. J. Transportation Logist.* 2(3):187–229.

Retail Banking Research Ltd. (RBR) (2014) Deposit automation and recycling. Technical report, RBR, London. http://www.rbrlondon.com/about/DA14_Press_Release_221214.pdf.

Shu J, Chou MC, Liu Q, Teo C-P, Wang I-L (2013) Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems. *Oper. Res.* 61(6):1346–1359.

Subramanian A, Battarra M (2013) An iterated local search algorithm for the travelling salesman problem with pickups and deliveries. *J. Oper. Res. Soc.* 64(3):402–409.

Subramanian A, Drummond LMA, Bentes C, Ochi LS, Farias R (2010) A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Comput. Oper. Res.* 37(11):1899–1911.

Toth P, Vigo D (2003) The granular tabu search and its application to the vehicle-routing problem. *INFORMS J. Comput.* 15(4):333–346.

Vidal T, Crainic TG, Gendreau M, Prins C (2013) A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Comput. Oper. Res.* 40(1):475–489.

Zachariadis EE, Tarantilis CD, Kiranoudis CT (2010) An adaptive memory methodology for the vehicle routing problem with simultaneous pick-ups and deliveries. *Eur. J. Oper. Res.* 202(2):401–411.

Zhao F, Li S, Sun J, Mei D (2009) Genetic algorithm for the one-commodity pickup-and-delivery traveling salesman problem. *Comput. Indust. Engrg.* 56(4):1642–1648.