Oper Res Int J (2019) 19:1–38
DOI 10.1007/s12351-016-0278-1

ORIGINAL PAPER

# Adaptive memory programming for the many-to-many vehicle routing problem with cross-docking

**Amalia I. Nikolopoulou**[1] · **Panagiotis P. Repoussis**[2,3] · **Christos D. Tarantilis**[1] · **Emmanouil E. Zachariadis**[1]

Received: 7 August 2016 / Revised: 3 November 2016 / Accepted: 7 November 2016 /
Published online: 18 November 2016

**Abstract** This paper presents a new generalized vehicle routing problem with a cross-dock. Basic features of the examined problem are the many-to-many relationship between the suppliers and customers, and the use of different vehicle fleets for performing the inbound and outbound routes. An adaptive memory programming method has been developed coupled with a Tabu Search algorithm. For generating new provisional solutions, elite subroutes with varying lengths are identified from the reference solutions and are used as building blocks, while multiple strategies are applied to maintain an effective interplay between diversification and intensification. Various computational experiments are conducted on existing as well as on new data sets with diverse features, regarding the geographic distribution of the nodes and the density of supplier-customer links. Overall, the proposed method performed very well and new best solutions have been found. Lastly, new insights regarding the impact of split options are reported.

**Keywords** Vehicle routing · Distribution · Pickup-and-delivery · Cross-dock · Heuristics

✉ Christos D. Tarantilis
tarantil@aueb.gr

[1] Department of Management Science and Technology, School of Business, Athens University of Economics and Business, Athens, Greece

[2] Department of Marketing and Communication, School of Business, Athens University of Economics and Business, Athens, Greece

[3] School of Business, Stevens Institute of Technology, Hoboken 07030, NJ, USA

## 1 Introduction

Transportation and product distribution costs often make up for a great portion of the total operating and logistics costs of a company either upstream or downstream in the supply chain. For this reason, it is important to focus on the design of distribution networks aiming to optimize product flows in pursuit of cost savings. In this context, cross-docking systems have been widely used in real-life distribution networks. Cross-docks can be seen as intermediate transshipment facilities that regulate the flow of products from origin to destination locations. The main role is to collect products coming from multiple sources, to sort and consolidate the products, and to arrange their shipping to the next echelon or to the final destinations. The main difference between modern cross-docking systems and traditional single or multi-echelon distribution network structures is that inbound products arrive at the cross-dock and are directly loaded to outbound vehicles. Therefore, the cross-docking system is lean and maintains little or no inventory. This paper presents a new problem, the so-called many-to-many vehicle routing problem with cross-docking, which according to our knowledge generalizes previous problem settings that appear in the literature. Key feature is that each customer requests products from multiple suppliers, while there is the restriction to visit each customer and supplier only once during the planning horizon.

The problem introduced in this paper has numerous practical applications. It is common for retailers located in urban areas to replenish their stock by receiving goods from multiple production warehouses located around the city limits. This situation emerges when a retailer receives different types of products from the same logistics provider, and not all of them are available at a single supplier location. This creates cargo consolidation opportunities, which can be exploited by the proposed problem setting with major cost savings. Furthermore, the single visit requirement at the supplier and customer nodes is essential in practice to minimize the administrative, handling, and various types of setup costs and times (Tarantilis et al. 2011). Finally, recent pro-environmental city logistics initiatives are aligned with the use of cargo consolidation operations outside urban areas, in pursuit of dispatching lighter, greener, or even electric vehicles in the city to minimize the impact on the urban environment (Zachariadis et al. 2015). Interested readers may also refer to the recent survey paper of Guastaroba et al. (2016) for vehicle routing problems with intermediate facilities.

Cross-docking has several advantages compared to other product distribution strategies both from the economic and environmental viewpoints. As it is discussed in Ma et al. (2011), logistics carriers often offer discount prices for truck-load orders; however, in practice it is common to have customers that raise less-than-truckload orders. In these situations, cross-docking offers a straightforward cargo consolidation opportunity, which enables products originated from various suppliers to be combined in truckload shipments and, thus, to be shipped with the discounted carrier rates. Moreover, this direct consolidation process avoids holding cargo in the intermediate nodes until sufficient products for forming a truckload shipment has arrived at the expense of inventory costs. In addition, the use of an intermediate node for consolidating products has been seen to reduce the total distance travelled and, thus, the transportation costs incurred compared to the direct shipping alternative, especially when the origin and destination points are located in different geographic regions and the intermediate node is located between these regions (Nikolopoulou et al. 2017). Another aspect that makes the cross-docking strategy an attractive transportation logistics practice is that it promotes short cycle times, rarely exceeding a single day. As a result, the service quality of the overall distribution system increases. In fact, cross-docking is considered to follow the Just-In-Time principle. Another favorable feature of the cross-docking strategy is that it enables the use of different vehicle fleets for handling the inbound and outbound transportation operations. This is of major importance when product flows are originated from areas outside the city center and the destination points are located in urban areas. In these cases, it is desirable to use large capacity heavy-duty vehicles for moving products from the suppliers to the cross-dock. On the contrary, it is often essential to use more compact vehicles for travelling within the core city road network. This will allow traversing narrow streets and avoiding traffic congestion, providing delivery service in confined customer spaces, and minimizing the environmental impact in urban areas.

Cross-docking practice has received research attention, especially regarding the operational aspects of the cross-dock. Focus is given on maximizing throughput by examining the layout of the cross-docks (Bartholdi and Gue 2004), the positioning of consolidated products inside the cross-dock (Vis and Roodbergen 2008), and the assignment of vehicles to dock doors (Tsui and Chang 1992; Cohen and Keren 2009). Another important research stream is devoted to location problems for the cross-dock facilities (Musa et al. 2010; Sung and Song 2003). In terms of the inbound/outbound distribution operations, there is a group of papers that call for the joint minimization of the total transportation and inventory costs incurred by cross-docking logistics systems. The common characteristic of these works is that vehicle trips are modeled as direct links between origin (suppliers) and destination (customers) nodes, or direct links between the cross-dock and the origin and destination locations. Under this setting, a vehicle cannot serve more than one supplier (or customer) order. These models aim to identify the optimal balance between the product flows directly shipped from suppliers to customers and the flows passing through intermediate consolidation facilities (Ma et al. 2011).

The routing component of cross-docking distribution networks has been initially examined by Lee et al. (2006). In this work the authors assume that vehicles may visit more than one service locations along their trips, subject to capacity constraints. The model introduced calls for the minimization of the routing costs involved for transferring products between suppliers and customers. It also assumes that all products collected from the suppliers are moved to the cross-dock, where they are consolidated and loaded to outbound vehicles for the shipment of these products to the customers. As far as the synchronization is concerned between the inbound and outbound trips, the authors assume that all pickup routes arrive at the cross-dock simultaneously. To solve the problem, Lee et al. (2006) propose a Tabu Search algorithm that is tested on instances with up to 50 nodes. Later, Liao et al. (2010) propose for the same problem an improved Tabu Search algorithm.

More recently, Wen et al. (2009) introduce a vehicle routing variant denoted as Vehicle Routing Problem with Cross-Dock (VRPCD) that extends the model of Lee et al. (2006). More specifically, time window constraints are imposed on both suppliers and customers, while the simultaneous vehicle arrival restriction at the cross-dock is dropped. Instead, more elaborate synchronization constraints are introduced to bridge the timing of inbound and outbound vehicle routes. Another important feature of the VRPCD model is that each transportation request is associated with a specific origin and destination location, where every pickup (supplier) node is paired with a single delivery (customer) node. Wen et al. (2009) propose an Adaptive Memory Procedure applied on problem instances involving up to 200 transportation orders. Later, Tarantilis (2013) studies the same problem and also proposes extensions with respect to the route structure and the consolidation activities at the cross-dock. In particular, an alternative scenario is examined where different vehicles are used for the pickup and delivery routes. This scenario implies that all products are unloaded to the cross-dock and reloaded onto the delivery leg vehicles for being transported to the customer locations. In addition, Tarantilis (2013) differentiates between closed and open route configurations for both inbound and outbound routes. A multi-restart Tabu Search algorithm is proposed for solving all aforementioned problem versions. Another work on the basic VRPCD is that of Morais et al. (2014). In this work an iterative local search algorithm is proposed for solving VRPCD instances of up to 500 customers. Another paper on the transportation costs incurred when products must be transferred between origin and destination points, is due to Nikolopoulou et al. (2017). In this work two alternative transportation strategies are compared, namely direct shipping and cross-docking. Several computational experiments are performed to gain insight on the role of temporal and operational parameters on the relative effectiveness of each strategy. Lastly, Santos et al. (2013) study a VRPCD variant with a hybrid network structure. Instead of requiring all products to pass through the cross-dock, the authors allow direct shipping between suppliers and customers. To solve the problem, denoted as the Pickup and Delivery Problem with Cross-docking (PDPCD), a branch and price algorithm is proposed and applied to instances with up to 30 transportation requests.

A real-life vehicle routing with cross-docking application is presented by Petersen and Ropke (2011). They assume that vehicles depart from the cross-dock and may visit both suppliers and customers before returning back to the cross-dock facility. Each vehicle may visit the cross-dock several times for unloading/loading operations and the cross-dock can serve as a short-term inventory holding point for the products. More specifically, the products of a given request may be pickedup along a morning route and shipped to the cross-dock, where they are held until shipped to the destination point by means of an evening vehicle trip. Dondo et al. (2011) develop an MILP formulation for scheduling the operations taking place in a single cross-dock facility. Their model aims to maximize the cross-dock productivity by jointly designing the inbound/outbound vehicle trips and assigning vehicles to cross-dock doors. The authors also consider cases where the number of cross-dock doors is lower than the vehicles used, and thus, there is not always a free door for every vehicle during the time horizon. Enderer (2014) introduces the combined

Dock-Door Assignment and Vehicle Routing Problem (DAVRP). This model aims to make decisions for the three distinct levels of a cross-dock system: (a) assignment of suppliers to inbound doors; (b) internal cross-dock routing plans and assignment of inbound products to outbound doors; and (c) vehicle routing plans for transporting the requested products to the customers using the outbound doors. Overall, the goal of the DAVRP is to minimize the total product handling and transportation costs. Finally, Grangier (2016) in his dissertation presents a matheuristic algorithm for the Vehicle Routing Problem with Dock Resource Constraints (VRPCD-DR).

The contribution of this paper is three-fold. First, a new generalized vehicle routing problem with a cross-dock is introduced. Most papers discussed above assume that every supplier is connected with only one customer (one-to-one VRPCD), while in most cases a common homogeneous fleet of vehicles is used for both inbound and outbound routes. Contrary, in the examined problem a many-to-many relationship between the suppliers and customers is considered, while different vehicle fleets are used to perform the pick and delivery routes. Second, an Adaptive Memory Programming method has been developed. Key characteristic is the mechanism for identifying and selecting elite components from the reference solutions. Particularly, the proposed mechanism assigns scores to all possible subroutes with varying length and gradually selects those with the highest score in terms of solution cost or diversity. A Tabu Search algorithm coupled with new long term memory structures is applied as the main optimization block for improving the quality of the new provisional solutions generated throughout the search process. Third, various computational experiments have been performed. Using well-known benchmark data sets for the one-to-one VRPCD, the proposed method proved to be efficient and effective compared to the current state-of-the-art. Furthermore, new best solutions have been found, while it seems that the method scales well with the problem size. On the other hand, the proposed method has been also tested on new data sets with diverse features regarding the geographic distribution of the network nodes as well as the density of supplier-customer links. This set of results provides several new insights regarding the effect of split options.

The remainder of the present paper is organized as follows: Sect. 2 describes the examined problem. Section 3 introduces the mathematical formulation for the problem. The proposed adaptive memory programming algorithm is presented in Sect. 4 and all its components are discussed. Next, Sect. 5 presents our computational study and findings on various benchmark data sets. Finally, Sect. 6 concludes the paper.

## 2 Problem description

Let $G = (V, A)$ be a graph, where $V$ is the node set and $A$ is the arc set. $V$ is composed of the cross-dock (node 0) and two node subsets $S'$ and $D'$, representing the supplier (pickup) and customer (delivery) locations, respectively. The arc set $A$ is considered to link every pair of nodes that belongs either to the set $S'' = S' \cup \{0\}$ (pickup arcs), or to the set $D'' = D' \cup \{0\}$ (delivery arcs).

Consider a set of transportation requests $O$. Each request $o \in O$ is associated with a supplier $i_o \in S'$, a customer $j_o \in D'$ and a demand $d_o$. It calls for the transportation of $d_o$ units of products from supplier $i_o$ to customer $j_o$. There is also the demand matrix $d_{ij}$ $(i \in S', j \in D')$ that represents the quantity of goods customer $j$ requires from supplier $i$. Each non-zero $d_{ij}$ $(i \in S', j \in D')$ actually refers to a transportation request contained in $O$. Note that the demand matrix may have multiple non-zero entries in each row, meaning that each supplier may send products to more than one customer. For each supplier $i$ $(i \in S')$, let $p_i$ be the total amount of products supplied by $i$. In addition, each column may also have multiple non-zero values, and this corresponds to cases where a customer receives products from more than one suppliers. There is also a demand $d_c$ associated with each customer $c$ $(c \in D')$.

The set of transportation requests are to be fulfilled by two distinct fleets of vehicles, denoted as $K_S$ and $K_D$. The vehicles from the set $K_S$ are considered to travel the $S''$ arcs, whereas vehicles from the $K_C$ set traverse only $D''$ arcs. The capacity of each vehicle $k \in K_S$ is equal to $Q_S$, whereas the capacity of each vehicle $l \in K_D$ is equal to $Q_D$. All pickup (or delivery) vehicles start and end their pickup (or delivery) routes at the cross-dock. In addition, the delivery routes should return to the cross-dock within a time limit $T$. Lastly, note that each supplier or customer must be visited only once.

The problem examined considers that vehicles of $K_S$, based at the cross-dock, are dispatched to the suppliers $S'$, in order to collect all products associated with the transportation orders in $O$. These vehicles return to the cross-dock where the products are unloaded, and appropriately consolidated to be loaded onto outbound vehicles. For this purpose, vehicles from the $K_D$ set are used. As soon as the products for each delivery route reach the cross-dock, the delivery trip is performed and the products are transported to the customer locations. All vehicles performing delivery trips return to the cross-dock.

The objective is to design the set of pickup and delivery routes that minimize the total travelled distance. The produced set of routes is subject to the following constraints:

(a) All inbound and outbound routes originate from the cross-dock and terminate at the cross-dock.
(b) Each supplier $i \in S'$ must be visited once by exactly one pickup route and, thus, all products from the corresponding transportation requests associated with $i$ must be assigned to the same vehicle.
(c) The total amount of products assigned to a pickup route should not exceed vehicle capacity $Q_S$, and similarly the total amount of products assigned to a delivery route should not exceed vehicle capacity $Q_D$.
(d) Each customer $i \in D'$ must be visited once by exactly one vehicle and, thus, all products from all transportation requests associated with $i$ must be assigned to the same vehicle.
(e) There is a maximum route duration $T$ which refers to the total time required for completing all pickup and delivery operations. More specifically, every delivery route must terminate at the cross-dock no later than time $T$. Note that we do not apply any duration length restrictions on individual pickup or

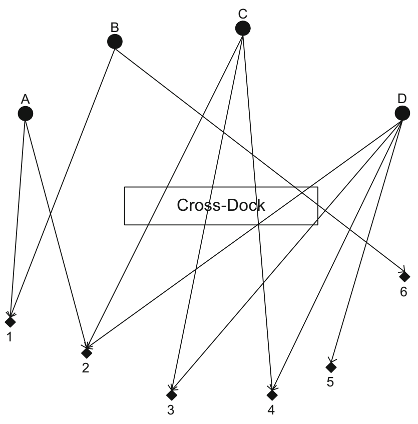
delivery routes, we only apply a maximum time limit $T$ that all delivery vehicles should be back at the cross-dock.

(f) A delivery vehicle cannot leave the cross-dock, until all products from the connected pickup routes arrive at the cross-dock.

Let a set $RS$ to denote all pickup routes. Each pickup route $r$ ($r \in RS$) visits a subset of suppliers $S_r$ and it is assumed to leave the cross-dock at time zero. The time $RT_r$ that the vehicle performing the pickup route $r$ is "released" (i.e., has finished its pickup route and has unloaded all products) at the cross-dock, is equal to the total traveling time plus the service time ($sp_s$) spent at the location of each supplier $s \in S_r$ plus the time spent for unloading operations ($u$) at the cross-dock. At the customer level, let $RD$ be the set of delivery routes. Each $r \in RD$ visits a subset of customers $D_r$. For every delivery route $r$, the total traveling time plus the service time ($sd_c$) spent at each customer location $c \in D_r$ is represented by $t_r$. In addition, let $DT_r$ denote the departing time from the cross-dock of a delivery route $r$. On this basis, the duration constraint ($e$) for every delivery route $r$ can be expressed as $DT_r + t_r \leq T$. Constraint ($f$) is associated with the temporal characteristics of the examined problem, and dictates the synchronization between the pickup and delivery routes. Necessary condition for the departure of a delivery route is that all products requested by each of the customers ($c \in D_r$) of the route must be available at the cross-dock in order to be sorted and loaded onto the delivery vehicle. Therefore, for every route $r \in RD$, $DT_r = a_r + l$, where $a_r$ is equal to the maximum release time at the cross dock of all the relevant pickup routes that contain the products requested by the $D_r$ customers. This is explained in detail in the example that follows (illustrated in Fig. 1). Finally, parameter $l$ denotes the time for unloading operations that take place at the cross-dock.

Figure 1 illustrates a problem instance with four suppliers and six customers. The link between a supplier and a customer node indicates that there is a transportation request for moving products between this node pair. Figure 1 also provides the demand matrix for the corresponding product flows. Supplier $A$ sends products to customers $1$ and $2$, Supplier $B$ send products to customers $1$ and $6$, and so on.

An example solution to the problem instance of Fig. 1 is shown in Fig. 2. Observe that there are two routes at the supplier level, and three vehicle routes at the customer level. The suppliers and customers are served by different vehicle types ($k1$, $k2 \in K_S$ and $k3$, $k4$, $k5 \in K_D$). The pickup route assigned to vehicle $k1$ visits suppliers $A$ and $B$ and collects the products requested by customers $1$, $2$, and $6$, whereas the pickup route performed by $k2$ visits suppliers $C$ and $D$ to pickup the products destined to customers $2$, $3$, $4$, and $5$. The capacity constraints for the inbound routes $k1$ and $k2$ ensure that $d_{A1} + d_{A2} + d_{B1} + d_{B6} \leq Q_S$ and $d_{C2} + d_{C3} + d_{C4} + d_{D2} + d_{D3} + d_{D4} + d_{D5} \leq Q_S$, respectively. For the outbound routes $k3$, $k4$, and $k5$, the capacity constraints are $d_{A1} + d_{B1} + d_{B6} \leq Q_D$, $d_{A2} + d_{C2} + d_{D2} + d_{C3} + d_{D4} \leq Q_D$, and $d_{C4} + d_{D4} + d_{D5} \leq Q_D$, respectively. Regarding the synchronization constraints discussed earlier, the release time for the products of delivery route $k3$ is $a_{k3} = RT_{k1} + l$. This is because $k3$ serves customers $1$ and $6$ who receive products from suppliers $A$ and $B$, both of them visited by $k1$. Similarly, the



| | | Customers | | | | | |
|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 |
| Suppliers | A | $d_{A1}$ | $d_{A2}$ | 0 | 0 | 0 | 0 |
| | B | $d_{B1}$ | 0 | 0 | 0 | 0 | $d_{B6}$ |
| | C | 0 | $d_{C2}$ | $d_{C3}$ | $d_{C4}$ | 0 | 0 |
| | D | 0 | $d_{D2}$ | $d_{D3}$ | $d_{D4}$ | $d_{D5}$ | 0 |

**Fig. 1** Problem instance with four suppliers and six customers

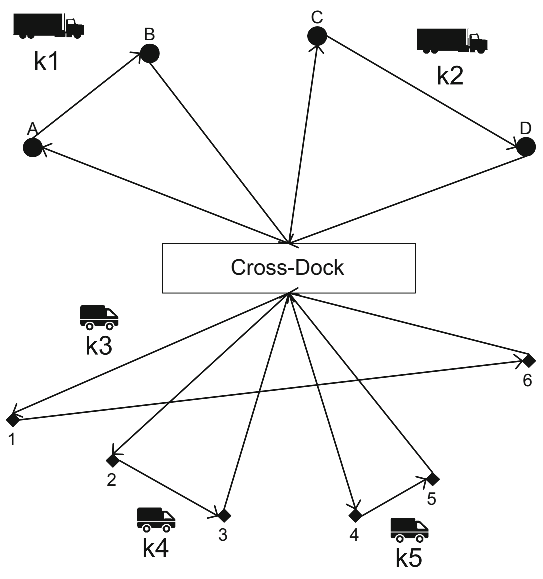


**Fig. 2** Example solution for the problem instance of Fig. 1

release times for routes *k4* and *k5* are $a_{k4} = \max\{RT_{k1}, RT_{k2}\} + l$ and $a_{k5} = RT_{k2} + l$, respectively.

The problem described above is closely related to the so-called Vehicle Routing Problem with Cross-Docking (VRPCD). As mentioned in the introduction section, the VRPCD has been introduced by Wen et al. (2009) and it assumes one-to-one relationships between suppliers and customers (i.e., each supplier is connected with only one customer and vice versa). Instead, in this paper a more generalized distribution network structure is considered with many-to-many relationships between the suppliers and the customers. Notably, many-to-many relationships can also be captured within a one-to-one VRPCD setting if we assume that collocated node copies are generated for every endpoint of the various transportation links. For example, consider the problem instance presented in Fig. 1. The many-to-many association between customers and suppliers can be translated into multiple one-to-one relations by generating a supplier-customer pair for each non-zero element in the demand matrix. As such, the resulting problem will contain 11 suppliers and 11 customer nodes. Besides the fact that the dimension of the problem increases, this setting would permit multiple visits per node (i.e., copies of the original supplier or customer nodes can be assigned to different vehicle routes). In practice, this is often undesirable given that the resulting service times from the multiple visits are typically higher due to the multiple setup times (e.g. for security procedures or for exchange of paperwork). On the contrary, in our model each supplier and customer node is served only once by exactly one vehicle.

## 3 Mathematical formulation

In this section we present and describe in detail the mathematical model for the many-to-many Vehicle Routing Problem with Cross-docking (many-to-many VRPCD) as described above. In addition to the notation introduced in Sect. 2, let us denote the set of pickup nodes by $S = \{0, 1,\ldots, n + 1\}$ and the set of delivery nodes by $D = \{0, 1,\ldots, n'+1\}$. The cross-dock is represented by the nodes 0, $n + 1$ and $n'+1$, where the nodes 0 and $n + 1$ in $S$ represent the starting and ending points for the pickup routes, while the nodes 0 and $n'+1$ in $D$ represent the starting and ending points for the delivery routes, respectively. Note that these four nodes are associated with a zero amount of supply/demand and with zero service time. The set $E$ denotes all the feasible arcs for the pickup operations. It consists of the arcs $\{(i, j)$: $i, j \in S, i \neq j\}$. The set $E'$ denotes all the feasible arcs for the delivery operations and consists of the arcs $\{(h, f)$: $h, f \in \mathrm{D}, h \neq f\}$. Each arc $(i, j) \in E$ is associated with a known non-negative distance $cs_{ij}$, that represents the travel distance from pickup node $i$ to pickup node $j$. Similarly, each arc $(h, f) \in E'$ is associated with a known non-negative distance $cc_{hf}$, that represents the travel distance from delivery node $h$ to delivery node $f$. In addition, $M$ is an arbitrarily large constant.

Indices and sets

| | |
|---|---|
| $i, j$ | Index for pickup nodes (suppliers) |
| $h, f$ | Index for delivery nodes (customers) |
| $k$ | Index for pickup vehicles |
| $l$ | Index for delivery vehicles |

Parameters

| | |
|---|---|
| $K_S$ | Number of available pickup vehicles |
| $K_D$ | Number of available delivery vehicles |
| $Q_S$ | Capacity of the pickup vehicles |
| $Q_D$ | Capacity of the delivery vehicles |

| | |
|---|---|
| $cs_{ij}$ | Travel distance between pickup node $i$ and pickup node $j$ ($(i, j)$: $i, j \in S$, $i \neq j\}$) |
| $cc_{hf}$ | Travel distance between delivery node $h$ and delivery node $f$ ($(h, f)$: $h, f \in D$, $h \neq f\}$) |
| $CV_{ih}$ | 1, if pickup node $i$ ($i \in S \cap \{0, n+1\}$) supplies (is connected to) delivery node $h$ ($h \in D \cap \{0, n'+1\}$); 0, otherwise |
| $p_i$ | Amount supplied by pickup node $i$ ($i \in S$) |
| $d_h$ | Amount delivered to delivery node $h$ ($h \in D$) |
| $sp_i$ | Service time spent at pickup node $i$ ($i \in S$) |
| $sd_h$ | Service time spent at delivery node $h$, ($h \in D$) |
| $u$ | Time for unloading operations at the cross-dock |
| $l$ | Time for loading operations at the cross-dock |
| $T$ | Maximum route duration |

Binary variables

| | |
|---|---|
| $x_{ij}^k$ | 1, if vehicle $k$ ($k \in K_S$) travels from pickup node $i$ to pickup node $j$ ($i, j \in S$); 0, otherwise |
| $z_{hf}^l$ | 1, if vehicle $l$ ($l \in K_D$) travels from delivery node $h$ to delivery node $f$ ($h, f \in D$); 0, otherwise |
| $y_i^k$ | 1, if pickup node $i$ ($i \in S$) is serviced by vehicle $k$ ($k \in K_S$); 0, otherwise |
| $v_h^l$ | 1, if delivery node $h$ ($h \in D$) is serviced by vehicle $l$ ($l \in K_D$); 0, otherwise |

Continuous non-negative variables

| | |
|---|---|
| $tp_i^k$ | Time at which vehicle $k$ leaves pickup node $i$ ($i \in S$, $k \in K_S$) |
| $td_h^l$ | Time at which vehicle $l$ leaves delivery node $h$ ($h \in D$, $l \in K_D$) |
| $RT_k$ | Release time of pickup vehicle $k$ at the cross-dock ($k \in K_S$) |
| $DT_l$ | Starting time of delivery vehicle $l$ at the cross dock ($l \in K_D$) |

Below we present the mathematical formulation of the many-to-many VRPCD as a mixed-integer linear programming model. The objective is to minimize total transportation costs.

$$\text{Minimize} \sum_{(i,j)\in E} \sum_{k\in K_S} cs_{ij} \cdot x_{ij}^k + \sum_{(h,f)\in E'} \sum_{l\in K_D} cc_{hf} \cdot z_{hf}^l \tag{3.1}$$

Subject to

$$\sum_{j\in S: i\neq j} x_{ij}^k = y_i^k \quad \forall i \in S \cap \{0, n+1\}, k \in K_S \tag{3.2}$$

$$\sum_{f\in D: h\neq f} z_{hf}^l = v_h^l \quad \forall h \in D \cap \{0, n'+1\}, l \in K_D \tag{3.3}$$

$$\sum_{k\in K_S} y_i^k = 1 \quad \forall i \in S \cap \{0, n+1\} \tag{3.4}$$

$$\sum_{l\in K_D} v_h^l = 1 \quad \forall h \in D \cap \{0, n'+1\} \tag{3.5}$$

$$\sum_{j\in S\cap\{0\}} x_{0j}^k = 1 \quad \forall k \in K_S \tag{3.6}$$

$$\sum_{f\in D\cap\{0\}} z_{0f}^l = 1 \quad \forall l \in K_D \tag{3.7}$$

$$\sum_{i\in S\cap\{n+1\}} x_{i,n+1}^k = 1 \quad \forall k \in K_S \tag{3.8}$$

$$\sum_{h\in D\cap\{n'+1\}} z_{h,n'+1}^l = 1 \quad \forall l \in K_D \tag{3.9}$$

$$\sum_{i\in S} x_{i0}^k = 0 \quad \forall k \in K_S \tag{3.10}$$

$$\sum_{h\in D} z_{h0}^l = 0 \quad \forall l \in K_D \tag{3.11}$$

$$\sum_{j\in S} x_{n+1,j}^k = 0 \quad \forall k \in K_S \tag{3.12}$$

$$\sum_{f\in D} z_{n+1,f}^l = 0 \quad \forall l \in K_D \tag{3.13}$$

$$\sum_{j\in S:i\neq j} x_{ij}^k = \sum_{j\in S:i\neq j} x_{ji}^k \quad \forall i \in S \cap \{0, n+1\}, k \in K_S \tag{3.14}$$

$$\sum_{f\in D:h\neq f} z_{hf}^l = \sum_{f\in D:h\neq f} z_{fh}^l \quad \forall h \in D \cap \{0, n'+1\}, l \in K_D \tag{3.15}$$

$$\sum_{i\in S} p_i \cdot y_i^k \le Q_S \quad \forall k \in K_S \tag{3.16}$$

$$\sum_{h\in D} d_h \cdot v_h^l \le Q_D \quad \forall l \in K_D \tag{3.17}$$

$$tp_{\mathrm{i}}^{\mathrm{k}} + cs_{ij} + sp_j - tp_{\mathrm{j}}^{\mathrm{k}} - M\left(1 - x_{ij}^k\right) \le 0 \quad \forall i \in S, j \in S \cap \{0, n+1\}, k \in K_S \tag{3.18}$$

$$tp_{\mathrm{i}}^{\mathrm{k}} + cs_{i,n+1} + u - RT_k - M\left(1 - x_{i,n+1}^k\right) \le 0 \quad \forall i \in S \cap \{n+1\}, k \in K_S \tag{3.19}$$

$$RT_k \le CV_{ih}DT_l + M\left(-y_i^k - CV_{\mathrm{ih}}v_h^l + 2\right) \quad \forall i \in S, k \in K_S, h \in D, l \in K_D \tag{3.20}$$

$$DT_l + l + cc_{0f} + sd_f - td_f^l - M\left(1 - z_{0f}^1\right) \le 0 \quad \forall f \in D \cap \{0, n' + 1\}, l \in K_D \tag{3.21}$$

$$td_h^l + cc_{hf} + sd_f - td_f^l - M\left(1 - z_{hf}^l\right) \le 0 \quad \forall (h,f) \in D \cap \{0, n' + 1\}, l \in K_D \tag{3.22}$$

$$td_h^l + cc_{h,n'+1} - td_{(n'+1)}^l - M\left(1 - z_{h,n'+1}^l\right) \le 0 \quad \forall h \in D \cap \{0\}, l \in K_D \tag{3.23}$$

$$td_{(n'+1)}^l \le T \quad \forall l \in K_D \tag{3.24}$$

$$tp_i^k \le My_i^k \quad \forall i \in S, k \in K_S \tag{3.25}$$

$$td_h^l \le Mv_h^l \quad \forall h \in D, l \in K_D \tag{3.26}$$

$$RT_k - M\left(1 - x_{0(n+1)}^k\right) \le 0 \quad \forall k \in K_S \tag{3.27}$$

$$DT_l - M\left(1 - z_{0(n'+1)}^l\right) \le 0 \quad \forall l \in K_D \tag{3.28}$$

$$\sum_{k \in K_S} y_0^k = 0 \tag{3.29}$$

$$\sum_{k \in K_S} y_{n+1}^k = 0 \tag{3.30}$$

$$\sum_{l \in K_D} v_0^l = 0 \tag{3.31}$$

$$\sum_{l \in K_D} v_{n+1}^l = 0 \tag{3.32}$$

The objective (3.1) is to minimize the total distance traveled. Constraints (3.2) and (3.3) ensure that each node is visited once by one vehicle. Constraints (3.4) and (3.5) guarantee that each request is pickedup or delivered in only one pickup or delivery route, respectively. Constraints (3.6)–(3.10) guarantee that every route leaves the corresponding starting point and returns to the ending point. There are also constraints to guarantee that routes do not return to the starting points nor leave the ending points (3.11)–(3.13). Constraints (3.14) and (3.15) are flow conservation constraints. Constraints (3.16) and (3.17) ensure that for each vehicle, the load on the pickup route and the delivery route does not exceed the vehicle capacity. Constraint (3.18) computes the time a pickup vehicle $k$ arrives at pickup node $j$ after visiting pickup node $i$. Constraint (3.19) computes the time at which vehicle $k$ arrives at the cross-dock from pickup route for unloading. Constraint (3.20) is the connectivity constraint among pickup and delivery routes.

This constraint states that the delivery route $l$ of an order $h$ cannot start until the corresponding request has been picked up, arrived at the cross-dock and been unloaded from pickup vehicle $k$. Next (3.21)–(3.24) are time constraints for the delivery routes. In particular, constraint (3.21) computes the arrival time at the first delivery node visited on route $l$. Constraint (3.22) computes the arrival time at delivery node $j$ which is visited after node $i$ on route $l$. Constraint (3.23) computes the arrival time of vehicle $l$ at the cross-dock and it is ensured that this time is within the maximum route duration limit $T$ (3.24). We have also added constraints (3.25) and (3.26) to keep the starting time of pickup/delivery nodes not visited on a pickup/delivery route at 0. Similarly, constraints (3.27) and (3.28) keep the finishing/starting time of a not-driven pickup/delivery route at 0. Constraints (3.29)–(3.32) ensure that the starting and ending points (cross-dock) are not serviced by any pickup or delivery vehicle.

The above MILP model was solved using IBM ILOG CPLEX 12.5. Computational experiments for small scale problem instances and implementation details are reported in Sect. 5.3.

## 4 Solution method

An Adaptive Memory Programming (AMP) metaheuristic algorithm has been developed for solving the many-to-many VRPCD. AMP is a general-purpose solution framework that focuses on the exploitation of strategic memory structures (Glover 1997). The adaptive memory rationale was introduced by Taillard et al. (2001). The main goal is to identify solution characteristics frequently found in the search history and use this information to produce high-quality solutions. AMP frameworks have also been developed to efficiently solve vehicle routing problems (Repoussis and Tarantilis 2010; Tarantilis 2005). Gounaris et al. (2014) presents an AMP framework to address the robust capacitated vehicle routing problem. Cardona-Valdés et al. (2014) have developed an AMP framework to address the design of a two-echelon production distribution network. Lastly, Paraskevopoulos et al. (2016) are among the first to present an AMP framework for the Resource-Constrained Project Scheduling Problem (RCPSP).

Following the earlier works of Taillard et al. (2001), Tarantilis (2005) and Gounaris et al. (2014), the proposed framework consists of two phases. During the initialization phase, the goal is to generate a reference set of high quality solutions. Subsequently, the exploitation phase is triggered, that incorporates adaptive memory mechanisms for identifying elite solution components that are used to generate new provisional solutions. In both phases, a Tabu Search algorithm is employed as a local search method applied for further improvement and for reaching high quality local optimal solutions.

**Algorithm 1**. AMP framework

```
Input parameters ($\mu, \lambda, \xi, \zeta, \vartheta, t_{lm}$)
1   P ← $\emptyset$, $x^B$ ← $\emptyset$
    //Initialization phase
2   while ($|P| < \mu$)
3       $x_0$ ← Send one vehicle to each supplier and customer node
4       $x'$ ← Generalized savings heuristic ($x_0, \lambda$)
5       $x''$ ← Tabu search ($x', \xi, \zeta$)
6       if f($x''$) < f($x^B$) then
7           $x^B \leftarrow x''$
8       end if
9       $P \leftarrow P \cup x''$
10  end while
    //Exploitation phase
11  while ( CPU time < $t_{lm}$) do
12      $x^{elite}$ ← Selection of Elite components ($P, \vartheta$)
13      $x'$ ← Generalized savings heuristic ($x^{elite}, \lambda$)
14      $x''$ ← Tabu search ($x', \xi, \zeta$)
15      if f($x''$) < f($x^B$) then
16          $x^B \leftarrow x''$
17      end if
18      $P$ ← Reference Set Update Method ($P, x''$)
19  end while
20  return $x^B$
```

The pseudocode of the proposed AMP algorithm is depicted in Algorithm 1. Starting from an empty reference set $P$, $\mu$ high quality solutions are generated during the initialization phase (Lines 2–10). A generalized savings heuristic (Line 4) is employed to generate random initial solutions. This is achieved via a greedy randomized mechanism controlled via parameter $\lambda$ (see Sect. 4.1). All initial solutions are locally improved via a Tabu Search metaheuristic algorithm (Line 5) that is controlled through parameters $\xi$ and $\zeta$, described in Sect. 4.4. Subsequently, the exploitation phase is triggered (Lines 11–19). During this phase, elite solution components from the reference set $P$ are extracted according to a deterministic set of criteria that make use of the parameter $\theta$. Given these components, an intermediate solution $x^{elite}$ is generated (Line 12) based on which the final provisional solution $x'$ is constructed via the saving construction heuristic (Line 13). The criteria of extracting solution components and generating the provisional solution $x'$ are described in Sect. 4.3. Next, the provisional solution $x'$ is further improved by the Tabu Search metaheuristic algorithm (Line 14), and after this local search process is completed the improved solution $x''$ is used to update the reference set $P$ (Line 18). The improvement phase terminates whenever a pre-specified time limit $t_{lm}$ is reached (Line 11), and the best feasible solution $x^{\mathrm{B}}$ encountered throughout the AMP framework (Line 20) is returned.

## 4.1 Greedy randomized savings construction heuristic algorithm

A generalized savings heuristic algorithm similar to that proposed in Gounaris et al. (2014) is adopted in this paper. The construction framework starts from an initial intermediate feasible solution ($x_0$). During the initialization phase, this is a solution where one pickup or delivery vehicle is assigned to each pickup or delivery node, respectively. On the other hand, during the exploitation phase, this is a solution where one pickup or delivery vehicle is assigned to an elite component/subset of connected pickup or delivery nodes, respectively. In any case, all pickup and delivery routes originate and terminate at the cross-dock, while all nodes are assigned to vehicle routes. Furthermore, only feasible combinations of pickup and delivery routes are considered with respect to the maximum route duration and synchronization constraints. Lastly, note that no limit is imposed on the total number of pickup or delivery routes.

Given the initial intermediate solution, at each iteration two pickup (or delivery) routes are selected and merged according to the following generalized savings scheme. Let two routes $r = \{0, \ldots, i, j, \ldots, 0\}$ and $r' = \{0, i', \ldots, j', 0\}$. The savings function evaluates to $c_{ij} + c_{0i'} + c_{j'0} - c_{ii'} - c_{jj'}$, where $c_{ij}$ is the travel distance for traversing arc $(i, j)$. The merging of route pairs is repeated either until all merging combinations in all possible positions produce negative savings or no feasible merging combination can be found with respect to the capacity, duration and synchronization constraints. Note that both pickup and delivery routes are consider during the construction process.

In an effort to diversify the reference set $P$, a probabilistic mechanism is added during the solution construction process similar to that proposed by Tarantilis et al. (2013). Particularly, a restricted candidate list of feasible merging combinations between pickup or delivery routes with positive saving is maintained at each iteration, which contains the route pairs with the highest saving. To that end, one route pair from the list is selected at random, and the corresponding vehicle routes are merged. In our implementation, the restricted savings list is fixed to a predefined size $\lambda$.

## 4.2 Reference set update method

In both initialization and exploitation phases the reference set $P$ contains a maximum number of solutions $\mu$. It is well documented in the literature that one essential element for the performance of population-based approaches is to maintain the diversity of the population during the search. This need intensifies for methods with relatively small populations, such as scatter search, path relinking and AMP approaches. In the proposed solution method, the reference set $P$ is updated with solutions that are not only attractive in terms of the total transportation costs, but also in terms of diversity (measured in terms of total number of different arcs with respect to the best encountered solution). From the implementation viewpoint, an elitist strategy is adopted. Let $x$ denote the candidate solution for insertion into $P$, $x^r$ any reference solution of $P$, and $x^B$ and $x^W$ the best and the worst solutions of $P$, respectively. If $x$ performs better than $x^B$ in terms of cost, $f(x) < f(x^B)$, then $x$ replaces $x^W$; otherwise, if $f(x) < f(x^r)$, then $x$ replaces $x^r$ only if $d(x, x^B) > d(x^r, x^B)$, where $d(x, x^B)$ represents the total number of different arcs between solutions $x$ and $x^B$.

## 4.3 Generation of provisional solutions

The goal of the exploitation phase is to generate new provisional solutions by combining elite components encountered in the reference solutions. An elite component refers to a subroute, i.e., an ordered subset of nodes (either pickup or delivery nodes) that appears frequently in the reference solutions. The first step is to select and isolate these subroutes considering both pickup and delivery routes. Let $l_z$ denote the length, i.e., the number of nodes (either suppliers or customers), of a subroute $z$. A component $z$ may be as large as a complete route and as short as an individual arc connecting two nodes, therefore $l_z \geq 2$ (i.e., singleton nodes do not qualify as components).

During the scoring and selection process, the final set of extracted components should consist of non-overlapping node sets in order to be suitable for recombination in a new solution. Furthermore, we always make sure that it is feasible to send one vehicle to serve all selected pickup and delivery subroutes with respect to the capacity, route duration and synchronization constraints. For scoring the subroutes, the metric introduced by Gounaris et al. (2014) is adopted and each elite component $z$ is assigned a score $H_z$ as follows: $H_z = \frac{\left(\sum_{x\in P} w_x I_{x,z}\right)}{(1-\vartheta)^{l_z-2}}$, where $w_x$ is the solution score and $I_{x,z}$ is a binary indicator taking the value of 1, if the subroute appears in the solution $x$, and 0 otherwise. The term $(1-\vartheta)^{l_z-2}$ quantifies the adoption of longer subroutes at the expense of their shorter subsets, where $\theta \in (0, 1)$. Two strategies are followed to calculate the weight factor $w_x$ that are used interchangeably with equal selection probability. The first strategy takes into account the dissimilarity $ds(x, x')$, i.e., number of different arcs, between two solutions $x$ and $x'$ and is calculated as follows [see also Tarantilis et al. (2013): $ds(x, x') = \sum_{(i,j)\in A} h_{ij}$, where $h_{ij}$ is a binary indicator taking the value of 1, if $(i, j)$ is an edge of either solution $x$ or $x'$ (but not both), and 0 otherwise]. In this strategy, the weight factor is calculated as $w_x = ds(x, x^B)/max_{x^r\in P} d(x^r, x^B)$, where $x^B$ is the current best reference solution. Contrary, the second strategy takes into account only the cost. In this case, the weight factor $w_x$ is calculated as: $w_x = (max_{x^r\in P} f(x^r) - f(x))/(max_{x^r\in P} f(x^r) - min_{x^r\in P} f(x^r))$.

Having scored each possible delivery and pickup subroute extracted from all reference solutions, the subroutes are sorted according to their score with decreasing order. Once a subroute is selected from the list as elite component, all remaining subroutes that share at least one supplier or customer with the selected subroute are removed from the sorted subroute list. The selection is repeated until either the list is empty or there is no way with the remaining subroute to generate a feasible solution. A feasible intermediate solution ($x^{elite}$) is then generated by assigning a vehicle to each of the selected elite components as well as to any singleton supplier or customer that does not participate in any of the selected elite components. At this point, the greedy randomized savings construction heuristic algorithm described in Sect. 4.1 is employed to obtain the new improved provisional solution. Notably, the provisional solution will often provide different combinations of the elite components since it goes through the route merging procedure.

## 4.4 Tabu search

All generated solutions are subject to local search via a Tabu Search metaheuristic algorithm. The latter explores the solution space by performing search trajectories based on edge-exchange moves from a current solution to the best admissible neighboring solution. The overall procedure iterates for a number of iterations and the best solution is returned. Below, each major component is described.

### *4.4.1 Neighborhood structures*

The edge-exchange neighborhood structures considered are 0–1 node relocation, 1–1 node exchanges and 2-Opt exchanges (Zachariadis and Kiranoudis 2010). A lexicographic neighborhood evaluation scheme is adopted, while only feasible solutions are considered. The size of 0–1 Relocation and 2-Opt is $O(n^2)$, while the size of 1–1 Exchange is $O(n^4)$. All of them involve a constant number of edge exchanges.

The 0–1 node relocation removes any supplier (or customer) node from its current position, and reinserts this node into any other available position. Note that both intra- and inter-route node relocation moves are considered. The 1–1 exchanges any two supplier (or customer) nodes served by two different pickup (or delivery) routes. More specifically, the first supplier (or customer) node can be pushed in any available insertion position in the route originally serving the second supplier (or customer) node. Similarly, the second supplier (or customer) node can be inserted in any available insertion position combination of the route originally serving the first supplier (or customer) node. Note that both intra- and inter-route exchange can be performed. Obviously, the two nodes involved in the exchange must be of the same type (pickup or delivery). Lastly, the 2-Opt exchange removes a pair of edges and replaces them by a new pair in the solution. 2-Opt can be applied on the same route (the visiting sequence of a subroute is reversed) and between a pair of routes (subroute segments each including the cross-dock are swapped).

### *4.4.2 Short-term memory, aspiration conditions and diversification mechanism*

Our Tabu Search framework operates according to the best admissible local move scheme. Specifically, all neighborhood structures of the current incumbent solution are exhaustively explored, and the highest quality feasible neighboring solution is selected at each iteration. To avoid an over-intensified search, a diversification mechanism is introduced similar to the Attribute Hill Climber presented by Whittley and Smith (2004). Each arc $(i, j) \in A$ is associated with a threshold tag $t_a$. Every time a move $m$ is applied to a solution $x$ with objective value $f(x)$, the threshold tags of the eliminated arcs ($E_m$) are set equal to $f(x)$, i.e., $t_a = f(x)$, $\forall\ a \in E_m$. Any move $m$ that forms a solution $x'$ is considered admissible only if the cost tags of the generated arcs ($C_m$) exceed the objective value of the modified solution $x'$, i.e., $t_a > f(x')$, $\forall\ a \in C_m$. Note that the threshold tags are re-initialized to a large value after a number of iterations $\xi$, while a maximum number of iterations $\zeta$ is imposed as termination condition.

## 5 Computational experiments

### 5.1 Experimental settings and parameter values

Various computational experiments have been performed to examine the inherent characteristics of the many-to-many VRPCD, as well as to access the performance of the proposed AMP solution method. At first, focus is given on the one-to-one VRPCD; the benchmark data set of Wen et al. (2009) and Morais et al. (2014) are used to compare our method with respect to the current state-of-the-art approaches of the literature. The results of the comparative performance analysis are reported in Sect. 5.2. Next, attention is given on the new generalized problem setting, and new benchmark data sets are generated with up to 200 nodes considering different geographical distributions and varying supplier-to-customer connection densities. The data generation method of the new benchmark data sets as well as the discussion of the computational results are reported in Sects. 5.3 and 5.4.

Regarding the experimental setup, the proposed AMP method was implemented in C# and the MILP was solved using the CPLEX 12.5 callable library. All experiments for each problem instance were executed on a single core of a computer system equipped with an Intel Xeon CPU E5-2650 v2 (2.60 GHz) and 16 GB of RAM under Windows Server 2012. A CPU time limit of 6 h was imposed for the execution of CPLEX. During the branch and cut process nodes were selected according to the best-bound criterion and all CPLEX heuristics were disabled. Note also that the best found solution from the AMP metaheuristic algorithm was used as the initial MIP start solution (upper bound) for CPLEX. On the other hand, for the AMP all computational times reported are in seconds. Unless otherwise stated, 10 simulation runs are performed for each problem instance, and the best out of the 10 runs is reported in the tables. All best solutions obtained for the new benchmark data sets are reported in "Appendix".

Regarding the AMP parameter settings, a single set has been used for all experiments. Although better results could be in principle obtained by varying the parameters for each individual problem instance, we selected to adopt after minor tuning the parameter set that seemed to perform well for the majority of problems. In particular, we set $\lambda = 12$, $\mu = 12$, $\theta = 0.4$, $\xi = 2$*(total number of requests) and $\zeta = 600$. Unless otherwise stated, the maximum time limit $t_{lim}$ for each run is set to 3600 s.

## 5.2 Assessment of the method and discussion of results for the one-to-one VRPCD

At first, the proposed AMP method is applied for the one-to-one VRPCD using the benchmark data set of Wen et al. (2009) with up to 200 requests as well as the data set of Morais et al. (2014) involving up to 500 requests. Previous algorithms solve these instances considering variable times for the unloading and loading operations taking place at the cross-dock as well as time windows for both suppliers and customers. In order to ensure a common and fair basis for comparison, we have modified our AMP algorithm accordingly to take into account all necessary operational realities, and in particular, we followed the so-called CS1 consolidation scenario for the cross-dock operations as described by Tarantilis (2013). This scenario assumes that the same vehicle fleet is utilized for both pickups and deliveries. In this case, only the pickup products that will be delivered to their corresponding customers by another vehicle will be unloaded from the pickup vehicles at the cross-dock, and subsequently they must be sorted and reloaded to the corresponding delivery vehicles. In addition, this scenario assumes that both suppliers and customers have to be serviced within predefined time windows.

The computational results obtained for the Wen et al. (2009) and Morais et al. (2014) instances are summarized in Tables 1 and 2, respectively. Our algorithm exhibits a reliable performance. In particular, for the Wen et al. (2009) instances, it managed to find one new best solution (50b instance). For the rest of the instances, the algorithm is close to the best known solutions (*BKS*)—which are highlighted in bold font- and it has a worst case performance of 1.00%, with an average deviation from the *BKS* of 0.24%. No significant variations with respect to the solution cost are observed over the 10 runs for each instance. Finally, it is worth noticing that the proposed method scales-up well in terms of solution quality and computational times with respect to the problem size. Better are the figures for the larger data set of Morais et al. (2014). The proposed AMP in this data set consistently improves the best solutions for 15 out of 16 instances. The average improvement on the best known solutions is −0.12%, ranging from +0.30 to −0.31%.

## 5.3 Small-scale many-to-many VRPCD instances

For the initial assessment of the proposed AMP algorithm we have generated 8 instances derived from the well-known data sets of Li and Lim (2001) originally introduced for the Pickup-and-Delivery Problem (PDP). The instances generated are from the *Random* class set of instances (lr100) involving cases where the suppliers and customers are randomly distributed. All of the instances involve 5 suppliers and 15 customers whose locations were randomly selected from the lr100 data set of Li and Lim (2001). Furthermore, in the first four instances (lr2_) customers request products from up to 2 suppliers, while in the last four instances (lr3_) customers request products from up to 3 suppliers. Note that for all instances we consider a common fixed service time for each supplier and customer location (as obtained from the original instances) as well as fixed times for the unloading and loading operations at the cross-dock. Table 3 reports the computational results produced by

**Table 1** Comparative results on the VRPCD problem instances

| Instance | W | | T | | M | N | | AMP | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $n \leq 200$ | *bst* | *t* | *bst* | *t* | *bst* | *bst* | *t* | *bst* | *avg* | *t* | *%avg* | *%bks* |
| 50a | 6471.90 | 3865 | **6450.28** | 28 | 6453.08 | 6450.28 | 68 | 6450.81 | 6483.67 | 48 | 0.51 | 0.01 |
| 50b | 7410.60 | 3185 | 7428.54 | 15 | 7434.90 | 7428.54 | 8 | **7406.16** | 7446.13 | 79 | 0.54 | −0.30 |
| 50c | 7330.60 | 3269 | **7311.77** | 13 | 7317.35 | 7311.77 | 39 | 7311.77 | 7360.66 | 68 | 0.67 | 0.00 |
| 50d | 7050.30 | 3658 | **7021.39** | 117 | 7035.50 | 7028.22 | 22 | 7028.66 | 7089.38 | 35 | 0.86 | 0.10 |
| 50e | 7516.80 | 3159 | **7451.42** | 20 | 7482.01 | 7451.42 | 32 | 7467.78 | 7548.63 | 26 | 1.08 | 0.22 |
| 100b | 14,526.10 | 9967 | 14,405.52 | 987 | 14,441.01 | **14,398.17** | 444 | 14,427.17 | 14,541.58 | 412 | 0.79 | 0.20 |
| 100c | 13,967.80 | 10,677 | 13,889.22 | 904 | 13,932.78 | **13,869.80** | 227 | 13,903.56 | 14,010.91 | 389 | 0.77 | 0.24 |
| 100d | 13,763.30 | 11,177 | **13,564.23** | 866 | 13,708.81 | 13,603.03 | 169 | 13,699.72 | 13,716.56 | 554 | 0.12 | 1.00 |
| 100e | 14,212.70 | 10,643 | **14,059.62** | 922 | 14,122.32 | 14,063.29 | 417 | 14,072.03 | 14,181.94 | 236 | 0.78 | 0.09 |
| 150a | 19,537.30 | 24,326 | 19,638.04 | 1302 | 19,532.28 | **19,391.16** | 350 | 19,528.58 | 19,662.09 | 865 | 0.68 | 0.71 |
| 150b | 20,974.80 | 24,461 | 20,922.27 | 1172 | 20,823.40 | **20,764.50** | 203 | 20,798.82 | 20,948.89 | 345 | 0.72 | 0.17 |
| 150c | 20,126.50 | 23,754 | 20,019.50 | 1004 | 19,964.59 | **19,864.86** | 326 | 19,922.34 | 20,059.50 | 567 | 0.69 | 0.29 |
| 150d | 20,549.40 | 24,468 | 20,600.33 | 673 | 20,509.97 | **20,355.27** | 366 | 20,381.98 | 20,693.25 | 744 | 1.53 | 0.13 |
| 150e | 19,848.50 | 23,400 | 19,782.00 | 877 | 19,716.87 | **19,634.47** | 269 | 19,686.25 | 19,838.45 | 644 | 0.77 | 0.26 |
| 200a | 27,324.40 | 46,586 | 27,397.31 | 1891 | 27,112.48 | **27,073.57** | 751 | 27,188.51 | 27,656.49 | 1043 | 1.72 | 0.42 |
| 200b | 27,637.70 | 43,653 | 27,582.87 | 1665 | 27,509.08 | **27,337.49** | 602 | 27,374.12 | 27,577.58 | 1446 | 0.74 | 0.13 |
| 200c | 26,358.60 | 46,389 | 26,425.29 | 1904 | 26,320.39 | **26,181.73** | 1024 | 26,300.51 | 26,642.51 | 1233 | 1.30 | 0.45 |
| 200d | 27,749.70 | 46,615 | 27,818.77 | 1789 | 27,686.75 | **27,439.50** | 1631 | 27,478.24 | 27,773.02 | 1080 | 1.07 | 0.14 |
| 200e | 26,620.60 | 45,649 | 26,704.81 | 1102 | 26,443.29 | **26,305.30** | 2033 | 26,388.89 | 26,572.83 | 993 | 0.70 | 0.32 |
| Average | | | | | | | | | | | | *0.24* |

*Methods* W, Wen et al. (2009), Dual Core AMD Opteron 175 2.2 GHz; T, Tarantilis (2013), Intel(R) Core(TM) 2 Extreme X7900 2.8 GHz; N, Nikolopoulou et al. (2017), Intel Xeon CPU E5-2650 2.60 GHz; and AMP—this paper

*Notation* $n$, total number of requests; *bst*, best solution over multiple runs (W 25 runs; T 3 runs; N 10 runs; AMP 20 runs); $t$, computational time (*Note* the average time required for generating the final solutions of the 25 runs is reported in W; All other methods report the total time for obtaining the best solution); *%avg*, the %deviation between the average and best solutions (=100($avg - bst$)/$bst$); and *%bks*, the %deviation between the best AMP solution and the best known solution (*BKS*) (=100*(AMP − *BKS*)/*BKS*)

**Table 2** Comparative results on the Morais set of instances

| Instance | M | | N | | AMP | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $n > 200$ | *bst* | *t* | *bst* | *t* | *bst* | *avg* | *t* | *%avg* | *%bks* |
| R1-4-1 | 15,445.28 | – | 15,208.84 | 1574 | **15,161.21** | 15,256.69 | 1983 | 0.63 | −0.31 |
| R1-4-2 | 14,850.75 | – | 14,614.02 | 1144 | **14,611.58** | 14,701.66 | 1941 | 0.62 | −0.02 |
| R1-4-3 | 14,332.27 | – | 14,101.73 | 973 | **14,079.12** | 14,183.27 | 1475 | 0.74 | −0.16 |
| R1-4-4 | 15,521.49 | – | 15,282.02 | 1009 | **15,275.40** | 15,353.66 | 827 | 0.51 | −0.04 |
| R1-6-1 | 33,511.04 | – | 32,696.90 | 4492 | **32,594.26** | 32,845.82 | 3091 | 0.77 | −0.31 |
| R1-6-2 | 33,540.56 | – | **32,623.17** | 3502 | 32,720.25 | 32,874.97 | 4632 | 0.47 | 0.30 |
| R1-6-3 | 33,282.54 | – | 32,624.50 | 2944 | **32,588.91** | 32,768.73 | 4009 | 0.55 | −0.11 |
| R1-6-4 | 33,468.72 | – | 32,748.70 | 2800 | **32,676.74** | 32,873.40 | 2167 | 0.60 | −0.22 |
| R1-8-1 | 60,300.22 | – | 58,961.82 | 3489 | **58,831.42** | 59,106.96 | 4868 | 0.47 | −0.22 |
| R1-8-2 | 58,113.85 | – | 56,894.76 | 6259 | **56,756.36** | 57,048.22 | 5708 | 0.51 | −0.24 |
| R1-8-3 | 58,558.94 | – | 57,124.27 | 5164 | **57,068.45** | 57,369.43 | 9177 | 0.53 | −0.10 |
| R1-8-4 | 60,502.26 | – | 59,169.87 | 4175 | **59,071.09** | 59,355.56 | 9556 | 0.48 | −0.17 |
| R1-10-1 | 94,080.68 | – | **91,657.18** | 5211 | 91,679.83 | 92,088.68 | 7893 | 0.45 | 0.02 |
| R1-10-2 | 92,792.34 | – | 91,001.23 | 6670 | **90,969.12** | 91,318.99 | 8941 | 0.38 | −0.04 |
| R1-10-3 | 93,222.85 | – | 91,016.40 | 6166 | **90,887.80** | 91,281.18 | 11,362 | 0.43 | −0.14 |
| R1-10-4 | 94,372.82 | – | 92,112.35 | 5662 | **91,943.18** | 92,051.81 | 6936 | 0.12 | −0.18 |
| Average | | | | | | | | | *−0.12* |

*Methods* M, Morais et al. (2014), Intel Core$^{TM}$ 2 Duo E8400 3.0 GHz; N, Nikolopoulou et al. (2017), Intel Xeon CPU E5-2650 2.60 GHz; and AMP—this paper

*Notation* $n$, total number of requests; *bst*, best solution over multiple runs (40 runs for M; 10 runs for N; 20 runs for AMP); $t$, computational time for obtaining the best solution; *%avg*, %deviation between the average and best solutions (=100($avg - bst$)/$bst$); and *%bks*, %deviation between the best AMP solution and the best known solution ($BKS$) (=100*(AMP − $BKS$)/$BKS$)

**Table 3** Computational experiments on small-scale many-to-many VRPCD problem instances

| Instance | AMP | | CPLEX | | | | |
|---|---|---|---|---|---|---|---|
| | *bst* | *AMP_t* | *RootLB* | *#Nodes* | *CPLEX_t* | *LB* | *%gap* |
| lr2_1 | 445.50 | 6 | 378.28 | 1,548,119 | 1216 | 445.50* | – |
| lr2_2 | 504.53 | 5 | 351.67 | 6,776,700 | 2438 | 504.53* | – |
| lr2_3 | 504.44 | 5 | 388.92 | 2,169,564 | 1069 | 504.44* | – |
| lr2_4 | 406.42 | 6 | 311.54 | 3,175,782 | 1175 | 406.42* | – |
| lr3_1 | 580.53 | 9 | 493.04 | 3,779,824 | 1925 | 580.53* | – |
| lr3_2 | 520.64 | 9 | 340.55 | 3,584,689 | 21,600 | 502.80 | 3.43 |
| lr3_3 | 633.09 | 8 | 460.49 | 13,119,000 | 21,600 | 617.34 | 2.54 |
| lr3_4 | 653.70 | 9 | 548.36 | 9,719,103 | 4116 | 653.70* | – |

*bst*, the best solution found from the AMP; *AMP_t*, the average time required by the AMP for generating the best solutions (s); *Root LB*, lower bound at the root node; #*Nodes*, number of CPLEX nodes explored; *CPLEX_t*, total time required by CPLEX (s) to close the gap; *LB*, Lower bound. The * sign is added when CPLEX closed the gap with respect to the best found solution obtained by the AMP with a 0.01% optimality tolerance, *%gap*: Gap between the best AMP solution and the *LB* obtained by CPLEX (=100/(*bst* − *LB*)/*bst*)

the AMP metaheuristic algorithm compared to the optimal solutions or lower bounds obtained from CPLEX.

The AMP algorithm exhibited a very stable performance. For all 8 problems, all ten runs obtained the same final solution (*bst*). In terms of the computational time, the average time required by the AMP for reaching the final solution of each of the ten runs ranged from 5.5 up to 8.75 CPU seconds for the lr2_ and lr_3 instance series, respectively. Concerning the MILP CPLEX runs, the smaller problems appeared to be easier to solve: for all 4 instances of the lr2_ instance series, CPLEX managed to close the gap and find the best solution obtained initially by the AMP (optimality tolerance was set equal to 0.01%), whereas for the lr_3 instance series, CPLEX found the best solution obtained initially by the AMP for 2 out of 4 instances.

## 5.4 Medium- and large-scale many-to-many VRPCD instances

### 5.4.1 Data set generation method

A new data set of benchmark problem instances has been generated. Similarly to the set of instances described in Sect. 5.3, these problem instances are also derived from the data sets of Li and Lim (2001). Three main classes of instances are developed, namely *Random, Random-Clustered* and *Clustered* referring to the geographic distribution of supplier and customer nodes. The *Random* class involves cases where the suppliers and customers are randomly located in the same geographic region, whereas the *Clustered* class involves cases where suppliers and customers are located in different areas (clusters). In the *Random-Clustered* case only some suppliers and customers are located in the same area. The *Random* and *Random-Clustered* instances are produced by modifying the following sets: lr100; lrc100;

LR1_2, and LRC1_2, respectively. The *Clustered* instances are produced by modifying the following sets: lc100 and LC1_2. Initially, collocated nodes are removed from the graphs. The cross-dock is set to be the original depot node. Then, the sets of suppliers and customers are generated from the nodes of the original instance. Let $n_{org}$ be the number of pickup and delivery nodes of the original nodes after the collocated points have been removed. Obviously, $|S'| + |D'| = n_{org}$. Three ratios of supplier participation $\sigma = |S'|/n_{org}$ are considered: 10, 30 and 40%. For the *Random* and *Random-Clustered* instances, the number of suppliers $|S'|$ is randomly picked out of the $n_{org}$ vertices. On the contrary, for the *Clustered* instances, the number of suppliers $|S'|$ is manually picked to be concentrated into different regions compared to the rest of the node population. The remaining $n_{org} - |S'|$ nodes are included in the set of customers.

To quantify the distribution of the node locations for the generated sets, the Nearest Neighbor Index (*NNI*) is used (Clark and Evans 1954). The *NNI* is defined as the ratio $V_{obs}/V_{ran}$, where the observed distance $V_{obs}$ represents the average over all distances between each point and its nearest neighbor, and the random distance $V_{ran}$ is the expected average distance that would occur, if the distribution was random. The former is given by $V_{obs} = (\sum_{i \in S \cup D} min_{j \in S \cup D, i \neq j}(d_{ij}))/n_{org}$. The latter can obtained as $V_{ran} = 0.5\sqrt{Y/n_{org}}$, where $Y$ is the area of the grid. *NNI* values <1, indicate some degree of clustering. In Table 4, we provide the *NNI* values for the three geographic distribution classes.

After separating the set of nodes into customers and suppliers, the transportation request flows are identified. At first, the demand of each customer $c$ is set equal to $d_c$, where $d_c$ is the absolute value of the original demand of node $c$ (note that in the original data set a node could be either a delivery or a pickup node). Next, we define the maximum number of suppliers serving each customer node $c$. For this purpose, we have used four different "supplier-customer connection density" classes $\varepsilon$ with 1, 2, 3, and 4 number of connections, respectively. To that end, the total $d_c$ quantity is randomly allocated to the $\varepsilon$ supplier nodes. Special care is given so that each supplier is associated with at least one customer node. From the demand matrix perspective presented in Sect. 2, note that $\sum_{i \in S} d_{ij} = d_c, \forall j \in D$ and $\sum_{j \in D} d_{ij} > 0, \forall i \in S$.

In terms of the capacities of the inbound and outbound vehicles, we used $Q_S = 5\sum_{c \in D} d_c/|D|$ and $Q_D = 0.5Q_S$ rounded down to the nearest ten. If these values cannot guarantee feasibility, they are set to the minimum capacity levels adequate for serving every supplier or customer node. Regarding the time characteristics of the problem, for all pickup/delivery nodes the service time $s_t$ is common and is equal to the original value used by Li and Lim (2001). In terms of the loading and

**Table 4** Nearest neighbor index (*NNI*) for the three geographic distribution classes

| Instance size ($n_{org}$) | R | RC | C |
|---|---|---|---|
| 100 | 1.50 | 1.04 | 0.61 |
| 200 | 1.05 | 0.97 | 0.51 |

unloading operations taking place at the cross-dock, we set $u = l = 2s_t$. For each problem instance and $\sigma$-class combination, all four $\varepsilon$-class instances are associated with a common maximum route duration ($T$) value. This $T$ value is heuristically obtained so as to ensure both a challenging interplay with the capacity constraints as well as the feasibility of the instances. In total, 708 instances are generated (59 original PDP problem instances x 3 $\sigma$ classes x 4 $\varepsilon$ classes).

### *5.4.2 Assessment of the proposed method on the new benchmark instances*

Table 5 summarizes the average results obtained for the many-to-many VRPCD on the new benchmark data sets (detailed solutions for all problem instances can be found in "Appendix"). The problem instances are initially divided in two sections referring to instances with 100 and 200 nodes. Furthermore, the problem instances are grouped according to their characteristics. In particular, each result represents the average solution costs for all problem instances of the same geographic distribution ($R$, $RC$, $C$) class, supplier-customer density ($\varepsilon$) class and supplier participation ratio ($\sigma$).

One main observation is that the transportation costs increase for all cases as the supplier participation ratio increases from 10 to 40%. This is not surprising as the problem instances with higher supplier participation ratio, involve vehicles with low capacities, and therefore, more pickup and delivery routes are required to serve all requests. Table 5 also reveals reduced costs for the clustered distribution class ($C$) for all instances with 100 and 200 nodes, compared to the random classes ($R$ and $RC$). We observe that the average costs decrease as the geographic distribution of the nodes becomes more clustered, for the cases where the supplier participation ratio is 10 and 30%. This indicates that the cross-docking operations might be more beneficial when the customers are clustered. However, this is not the case for instances where the supplier participation is high (40%). This is possibly due to the fact that the original Li and Lim (2001) instances of the $C$ class have larger $T$ values, and larger cross-dock preparation times compared to the instances of the $R$ and $RC$ classes. Therefore, when the supplier participation increases, vehicles perform routes to more isolated/distant nodes, and the loading and unloading operations taking place at the cross-dock are more time-consuming.

Considering the implications on transportation costs with respect to the supplier-customer density class, a rather unexpected result occurs. We observe reduced transportation costs when customers are associated with more than one suppliers, compared to cases where there is a one-to-one relationship between suppliers and customers. More specifically, for all the groups of problems where the supplier participation ratio is low (10%), transportation costs decrease for the more dense groups ($\varepsilon = 4$), compared to the problem groups with $\varepsilon = 1$. The same finding is also observed for the $R$ and $RC$ classes of all instances with 30% supplier participation ratio. This shows that cost savings can be achieved when customers are able to receive the total delivery quantity by multiple suppliers, not just one. Similar are the findings for the average solutions obtained for problem instances with 200 nodes among the $R$, $RC$ and $C$ classes.

**Table 5** Average solution costs obtained for the new benchmark data sets

| *Geographic distribution* | R | | | RC | | | C | | |
|---|---|---|---|---|---|---|---|---|---|
| $\varepsilon/\sigma$ | 10% | 30% | 40% | 10% | 30% | 40% | 10% | 30% | 40% |
| $n_{\text{org}} = 100$ | | | | | | | | | |
| 1 | 999.26 | 1603.79 | 1983.35 | 1010.15 | 1897.45 | 2461.54 | 782.40 | 1577.58 | 2032.24 |
| 2 | 999.05 | 1626.25 | 2007.53 | 1010.28 | 1913.52 | 2451.30 | 787.58 | 1589.41 | 2030.04 |
| 3 | 998.95 | 1606.04 | 2008.31 | 1010.68 | 1925.23 | 2446.68 | 786.07 | 1593.29 | 2051.68 |
| 4 | 999.05 | 1609.59 | 2005.93 | 1010.15 | 1958.15 | 2460.32 | 782.26 | 1590.31 | 2052.22 |
| $n_{\text{org}} = 200$ | | | | | | | | | |
| 1 | 2817.26 | 5692.13 | 6973.20 | 2640.02 | 5514.90 | 6904.23 | 2394.73 | 5342.60 | 7008.08 |
| 2 | 2803.50 | 5682.68 | 7054.67 | 2633.88 | 5515.28 | 6863.73 | 2398.26 | 5399.25 | 7139.56 |
| 3 | 2801.83 | 5669.92 | 7071.24 | 2640.79 | 5519.49 | 6877.18 | 2415.05 | 5400.93 | 7137.10 |
| 4 | 2800.70 | 5682.24 | 7033.00 | 2639.02 | 5502.95 | 6876.78 | 2399.31 | 5383.61 | 7155.74 |

### *5.4.3 Computational experiments with collocated nodes and split options*

In this section, the benefit of splitting requests by alleviating the restriction that each node should be visited only once is examined. For this purpose, we transform the many-to-many data sets presented in Sect. 5.4.1 so as to consider a one-to-one mapping between suppliers and customers (as such different requests may involve collocated suppliers and customers).

To make this transformation, we generate for each supplier node $i$ that is connected with more than one customers, collocated supplier nodes that are copies of the original $i$ node. Obviously, the total number of collocated suppliers generated for supplier $i$ is equal to the total number of non-zero entries in row $i$ of the demand matrix. Similarly, the total number of collocated customers generated for customer $j$ of the original problem instance, is equal to the total number of non-zero entries in column $j$ of the demand matrix. To generate the service times for each node of the modified instance, we divide the original service time value $s_t$ by the number of node copies generated. In what follows, we denote as Data Set I the transformed data set with originally 100 nodes and as Data Set II the transformed data set with originally 200 nodes.

Table 6 summarizes the average results obtained for all groups of problems on the transformed data set. Detailed solutions for all problem instances can be found in "Appendix". Similar to Table 5, the problem instances are initially divided into two sections with respect to the problem size, i.e., 100 and 200 nodes. Note that for all problem instances of the same geographic distribution ($R$, $RC$ or $C$) class, supplier-customer density ($\varepsilon$) class and supplier participation ratio ($\sigma$), Table 6 presents the average solution costs for each group of problems as well as the average number of split requests #$s$, including both pickups and deliveries. Let $k_{\mathrm{i}}$ be the total number of vehicle routes serving the node copies of supplier $i$. Then, the number of split pickup requests for supplier $i$ is defined as $s_i = \left(\sum_{i \in S} k_i\right) - 1$. Similarly, let $k_{\mathrm{j}}$ be the total number of vehicle routes serving the node copies of customer $j$. Then, the number of split delivery requests for customer $j$ is defined as $s_j = \left(\sum_{j \in D} k_j\right) - 1$.

Compared to Table 5, we can clearly observe that better on average costs can be obtained when splitting of the transportation requests is allowed. This is not surprising, as splitting yields more opportunities for cost improvement through combined pickup and/or delivery routes. Another observation from Table 6 is that for the cases where the supplier participation ratio is low (10%), splitting requests does not improve the overall costs. However, this is not the case for the clustered instances ($C$ class). As shown in Table 6, allowing requests that belong to $C$ class to be split, results in reduced average transportation costs for all classes of the supplier participation ratio. Lastly, Table 6 reveals that the average number of split requests is higher for the random ($R$) distribution class, and in particular for the larger supplier-customer connection density instance ($\varepsilon$) classes. Allowing requests to be split in this case, may yield substantial savings. Note that the total service time of each pickup (or delivery) node in the transformed data set is divided by the number of customers (or suppliers) connected to this node so that the comparison between

**Table 6** Average solution costs obtained for the transformed data set (data set I and II) with collocated nodes and split options

| ε/σ | R | | | | | | RC | | | | | | C | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 10% | #*s* | 30% | #*s* | 40% | #*s* | 10% | #*s* | 30% | #*s* | 40% | #*s* | 10% | #*s* | 30% | #*s* | 40% | #*s* |
| 100 | Data set I | | | | | | | | | | | | | | | | | |
| 1 | 999.26 | – | 1603.79 | – | 1983.35 | – | 1010.15 | – | 1897.45 | – | 2461.54 | – | 782.40 | – | 1577.58 | – | 2032.24 | – |
| 2 | 999.05 | 0.00 | 1602.02 | 8.50 | 1979.84 | 11.50 | 1010.28 | 0.00 | 1910.45 | 3.75 | 2437.34 | 7.25 | 781.10 | 0.67 | 1584.54 | 4.22 | 2014.16 | 6.33 |
| 3 | 998.95 | 0.00 | 1597.44 | 7.67 | 1978.63 | 12.42 | 1010.68 | 0.00 | 1924.79 | 0.50 | 2429.45 | 9.88 | 779.11 | 0.89 | 1578.78 | 4.44 | 2036.72 | 7.78 |
| 4 | 999.05 | 0.00 | 1604.73 | 3.92 | 1984.8 | 13.67 | 1010.15 | 0.00 | 1957.48 | 0.88 | 2440.90 | 17.25 | 779.56 | 1.00 | 1547.50 | 9.89 | 2048.59 | 1.78 |
| 200 | Data set II | | | | | | | | | | | | | | | | | |
| 1 | 2817.26 | – | 5692.13 | – | 6964.93 | – | 2640.02 | – | 5514.90 | – | 6895.21 | – | 2392.30 | – | 5322.65 | – | 7005.46 | – |
| 2 | 2803.42 | 0.10 | 5681.70 | 3.20 | 7005.49 | 21.50 | 2633.88 | 0.00 | 5508.57 | 1.50 | 6841.16 | 7.10 | 2369.89 | 2.30 | 5372.73 | 2.10 | 7122.00 | 5.20 |
| 3 | 2801.83 | 0.00 | 5662.70 | 9.00 | 7028.08 | 29.90 | 2640.79 | 0.00 | 5510.88 | 3.30 | 6857.47 | 2.80 | 2377.52 | 3.10 | 5384.83 | 2.70 | 7135.98 | 0.00 |
| 4 | 2800.70 | 0.00 | 5682.24 | 0.00 | 7018.27 | 10.40 | 2639.02 | 0.00 | 5502.95 | 0.00 | 6864.16 | 3.50 | 2371.07 | 1.90 | 5378.66 | 0.00 | 7146.89 | 0.00 |

the two data sets is fair. However, in practice we often need to add extra setup time the service at each node, and this is the trade-off we need to consider.

## 6 Conclusions

This paper examined a new vehicle routing problem with pickups, deliveries and a cross-dock facility. Key feature of the examined problem was that each customer might request products from multiple suppliers, while there is the restriction to visit each customer and supplier node only once by exactly one vehicle. To our knowledge this many-to-many relationship for the VRPCD appears for the first time in the literature, and it can be seen as a generalization of one-to-one settings with and without collocated nodes. A mathematical formulation was introduced that captures all the critical characteristics of the problem considered. For solving the problem, an Adaptive Memory Programming method was proposed. Key methodological component was the procedure followed for identifying and selecting elite subroutes of varying length from the reference solutions.

Various computational experiments were conducted to assess the performance of the proposed method and to examine the characteristics of the new problem. On existing benchmark data sets for the one-to-one VRPCD, the AMP performed very well compared to the current state-of-the-art approaches of the literature. On the other hand, computational results on new data sets provided new useful insights. For example, the solution costs increased as the supplier participation ratio increased. The solution costs decreased as the geographic distribution became more clustered for all cases with supplier participation ratio <30%. Significant savings may also be observed if splitting is allowed for the *R* and *RC* classes combined with cases where the supplier participation ratio is >30%. Finally, more splits are observed for the cases where supplier and customers are randomly distributed in a geographic region with large supplier-customer connection density.

**Acknowledgements** This work was supported by the IKY fellowships of excellence for postgraduate studies in Greece—Siemens program, as well as by the Research Center of the Athens University of Economics and Business (EP-2349-01). Support from the National Science Foundation under Award Number 1434432 is also gratefully acknowledged.

## Appendix

Detailed solutions for all problem instances of the new benchmark data sets are reported in Tables 7, 8, 9 and 10. In particular, Tables 7 and 8 present the solutions obtained for the many-to-many VRPCD with no split options with up to 100 and 200 nodes, respectively. Note that the average results for these problem instances are reported in Table 4. Tables 9 and 10 present the solution obtained for the transformed VRPCD with collocated nodes and split options, with up to 100 and 200 nodes, respectively. In both cases the problem instances are grouped according to the geographic distribution class, supplier-customer density ($\varepsilon$) class, and supplier participation ratio ($\sigma$).

**Table 7** Detailed solution scores for the many-to-many VRPCD data set

| $\sigma$ | 10% | | | | 30% | | | | 40% | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance/ε | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| lr | | | | | | | | | | | | |
| lr101 | 1000.36 | 1000.36 | 1000.36 | 1000.36 | 1629.49 | 1617.71 | 1598.5 | 1610.82 | 2068.03 | 2088.92 | 2094.05 | 2061.91 |
| lr102 | 1000.14 | 1000.14 | 1000.14 | 1000.14 | 1580.55 | 1589.37 | 1615.97 | 1598.81 | 1927.04 | 1886.74 | 1906.74 | 1900.12 |
| lr103 | 1001.30 | 1000.36 | 1000.36 | 1000.36 | 1634.00 | 1658.42 | 1631.56 | 1619.89 | 2009.53 | 2025.97 | 1996.48 | 2068.55 |
| lr104 | 991.34 | 991.34 | 991.34 | 991.34 | 1590.89 | 1597.89 | 1577.02 | 1617.27 | 2064.53 | 2034.39 | 2059.61 | 2022.80 |
| lr105 | 1002.89 | 1002.89 | 1002.89 | 1002.89 | 1612.68 | 1592.93 | 1600.19 | 1596.47 | 1966.87 | 1999.10 | 1977.73 | 1971.33 |
| lr106 | 1002.96 | 1002.96 | 1002.96 | 1002.96 | 1543.98 | 1541.88 | 1555.57 | 1546.85 | 1968.20 | 1963.09 | 1974.97 | 1963.12 |
| lr107 | 994.15 | 994.15 | 994.15 | 994.15 | 1536.13 | 1550.50 | 1561.18 | 1576.46 | 1965.24 | 1922.12 | 1951.35 | 1951.96 |
| lr108 | 994.15 | 994.15 | 994.15 | 994.15 | 1606.34 | 1622.95 | 1591.58 | 1622.33 | 1890.41 | 1917.91 | 1886.37 | 1910.00 |
| lr109 | 1000.36 | 1000.36 | 1000.36 | 1000.36 | 1676.86 | 1646.53 | 1648.61 | 1647.70 | 2003.40 | 1989.68 | 1975.82 | 2022.61 |
| lr110 | 1006.18 | 1006.18 | 1006.18 | 1006.18 | 1553.48 | 1566.57 | 1562.55 | 1581.93 | 1931.82 | 1947.15 | 1943.72 | 1957.36 |
| lr111 | 994.15 | 994.15 | 994.15 | 994.15 | 1673.41 | 1667.81 | 1658.59 | 1658.49 | 2078.25 | 2042.26 | 2023.99 | 2049.29 |
| lr112 | 1003.16 | 1001.64 | 1000.36 | 1001.64 | 1607.70 | 1571.70 | 1567.93 | 1579.68 | 1926.91 | 1940.76 | 1952.65 | 1938.60 |
| lc | | | | | | | | | | | | |
| lc101 | 807.71 | 804.66 | 802.79 | 802.79 | 1592.48 | 1580.17 | 1610.29 | 1543.76 | 2116.08 | 2086.22 | 2149.85 | 2161.91 |
| lc102 | 804.27 | 800.69 | 801.12 | 801.12 | 1600.75 | 1608.15 | 1604.99 | 1583.01 | 1936.19 | 1940.51 | 1943.78 | 1962.68 |
| lc103 | 805.41 | 802.82 | 802.67 | 802.68 | 1625.22 | 1634.07 | 1651.32 | 1595.56 | 2113.21 | 2095.39 | 2115.25 | 2099.79 |
| lc104 | 804.82 | 802.82 | 802.67 | 802.62 | 1550.72 | 1558.55 | 1528.66 | 1515.52 | 2018.67 | 1992.02 | 2018.73 | 1998.47 |
| lc105 | 806.15 | 804.66 | 802.79 | 802.86 | 1565.46 | 1554.95 | 1555.37 | 1514.62 | 1946.66 | 1959.47 | 1962.75 | 1974.67 |
| lc106 | 724.29 | 723.78 | 719.46 | 720.96 | 1519.68 | 1548.41 | 1548.86 | 1530.58 | 2121.51 | 2110.11 | 2122.29 | 2151.89 |
| lc107 | 763.50 | 767.71 | 759.94 | 761.82 | 1614.94 | 1628.71 | 1628.44 | 1606.14 | 1994.93 | 1973.85 | 1997.07 | 2029.57 |
| lc108 | 804.27 | 800.69 | 801.12 | 801.12 | 1562.36 | 1574.53 | 1523.89 | 1508.41 | 2069.93 | 2038.03 | 2074.32 | 2101.60 |
| lc109 | 721.13 | 722.08 | 719.46 | 720.07 | 1566.65 | 1573.33 | 1557.22 | 1529.92 | 1973.00 | 1931.84 | 1946.47 | 1956.72 |

**Table 7** continued

| σ | 10% | | | | 30% | | | | 40% | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance/ε | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| lrc | | | | | | | | | | | | |
| lrc101 | 1008.73 | 1008.73 | 1008.73 | 1008.73 | 1989.40 | 1983.89 | 2061.30 | 2069.71 | 2499.44 | 2481.41 | 2481.09 | 2475.91 |
| lrc102 | 1002.42 | 1002.42 | 1002.42 | 1002.42 | 1826.15 | 1861.89 | 1841.01 | 1906.66 | 2471.91 | 2472.86 | 2459.51 | 2473.06 |
| lrc103 | 1007.95 | 1007.95 | 1011.6 | 1007.95 | 1883.99 | 1878.24 | 1900.43 | 1943.96 | 2415.88 | 2427.55 | 2371.84 | 2420.01 |
| lrc104 | 1007.38 | 1007.38 | 1007.38 | 1007.38 | 1862.02 | 1859.70 | 1861.93 | 1894.14 | 2395.26 | 2345.99 | 2410.29 | 2372.80 |
| lrc105 | 1016.38 | 1016.38 | 1016.38 | 1016.38 | 1932.55 | 1938.78 | 1942.64 | 1973.02 | 2392.51 | 2392.03 | 2379.21 | 2345.97 |
| lrc106 | 1014.55 | 1014.55 | 1014.55 | 1014.55 | 1901.24 | 1904.35 | 1933.47 | 1962.95 | 2392.79 | 2328.66 | 2333.88 | 2398.99 |
| lrc107 | 1012.72 | 1012.72 | 1013.33 | 1012.72 | 1921.26 | 1970.79 | 1965.62 | 2011.12 | 2601.34 | 2538.88 | 2530.89 | 2556.10 |
| lrc108 | 1011.03 | 1012.11 | 1011.03 | 1011.03 | 1862.98 | 1885.99 | 1891.92 | 1898.30 | 2523.17 | 2511.32 | 2468.85 | 2484.33 |

**Table 8** Detailed solutions for the many-to-many VRPCD data set

| $\sigma$ | 10% | | | | 30% | | | | 40% | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance/ε | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| LR | | | | | | | | | | | | |
| LR1_2_1 | 2773.97 | 2803.45 | 2856.76 | 2827.79 | 5697.90 | 5660.80 | 5623.51 | 5658.57 | 7252.00 | 7277.62 | 7328.01 | 7353.93 |
| LR1_2_2 | 2782.07 | 2835.49 | 2793.11 | 2789.61 | 5553.50 | 5533.49 | 5508.75 | 5526.32 | 7022.90 | 7136.52 | 7113.62 | 7088.01 |
| LR1_2_3 | 2865.31 | 2889.63 | 2808.82 | 2816.28 | 5835.19 | 5798.40 | 5820.06 | 5823.12 | 6888.66 | 6935.09 | 6987.29 | 6941.87 |
| LR1_2_4 | 2850.38 | 2743.80 | 2794.83 | 2795.61 | 5629.74 | 5620.23 | 5614.20 | 5618.24 | 6954.23 | 6974.97 | 6991.49 | 7013.56 |
| LR1_2_5 | 2791.65 | 2863.57 | 2777.49 | 2778.59 | 5559.51 | 5660.84 | 5560.41 | 5556.38 | 6783.37 | 6826.06 | 6846.15 | 6867.30 |
| LR1_2_6 | 2801.26 | 2831.61 | 2782.99 | 2784.96 | 5781.66 | 5827.82 | 5808.10 | 5839.04 | 6864.68 | 6891.40 | 6915.88 | 6930.72 |
| LR1_2_7 | 2806.14 | 2734.29 | 2811.87 | 2825.59 | 5596.45 | 5574.75 | 5560.84 | 5599.97 | 6921.38 | 6902.21 | 6976.76 | 6898.74 |
| LR1_2_8 | 2830.45 | 2791.49 | 2780.52 | 2822.44 | 5735.52 | 5709.92 | 5701.08 | 5739.75 | 7103.08 | 7082.39 | 7076.27 | 7102.47 |
| LR1_2_9 | 2881.92 | 2738.87 | 2833.87 | 2787.06 | 5673.88 | 5642.80 | 5654.44 | 5674.85 | 6879.25 | 7013.27 | 7028.66 | 6967.95 |
| LR1_2_10 | 2789.49 | 2802.01 | 2778.02 | 2779.08 | 5857.95 | 5787.98 | 5775.58 | 5786.13 | 6979.77 | 7015.37 | 7016.67 | 7018.14 |
| LC | | | | | | | | | | | | |
| LC1_2_1 | 2329.48 | 2306.24 | 2314.15 | 2310.74 | 5232.88 | 5253.25 | 5295.87 | 5289.13 | 7148.52 | 7240.07 | 7281.98 | 7327.87 |
| LC1_2_2 | 2415.45 | 2335.6 | 2349.21 | 2339.55 | 5280.75 | 5301.26 | 5303.85 | 5292.28 | 7291.12 | 7470.23 | 7478.52 | 7476.74 |
| LC1_2_3 | 2401.37 | 2405.44 | 2426.52 | 2400.68 | 5253.02 | 5365.01 | 5408.38 | 5365.70 | 6789.89 | 6865.97 | 6918.31 | 6914.91 |
| LC1_2_4 | 2407.32 | 2413.94 | 2422.54 | 2405.96 | 5459.50 | 5559.86 | 5566.34 | 5553.28 | 6892.99 | 7074.70 | 7031.09 | 7054.35 |
| LC1_2_5 | 2395.98 | 2400.53 | 2423.59 | 2401.06 | 5214.71 | 5270.79 | 5251.24 | 5243.55 | 7356.01 | 7392.62 | 7430.01 | 7457.24 |
| LC1_2_6 | 2417.07 | 2410.61 | 2427.57 | 2408.77 | 5527.85 | 5633.32 | 5664.53 | 5666.40 | 7029.16 | 7148.57 | 7172.69 | 7182.25 |
| LC1_2_7 | 2400.86 | 2339.49 | 2323.57 | 2353.72 | 5222.21 | 5268.77 | 5255.19 | 5288.83 | 7086.97 | 7200.59 | 7237.34 | 7238.18 |
| LC1_2_8 | 2402.54 | 2408.59 | 2419.77 | 2412.47 | 5417.41 | 5444.16 | 5515.50 | 5506.80 | 6913.33 | 7058.66 | 7090.68 | 7087.27 |
| LC1_2_9 | 2349.37 | 2281.76 | 2274.30 | 2280.15 | 5310.33 | 5291.03 | 5277.41 | 5285.37 | 6805.90 | 6959.56 | 6918.42 | 6928.78 |
| LC1_2_10 | 2403.54 | 2403.37 | 2425.28 | 2422.52 | 5307.83 | 5339.86 | 5313.12 | 5295.32 | 6740.70 | 6809.06 | 6800.72 | 6801.31 |

**Table 8** continued

| $\sigma$ | 10% | | | | 30% | | | | 40% | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance/ε | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| LRC | | | | | | | | | | | | |
| LRC1_2_1 | 2627.72 | 2645.88 | 2658.29 | 2654.86 | 5523.30 | 5523.20 | 5524.26 | 5512.14 | 7201.85 | 7152.59 | 7137.75 | 7216.56 |
| LRC1_2_2 | 2632.54 | 2627.70 | 2656.47 | 2624.08 | 5537.32 | 5523.88 | 5613.31 | 5558.96 | 6761.08 | 6687.38 | 6764.65 | 6708.51 |
| LRC1_2_3 | 2630.26 | 2610.88 | 2609.63 | 2610.88 | 5436.61 | 5434.22 | 5423.21 | 5374.95 | 6900.20 | 6846.2 | 6874.49 | 6818.94 |
| LRC1_2_4 | 2652.76 | 2639.24 | 2639.24 | 2634.14 | 5469.09 | 5485.85 | 5474.29 | 5492.36 | 7057.22 | 6962.15 | 7043.01 | 7027.84 |
| LRC1_2_5 | 2637.13 | 2629.33 | 2614.44 | 2644.09 | 5544.30 | 5563.57 | 5533.66 | 5541.87 | 7201.95 | 7145.12 | 7136.57 | 7185.97 |
| LRC1_2_6 | 2646.03 | 2640.87 | 2634.28 | 2624.11 | 5676.42 | 5654.44 | 5638.50 | 5632.26 | 6890.04 | 6906.95 | 6829.99 | 6829.53 |
| LRC1_2_7 | 2658.93 | 2657.59 | 2669.32 | 2707.27 | 5486.07 | 5490.38 | 5488.62 | 5477.60 | 6984.99 | 6899.35 | 6959.19 | 6981.96 |
| LRC1_2_8 | 2644.32 | 2636.30 | 2649.20 | 2636.30 | 5446.05 | 5428.69 | 5448.99 | 5452.41 | 6873.16 | 6858.18 | 6834.44 | 6902.70 |
| LRC1_2_9 | 2647.96 | 2644.04 | 2652.63 | 2650.81 | 5530.64 | 5477.44 | 5515.56 | 5494.81 | 6537.70 | 6481.12 | 6506.87 | 6481.45 |
| LRC1_2_10 | 2622.50 | 2606.97 | 2624.39 | 2603.65 | 5499.23 | 5504.05 | 5448.44 | 5492.19 | 6543.87 | 6472.55 | 6487.78 | 6513.56 |

Table 9 Detailed solutions for the transformed VRPCD data set with collocated nodes and split options (Data Set I)

| σ | 10% | | | | 30% | | | | 40% | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance/ε | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| lr | | | | | | | | | | | | |
| lr101 | 1000.36 | 1000.36 | 1000.36 | 1000.36 | 1629.49 | 1617.71 | 1598.5 | 1610.82 | 2068.03 | 2088.92 | 2094.05 | 2061.91 |
| lr102 | 1000.14 | 1000.14 | 1000.14 | 1000.14 | 1580.55 | 1589.37 | 1615.97 | 1598.81 | 1927.04 | 1886.74 | 1906.74 | 1900.12 |
| lr103 | 1001.30 | 1000.36 | 1000.36 | 1000.36 | 1634.00 | 1658.42 | 1631.56 | 1619.89 | 2009.53 | 2025.97 | 1996.48 | 2068.55 |
| lr104 | 991.34 | 991.34 | 991.34 | 991.34 | 1590.89 | 1597.89 | 1577.02 | 1617.27 | 2064.53 | 2034.39 | 2059.61 | 2022.8 |
| lr105 | 1002.89 | 1002.89 | 1002.89 | 1002.89 | 1612.68 | 1592.93 | 1600.19 | 1596.47 | 1966.87 | 1999.1 | 1977.73 | 1971.33 |
| lr106 | 1002.96 | 1002.96 | 1002.96 | 1002.96 | 1543.98 | 1541.88 | 1555.57 | 1546.85 | 1968.20 | 1963.09 | 1974.97 | 1963.12 |
| lr107 | 994.15 | 994.15 | 994.15 | 994.15 | 1536.13 | 1550.50 | 1561.18 | 1576.46 | 1965.24 | 1922.12 | 1951.35 | 1951.96 |
| lr108 | 994.15 | 994.15 | 994.15 | 994.15 | 1606.34 | 1622.95 | 1591.58 | 1622.33 | 1890.41 | 1917.91 | 1886.37 | 1910.00 |
| lr109 | 1000.36 | 1000.36 | 1000.36 | 1000.36 | 1676.86 | 1646.53 | 1648.61 | 1647.7 | 2003.40 | 1989.68 | 1975.82 | 2022.61 |
| lr110 | 1006.18 | 1006.18 | 1006.18 | 1006.18 | 1553.48 | 1566.57 | 1562.55 | 1581.93 | 1931.82 | 1947.15 | 1943.72 | 1957.36 |
| lr111 | 994.15 | 994.15 | 994.15 | 994.15 | 1673.41 | 1667.81 | 1658.59 | 1658.49 | 2078.25 | 2042.26 | 2023.99 | 2049.29 |
| lr112 | 1003.16 | 1001.64 | 1000.36 | 1001.64 | 1607.70 | 1571.7 | 1567.93 | 1579.68 | 1926.91 | 1940.76 | 1952.65 | 1938.6 |
| lc | | | | | | | | | | | | |
| lc101 | 807.71 | 804.66 | 802.79 | 802.79 | 1592.48 | 1580.17 | 1610.29 | 1543.76 | 2116.08 | 2086.22 | 2149.85 | 2161.91 |
| lc102 | 804.27 | 800.69 | 801.12 | 801.12 | 1600.75 | 1608.15 | 1604.99 | 1583.01 | 1936.19 | 1940.51 | 1943.78 | 1962.68 |
| lc103 | 805.41 | 802.82 | 802.67 | 802.68 | 1625.22 | 1634.07 | 1651.32 | 1595.56 | 2113.21 | 2095.39 | 2115.25 | 2099.79 |
| lc104 | 804.82 | 802.82 | 802.67 | 802.62 | 1550.72 | 1558.55 | 1528.66 | 1515.52 | 2018.67 | 1992.02 | 2018.73 | 1998.47 |
| lc105 | 806.15 | 804.66 | 802.79 | 802.86 | 1565.46 | 1554.95 | 1555.37 | 1514.62 | 1946.66 | 1959.47 | 1962.75 | 1974.67 |
| lc106 | 724.29 | 723.78 | 719.46 | 720.96 | 1519.68 | 1548.41 | 1548.86 | 1530.58 | 2121.51 | 2110.11 | 2122.29 | 2151.89 |
| lc107 | 763.50 | 767.71 | 759.94 | 761.82 | 1614.94 | 1628.71 | 1628.44 | 1606.14 | 1994.93 | 1973.85 | 1997.07 | 2029.57 |
| lc108 | 804.27 | 800.69 | 801.12 | 801.12 | 1562.36 | 1574.53 | 1523.89 | 1508.41 | 2069.93 | 2038.03 | 2074.32 | 2101.60 |
| lc109 | 721.13 | 722.08 | 719.46 | 720.07 | 1566.65 | 1573.33 | 1557.22 | 1529.92 | 1973.00 | 1931.84 | 1946.47 | 1956.72 |

**Table 9** continued

| σ | 10% | | | | 30% | | | | 40% | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance/ε | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| lrc | | | | | | | | | | | | |
| lrc101 | 1008.73 | 1008.73 | 1008.73 | 1008.73 | 1989.40 | 1983.89 | 2061.3 | 2069.71 | 2499.44 | 2481.41 | 2481.09 | 2475.91 |
| lrc102 | 1002.42 | 1002.42 | 1002.42 | 1002.42 | 1826.15 | 1861.89 | 1841.01 | 1906.66 | 2471.91 | 2472.86 | 2459.51 | 2473.06 |
| lrc103 | 1007.95 | 1007.95 | 1011.6 | 1007.95 | 1883.99 | 1878.24 | 1900.43 | 1943.96 | 2415.88 | 2427.55 | 2371.84 | 2420.01 |
| lrc104 | 1007.38 | 1007.38 | 1007.38 | 1007.38 | 1862.02 | 1859.7 | 1861.93 | 1894.14 | 2395.26 | 2345.99 | 2410.29 | 2372.8 |
| lrc105 | 1016.38 | 1016.38 | 1016.38 | 1016.38 | 1932.55 | 1938.78 | 1942.64 | 1973.02 | 2392.51 | 2392.03 | 2379.21 | 2345.97 |
| lrc106 | 1014.55 | 1014.55 | 1014.55 | 1014.55 | 1901.24 | 1904.35 | 1933.47 | 1962.95 | 2392.79 | 2328.66 | 2333.88 | 2398.99 |
| lrc107 | 1012.72 | 1012.72 | 1013.33 | 1012.72 | 1921.26 | 1970.79 | 1965.62 | 2011.12 | 2601.34 | 2538.88 | 2530.89 | 2556.10 |
| lrc108 | 1011.03 | 1012.11 | 1011.03 | 1011.03 | 1862.98 | 1885.99 | 1891.92 | 1898.30 | 2523.17 | 2511.32 | 2468.85 | 2484.33 |

**Table 10** Detailed solutions for the transformed VRPCD data set with collocated nodes and split options (Data Set II)

| σ | 10% | | | | 30% | | | | 40% | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance/ε | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| LR | | | | | | | | | | | | |
| LR1_2_1 | 2773.97 | 2803.45 | 2856.76 | 2827.79 | 5697.90 | 5660.80 | 5623.51 | 5658.57 | 7252.00 | 7277.62 | 7328.01 | 7353.93 |
| LR1_2_2 | 2782.07 | 2835.49 | 2793.11 | 2789.61 | 5553.50 | 5533.49 | 5508.75 | 5526.32 | 7022.90 | 7136.52 | 7113.62 | 7088.01 |
| LR1_2_3 | 2865.31 | 2889.63 | 2808.82 | 2816.28 | 5835.19 | 5798.40 | 5820.06 | 5823.12 | 6888.66 | 6935.09 | 6987.29 | 6941.87 |
| LR1_2_4 | 2850.38 | 2743.8 | 2794.83 | 2795.61 | 5629.74 | 5620.23 | 5614.20 | 5618.24 | 6954.23 | 6974.97 | 6991.49 | 7013.56 |
| LR1_2_5 | 2791.65 | 2863.57 | 2777.49 | 2778.59 | 5559.51 | 5660.84 | 5560.41 | 5556.38 | 6783.37 | 6826.06 | 6846.15 | 6867.30 |
| LR1_2_6 | 2801.26 | 2831.61 | 2782.99 | 2784.96 | 5781.66 | 5827.82 | 5808.10 | 5839.04 | 6864.68 | 6891.4 | 6915.88 | 6930.72 |
| LR1_2_7 | 2806.14 | 2734.29 | 2811.87 | 2825.59 | 5596.45 | 5574.75 | 5560.84 | 5599.97 | 6921.38 | 6902.21 | 6976.76 | 6898.74 |
| LR1_2_8 | 2830.45 | 2791.49 | 2780.52 | 2822.44 | 5735.52 | 5709.92 | 5701.08 | 5739.75 | 7103.08 | 7082.39 | 7076.27 | 7102.47 |
| LR1_2_9 | 2881.92 | 2738.87 | 2833.87 | 2787.06 | 5673.88 | 5642.80 | 5654.44 | 5674.85 | 6879.25 | 7013.27 | 7028.66 | 6967.95 |
| LR1_2_10 | 2789.49 | 2802.01 | 2778.02 | 2779.08 | 5857.95 | 5787.98 | 5775.58 | 5786.13 | 6979.77 | 7015.37 | 7016.67 | 7018.14 |
| LC | | | | | | | | | | | | |
| LC1_2_1 | 2329.48 | 2306.24 | 2314.15 | 2329.48 | 5232.88 | 5253.25 | 5295.87 | 5289.13 | 7148.52 | 7240.07 | 7281.98 | 7327.87 |
| LC1_2_2 | 2415.45 | 2335.60 | 2347.64 | 2415.45 | 5280.75 | 5301.26 | 5303.85 | 5292.28 | 7291.12 | 7470.23 | 7478.52 | 7476.74 |
| LC1_2_3 | 2401.37 | 2405.44 | 2422.35 | 2401.37 | 5253.02 | 5365.01 | 5408.38 | 5365.70 | 6789.89 | 6865.97 | 6918.31 | 6914.91 |
| LC1_2_4 | 2407.32 | 2413.94 | 2422.54 | 2405.96 | 5459.50 | 5559.86 | 5566.34 | 5553.28 | 6892.99 | 7074.70 | 7031.09 | 7054.35 |
| LC1_2_5 | 2395.98 | 2400.53 | 2423.59 | 2395.98 | 5214.71 | 5270.79 | 5251.24 | 5243.55 | 7356.01 | 7392.62 | 7430.01 | 7457.24 |
| LC1_2_6 | 2417.07 | 2403.91 | 2420.64 | 2417.07 | 5527.85 | 5633.32 | 5661.41 | 5666.40 | 7029.16 | 7148.57 | 7172.69 | 7182.25 |
| LC1_2_7 | 2400.86 | 2339.49 | 2323.57 | 2340.45 | 5222.21 | 5268.77 | 5255.19 | 5288.83 | 7086.97 | 7200.59 | 7237.34 | 7238.18 |
| LC1_2_8 | 2402.54 | 2408.59 | 2419.77 | 2411.39 | 5417.41 | 5444.16 | 5515.50 | 5506.80 | 6913.33 | 7058.66 | 7090.68 | 7087.27 |
| LC1_2_9 | 2349.37 | 2281.76 | 2263.39 | 2273.94 | 5310.33 | 5291.03 | 5277.41 | 5285.37 | 6805.90 | 6959.56 | 6918.42 | 6928.78 |
| LC1_2_10 | 2403.54 | 2403.37 | 2417.56 | 2403.54 | 5307.83 | 5339.86 | 5313.12 | 5295.32 | 6740.70 | 6809.06 | 6800.72 | 6801.31 |

**Table 10** continued

| $\sigma$ | 10% | | | | 30% | | | | 40% | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance/ε | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| LRC | | | | | | | | | | | | |
| LRC1_2_1 | 2627.72 | 2645.88 | 2658.29 | 2654.86 | 5523.30 | 5523.20 | 5524.26 | 5512.14 | 7201.85 | 7152.59 | 7137.75 | 7216.56 |
| LRC1_2_2 | 2632.54 | 2627.70 | 2656.47 | 2624.08 | 5537.32 | 5523.88 | 5613.31 | 5558.96 | 6761.08 | 6687.38 | 6764.65 | 6708.51 |
| LRC1_2_3 | 2630.26 | 2610.88 | 2609.63 | 2610.88 | 5436.61 | 5434.22 | 5423.21 | 5374.95 | 6900.20 | 6846.20 | 6874.49 | 6818.94 |
| LRC1_2_4 | 2652.76 | 2639.24 | 2639.24 | 2634.14 | 5469.09 | 5485.85 | 5474.29 | 5492.36 | 7057.22 | 6962.15 | 7043.01 | 7006.89 |
| LRC1_2_5 | 2637.13 | 2629.33 | 2614.44 | 2644.09 | 5544.30 | 5563.57 | 5533.66 | 5541.87 | 7201.95 | 7145.12 | 7136.57 | 7185.97 |
| LRC1_2_6 | 2646.03 | 2640.87 | 2634.28 | 2624.11 | 5676.42 | 5654.44 | 5638.50 | 5632.26 | 6890.04 | 6906.95 | 6829.99 | 6829.53 |
| LRC1_2_7 | 2658.93 | 2657.59 | 2669.32 | 2707.27 | 5486.07 | 5490.38 | 5488.62 | 5477.60 | 6984.99 | 6899.35 | 6959.19 | 6956.53 |
| LRC1_2_8 | 2644.32 | 2636.30 | 2649.20 | 2636.30 | 5446.05 | 5428.69 | 5448.99 | 5452.41 | 6873.16 | 6858.18 | 6834.44 | 6902.70 |
| LRC1_2_9 | 2647.96 | 2644.04 | 2652.63 | 2650.81 | 5530.64 | 5477.44 | 5515.56 | 5494.81 | 6537.70 | 6481.12 | 6506.87 | 6481.45 |
| LRC1_2_10 | 2622.50 | 2606.97 | 2624.39 | 2603.65 | 5499.23 | 5504.05 | 5448.44 | 5492.19 | 6543.87 | 6472.55 | 6487.78 | 6513.56 |

## References

Bartholdi JJ III, Gue KR (2004) The best shape for a cross-dock. Transp Sci 38(2):235–244

Cardona-Valdés Y, Álvarez A, Pacheco J (2014) Metaheuristic procedure for a bi-objective supply chain design problem with uncertainty. Transp Res Part B Methodol 60(8):66–84

Clark PJ, Evans FC (1954) Distance to nearest neighbor as a measure of spatial relationships in populations. Ecology 35:445–453

Cohen Y, Keren B (2009) Trailer to door assignment in a synchronous cross-dock operation. Int J Logist Syst Manag 5(5):574–590

Dondo R, Mèndez CA, Cerdá J (2011) The multi-echelon vehicle routing problem with cross-docking in supply chain management. Comput Chem Eng 35(12):3002–3024

Enderer F (2014) Integrating dock-door assignment and vehicle routing in cross-docking. MSc Thesis, Concordia University

Glover F (1997) Tabu search and adaptive memory programming—advances, applications and challenges. In: Barr RS, Helgason RV, Kennington JL (eds) Interfaces in computer science and operation research: advances in metaheuristics. Kluwer, Boston, pp 1–75

Gounaris CE, Repoussis PP, Tarantilis CD, Wiesemann W, Floudas C (2014) An Adaptive memory programming framework for the robust capacitated vehicle routing problem. Transp Sci. doi:10.1287/trsc.2014.0559

Grangier P (2016) Résolution de problèmes de tournées avec synchronisation: applications au cas multi-échelons et au cross-docking. Dissertation, École nationale supérieure des Mines de Nantes

Guastaroba G, Speranza MG, Vigo D (2016) Intermediate facilities in freight transportation planning: a survey. Transp Sci 50(3):763–789

Lee YH, Jung JW, Lee KM (2006) Vehicle routing scheduling for cross-docking in the supply chain. Comput Ind Eng 51(2):247–256

Li H, Lim A (2001) A metaheuristic for the pickup and delivery problem with time windows. In: IEEE proceedings of the international conference on tools with artificial intelligence. pp 160–167

Liao ChJ, Lin Y, Shih SC (2010) Vehicle routing with cross-docking in the supply chain. Expert Syst Appl 37:6868–6873

Ma H, Miao Z, Lim A, Rodriguez B (2011) Cross-docking distribution networks with setup cost and time window constraint. Omega 39:64–72

Morais VWC, Mateus GR, Noronha TF (2014) Iterated local search heuristics for the vehicle routing problem with cross-docking. Expert Syst Appl 41(16):7495–7506

Musa R, Arnaout JP, Jung H (2010) Ant colony optimization algorithm to solve for the transportation problem of cross-docking network. Comput Ind Eng 59(1):85–92

Nikolopoulou AI, Repoussis PP, Tarantilis CD, Zachariadis EE (2017) Moving products between location pairs: cross-docking versus direct-shipping. Eur J Oper Res 256(3):803–819

Paraskevopoulos DC, Tarantilis CD, Ioannou G (2016) An adaptive memory programming framework for the resource-constrained project scheduling problem. Int J Prod Res 54(16):3938–4956

Petersen HL, Ropke S (2011) The pickup and delivery problem with cross-docking opportunity. In: Proceedings of international conference on computational logistics, Hamburg, Germany. pp 101–113

Repoussis PP, Tarantilis CD (2010) Solving the fleet size and mix vehicle routing problem with time windows via adaptive memory programming. Transp Res Part C 18:695–712

Santos FA, Mateus GR, da Cunha AS (2013) The pickup and delivery problem with cross-docking. Comput Oper Res 40:1085–1093

Sung CS, Song SH (2003) Integrated service network design for a cross-docking supply chain network. J Oper Res Soc 54(12):1283–1295

Taillard ED, Gambardella LM, Gendreau M, Potvin JY (2001) Adaptive memory programming: a unified view of metaheuristics. Eur J Oper Res 135(1):1–16

Tarantilis CD (2005) Solving the vehicle routing problem with adaptive memory programming methodology. Comput Oper Res 32(9):2309–2327

Tarantilis CD (2013) Adaptive multi-restart tabu search algorithm for the vehicle routing problem with cross-docking. Optim Lett 7(7):1583–1596

Tarantilis CD, Stauropoulou F, Repoussis PP (2011) A template-based tabu search algorithm for the consistent vehicle routing problem. Expert Syst Appl 39(4):4233–4239

Tarantilis CD, Anagnostopoulou A, Repoussis PP (2013) Adaptive path relinking for vehicle routing and scheduling problems with product returns. Transp Sci 47:356–379

Tsui LY, Chang CH (1992) An optimal solution to a dock door assignment problem. Comput Ind Eng 23(1–4):283–286

Vis IF, Roodbergen KJ (2008) Positioning of goods in a cross-docking environment. Comput Ind Eng 54(3):677–689

Wen M, Larsen J, Clausen J, Cordeau JF, Laporte G (2009) Vehicle routing with cross-docking. J Oper Res Soc 60(12):1708–1718

Whittley IM, Smith GD (2004) The attribute based hill climber. J Math Model Algorithms 3(2):167–178

Zachariadis EE, Kiranoudis CT (2010) A strategy for reducing the computational complexity of local search-based methods for the vehicle routing problem. Comput Oper Res 37(12):2089–2105

Zachariadis EE, Tarantilis CD, Kiranoudis CT (2015) The load-dependent vehicle routing problem and its pick-up and delivery extension. Transp Res Part B Methodol 71:158–181