A Tabu Search Algorithm for the Vehicle Routing Problem with Discrete Split Deliveries and Pickups

Meng Qiu [a], Zhuo Fu [a, ∗], Richard Eglese [b], Qiong Tang [a, c]

[a] *School of Traffic and Transportation Engineering, Central South University, Changsha 410075, PR China.*
[b] *Lancaster University Management School, Lancaster, UK*
[c] *College of Economics and Management, Hengyang Normal University, Hengyang, 421008, PR China*

The Vehicle Routing Problem with Discrete Split Deliveries and Pickups is a variant of the Vehicle Routing Problem with Split Deliveries and Pickups, in which customers' demands are discrete in terms of batches (or orders). It exists in the practice of logistics distribution and consists of designing a least cost set of routes to serve a given set of customers while respecting constraints on the vehicles' capacities. In this paper, its features are analyzed. A mathematical model and tabu search algorithm with specially designed batch combination and item creation operation are proposed. The batch combination operation is designed to avoid unnecessary travel costs, while the item creation operation effectively speeds up the search and enhances the algorithmic search ability. Computational results are provided and compared with other methods in the literature, which indicate that in most cases the proposed algorithm can find better solutions than those in the literature.

Keywords: routing; pickup and delivery; discrete split; tabu search.

1. Introduction

In the classical Vehicle Routing Problem (VRP), customers have only a single (delivery or pickup) demand. However, the increasing focus on environmental protection has led to the development of reverse logistics; in addition to distribution to customers, recycled or remanufactured items have to be transported in the reverse direction. Variants of the VRP have been proposed in response to such conditions and are named VRPs with Deliveries and Pickups (VRPDPs) in this paper. Parragh et al. (2008) classified VRPDPs into four sub-problems: the VRP with Clustered Backhauls (VRPCB), VRP with Mixed Linehauls and Backhauls (VRPMB), VRP with Simultaneous Delivery and Pickup (VRPSDP) and VRP with Divisible Delivery and Pickup (VRPDDP). The

∗ Corresponding author. *Z Fu, School of Traffic and Transportation Engineering, Central South University, Changsha 410075, P.R.China*. Tel.: +86 13973134613. Fax: +86 731 82655315. *E-mail:* zhfu@csu.edu.cn

linehaul and backhaul customers in the VRPCB and VRPMB are different, which means that all the customers can only have one kind of demand (delivery or pickup). The difference between the VRPCB and the VRPMB is that the former requires that all linehauls must be visited before any backhaul, and the latter allows linehauls and backhauls to occur in any sequence during a trip. In contrast to the VRPCB and VRPMB, customers in the VRPSDP and VRPDDP can have both delivery and pickup demands. In the VRPSDP, the restriction of serving these two demands per customer at the same time must be satisfied. However, in the VRPDDP it is not required that every customer is only visited once. Rather, two visits, one for delivery and one for pickup are possible.

Considering that customers may prefer one vehicle stop for the sake of convenience, the VRP and VRPDP always assume that each customer can only be visited once, belonging to problems without split demand. However, it is not rare to find that the demands of customers are transported by a variety of vehicles, and both theoretical research and practical applications have proved that splitting loads is beneficial by taking advantage of the vehicle capacity and reducing the vehicle travel cost. Problems of this type are classified as ones with split demand. Since Dror and Trudeau (1989) introduced the Split Delivery Vehicle Routing Problem (SDVRP), which is well known in the literature, a growing number of academics have worked in the field of split demand. To cater to the split case of VRPDPs, Mitra (2005, 2008) proposed the Vehicle Routing Problem with Split Deliveries and Pickups (VRPSPDP).

In the literature on problems with split loads, most are assumed to be continuous ones, in which customers' demands can be split flexibly into loads of any amount in terms of units (minimum unit of measure). This assumption has a practical application background but also has limitations. In an actual logistics operation, loads may be assigned to a number of independent batches or orders. One batch or order, composed of several units or only one unit, is taken as a whole and cannot be split. This means that customers' demands are discrete. For example, one unit can be one kilogram (kg) or one cubic meter (m^3) according to whether the loads are measured by weight or volume. If a laptop weighs 4kg (units), it cannot be split into four parts with each part weighing one unit, so the laptop can be seen as a discrete batch (order). Considering another situation, because diverse types of cargoes may require different loading and unloading equipment and conveyance, allocating the same kind of loads to one batch (order) as a whole may be preferred for efficiency. To the best of our knowledge, the VRP with Discrete Split Deliveries and Pickups (VRPDSPDP) is presented in the literature for the first time in this study.

The contribution of this study on the VRPSDPDP is twofold: firstly, it describes the problem and establishes a corresponding mathematical model, analyses the features of the VRPDSPDP and presents the optimal solution properties of the problem; secondly, it develops a tabu search algorithm with two individual operations designed to avoid unnecessary travel costs, speed up the search and enhance the algorithmic search ability. The proposed heuristic is tested, and the results are compared with those in the related literature.

2. Literature Review

2.1. Problems with Split Loads

Splitting loads such that the delivery of certain loads is conveyed by multiple vehicles rather than one results in opportunities for a reduction in number of vehicles used and travel distance. Several studies have shown the benefit of split loads for the classic VRP. (Dror et al., 1989, 1990; Belenguer et al., 2000; Ho and Haugland, 2004; Archetti et al., 2006a, 2006b, 2008a, 2008b, 2011, 2014; Derigs et al., 2010; Moreno et al., 2010; Wilck IV and Cavalier, 2012a, 2012b; Berbotto et al., 2014; Khmelev and Kochetov, 2015; Silva et al., 2015; Ozbaygin et al., 2018). The SDVRP is reviewed by Archetti and Speranza (2008), while in Golden et al. (2008), a more general overview of the literature where split deliveries are considered is presented. More recently, Archetti and Speranza (2012) provided a survey on the SDVRP that also overviews its variants and in general all routing problems that consider split deliveries.

There has been nearly 30 years since the earliest concept of split (Dror et al., 1989, 1990), and in the previous 10 years, split was only considered in single demand problems (VRPs) until the first paper on the VRPMB with split loads (SVRPMB) appeared in Mosheiov (1998). His model created several fictitious co-located customers each with unit demand for each original customer. The first paper studied on the VRPSPDP is presented by Mitra (2005). In the VRPSPDP, split deliveries and pickups are permitted, which implies that each customer may be visited by more than one vehicle and more than once by the same vehicle. In Mitra (2005), a MILP formulation is presented and the same problem is also studied in Mitra (2008). On the base of Mitra's research, Wang et al. (2010) considered time windows in the VRPSPDP. Yin et al. (2013) proposed a special mathematical model for the VRPSPDP, in which two new pre-conditions were assumed. One was an identical maximum travel distance constraint for all vehicles, and the other restricted each customer's demand to be split only once. In connection to the VRPSDP, the VRPDDP is an interesting split case problem, in which the customers may have both delivery and pickup demands and be served twice. There is a significant distinction between the VRPDDP and the VRPSPDP. While at most two separate visits to each customer in the VRPDDP are allowed, one visit for deliveries and the other for pickups, there is no restriction on the number of visits in the VRPSPDP. Thus, the authors consider that regarding the VRPDDP as a constrained VRPSPDP is reasonable. Assuming that each customer's delivery and pickup demands in the VRPDDP are two discrete batches (orders), the VRPDDP can be viewed as a problem with discrete split loads.

2.2. Problems with Discrete Demands

To the best of our knowledge, the concept of discrete demands was introduced by Nakao and Nagamochi (2007), who defined a VRP with such a requirement as a VRP with Discrete Split Demand (DSDVRP). In this problem, each customer requires the demand of different batches (orders). The demand of each batch (order) can be greater than one (unit) and, while a customer can be visited more than once, each item has to be served by exactly one vehicle. Following Nakao and Nagamochi, Salani and Vacca (2011) added a restriction of time windows to the DSDVRP. Chen et al. (2017) proposed two *a priori* split strategies and aimed to split each customer's demands into several discrete items. The next paper to focus on discrete demands was presented by Fu et al. (2017), who introduced a related problem to the DSDVRP named the VRP with Soft Time Windows Split Deliveries by Order (VRPSTWSDO). Xia et al. (2018) extended the research on the VRPSTWSDO.

There are other papers on problems related to the DSDVRP. Gulczynski et al. (2010) considered a special SDVRP in which split deliveries are allowed only if a minimum fraction of a customer's demand is serviced by a vehicle and named the problem the SDVRP with Minimum Delivery Amounts (SDVRP-MDA). Minimum delivery can be regarded as a batch (order) that cannot be split any further; thus, the SDVRP-MDA falls into the field of the DSDVRP. Han and Chu (2016) also proposed the SDVRP-MDA with a more restricted constraint that the minimum demand amounts cannot exceed half of the vehicle capacity.

In this paper, the VRPDDP can be classified as a special case of problem with discrete demands. Anily (1996) firstly presented the study on the VRPCB, which considers divisible pickups and deliveries. The VRPDDP itself was introduced by Salhi and Nagy (1999) and Nagy and Salhi (2005). Nagy and Salhi (2005) proposed a new set of routines devised specially for the VRPDDP, such as Neck (split a customer into a linehaul and backhaul) and Unneck (merge the linehaul and backhaul of the same customer). Recently Wassan et al. (2013), Nagy et al. (2015) and Polat (2017) focused on this interesting but rarely addressed problem. A brief list of literature for problems with discrete demands is illustrated in Table 1. However, limited work has considered discrete demands in the VRPSPDP. To the best of our knowledge, the VRP with Discrete Split Deliveries and Pickups (VRPDSPDP) is presented in the literature for the first time in this study.

Table 1

Literature for problems with discrete demands

Paper	Problem	Paper	Problem
Nakao and Nagamochi (2007)	DSDVRP	Anily (1996)	DVRPCB
Gulczynski et al. (2010)	SDVRP-MDA	Salhi and Nagy (1999)	VRPDDP
Han and Chu (2016)	SDVRP-MDA	Nagy and Salhi (2005)	VRPDDP
Chen et al. (2017)	DSDVRP	Wassan et al. (2013)	VRPDDP
Fu et al. (2017)	VRPSTWSDO	Nagy et al. (2015)	VRPDDP
Xia and Fu (2018)	VRPSTWSDO	Polat (2017)	VRPDDP

2.3. Tabu Search and Local Search Algorithms

For the sake of conciseness, our review is restricted to VRPDPs only. Koç and Laporte (2018) provided a general overview of papers for the VRPDPs until 2017 with detailed comparison of computational performance of solution methods, while Irnich et al. (2014) provided a general overview of VRPDPs papers from 2002 to 2014, without comparisons of computational performance. Parragh et al. (2008) comprehensively reviewed the pickup and delivery problems until 2007. Since the VRPDPs are NP-hard, more advanced procedures rather than simple interchange schemes have been developed in these years. Koc and Laporte (2018) pointed out that the main methods are the constructive heuristics integrated with Tabu Search (TS) and local search algorithms.

For the problems without split loads, Osman and Wassan (2002) firstly introduced a reactive TS for the VRPCB. The reactive concept is used to speed up the neighborhood structures for intensification and diversification phases. Later, Wassan (2007) combined the TS by Osman and Wassan (2002) with an adaptive memory programming scheme, which is capable of maintaining a set of elite solutions for searching the unexplored regions of the solution space and has been extended for further research on the VRPSDP (Wassan, 2008), the VRPDDP (Wassan, 2013) and the VRP with Restricted mixing of Deliveries and Pickups (VRPRMDP) (Nagy et al., 2013). Crispim and Brandão (2005) proposed a hybrid algorithm for both VRPMB and VRPSDP that combines TS and variable neighborhood descent and Brandão (2006) presents a new TS algorithm that starts from pseudo-lower bounds. Chen and Wu (2006) developed a hybrid heuristic based on the record-to-record travel (RRT) method of Dueck (1993) and tabu lists from the TS. Meanwhile, Yu and Qi (2014) proposed two TS algorithms based on RRT for an industrial application case study. Hoff et al. (2009) studied lasso solution strategies for the VRPDPs and developed a TS heuristic that is capable of generating lasso solutions. Avci and Topaloglu (2016) studied a heterogeneous VRPSDP (HVRPSDP) and developed a hybrid local search algorithm in which a non-monotone threshold adjusting strategy is integrated with tabu search. In addition, there are other relevant papers applying TS in the literature, such as Alfredo Tang Montané and Galvão (2006), Bianchessi and Righini (2007), Zachariadis et al. (2009).

Until 2013, several heuristics mainly based on constructive methods have been developed for the problems with split loads. Two heuristics of a Tour Partitioning (TP) type were presented for the SVRPMB: Exhaustive Iterated Tour Partitioning (EITP) and Full Capacity Iterated Tour Partitioning (FCITP). Mitra (2005) proposed a route construction heuristic for the first VRPSPDP, on the basis of which a parallel clustering technique was developed three years later (Mitra, 2008). Wang et al. (2010) introduced a novel cluster-first-routing-second heuristic, Competitive Decision Algorithm (CDA), to solve the problem of VRPSPDP with Time windows (VRPSPDPTW). In 2015, Wang et al. achieved better solutions for the VRPSPDP using the CDA. The only TS heuristic for the VRPSPDP was introduced by Yin et al. (2013), except the Adaptive Guidance (AG) mechanism with TS from Archetti et al. (2006b) for the Vehicle Routing Problem with Splits and Clustered Backhauls (SVRPCB) (Lai et al., 2015). There are three VRPDDP papers in the literature more recently. Wassan and Nagy (2013) and Nagy et al. (2015) studied the problem both using the meta-heuristic of reactive Tabu Search algorithm, and Polat (2017) solved the problem by a parallel variable neighborhood search. A brief list of literature for VRPDPs with split loads is presented in Table 2.

Table 2

Literature for VRPDPs with split loads

Paper	Problem	Heuristic
Polat (2017)	VRPDDP	Parallel Variable Neighborhood
Wang et al. (2015)	VRPSPDP	Competitive Decision Algorithm (CDA)
Nagy et al. (2015)	VRPDDP	Tried-and-tested Reactive Tabu Search Algorithm of Wassan et al. (2008)
Lai et al. (2015)	SVRPCB	Adaptive Guidance (AG) Meta-Heuristic
Wang et al. (2014)	VRPSPDP	Two-stage Heuristic
Yin et al. (2013)	VRPSPDP	Tabu Search (TS) Algorithm
Wassan and Nagy (2013)	VRPDDP	Reactive Tabu Search (RTS) Meta-Heuristic
Wang et al. (2013)	VRPSPDPTW	Two-stage Heuristic
Wang et al. (2010)	VRPSPDPTW	Competitive Decision Algorithm (CDA)
Mitra (2008)	VRPSPDP	Parallel Clustering Algorithm
Mitra (2005)	VRPSPDP	Constructive Algorithm (Tour Partitioning Algorithm)
Mosheiov (1998)	SVRPMB	Exhaustive Iterated Tour Partitioning (EITP)
		Full Capacity Iterated Tour Partitioning (FCITP) Heuristics

3. Problem Description and Properties

The VRPDSPDP aims to design the optimal set of routes to serve all customers with demands at the lowest cost (with the fewest vehicles and shortest trip distance) while satisfying the constraints as follows:

(1) All the vehicles depart from and return to the same depot;

(2) All the vehicles are homogeneous, and overloading is not allowed;

(3) Customers may have both delivery and pickup demands, either of which may exceed the vehicle capacity;

(4) Each customer can be visited by a variety of vehicles or several times by one vehicle;

(5) Linehauls and backhauls can occur in any sequence along a vehicle route;

(6) There is no restriction on time windows or maximum route length.

In general, the fixed costs associated with owning or hiring one more vehicle are much larger than the saving by shortening the trip distance at the cost of using more vehicles. Thus, we set the minimum number of vehicles used as an input parameter, which is calculated by the equation introduced by Mitra (2005):

$$K=\left\lceil \frac{\max\left(cumulative\ \ delivery\ \ demand,\ cumulative\ \ pickup\ \ demand\right)}{vehicle\ \ capacity} \right\rceil$$

Let $G=(V,E)$ be a given undirected network, where $V=\{0\}\cup N$ is the vertex set (vertices $i=1,\ldots,n$ correspond to the customers, while vertex 0 corresponds to the depot) and $E=\{(i,j)\,|\,i,j\in V, i\neq j\}$ is the edge set. Each edge $(i,j)\in E$ corresponds to a non-negative c_{ij}, representing the cost travelling from vertex *i* to vertex *j*. The maximum vehicle capacity is *Q*, and the number of vehicles used is *K*. We set DM_i and PM_i as the number of delivery batches and pickup batches of customer *i*, respectively. d_i^m ($m=1,\ldots,DM_i$) is the demand load of the *m*th delivery batch of customer *i*; thus, $D_i=\sum_{m=1}^{DM_i} d_i^m$ expresses the total amount of delivery goods of customer *i*. p_i^m ($m=1,\ldots,PM_i$) is the *m*th pickup batch demand of customer *i*; $P_i=\sum_{m=1}^{PM_i} p_i^m$ is the total amount of pickup goods of customer *i*.

3.1. Formulation of the VRPDSPDP

The following are decision variables:

d_{ij}^k : amount of delivery loads moved from customer *i* to customer *j* by vehicle *k*;

p_{ij}^k : amount of pickup loads moved from customer *i* to customer *j* by vehicle *k*;

x_{ij}^k : equals 1 if vehicle *k* travels from customer *i* to customer *j*; otherwise, it equals 0;

y_{im}^k : equals 1 if the *m*th delivery batch (order) of customer *i* is transported by vehicle *k*; otherwise, it equals 0;

z_{im}^k : equals 1 if the *m*th pickup batch (order) of customer *i* is transported by vehicle *k*; otherwise, it equals 0.

The formulation of the VRPDSPDP modified on the base of Mitra (2005) is given below:

$$MinZ = \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=1}^{K} c_{ij} x_{ij}^k \quad (1)$$

s.t.

$$\sum_{i=0}^{n}\sum_{k=1}^{K} x_{ij}^k \geq 1, \quad j = 1,...,n \quad (2)$$

$$\sum_{i=1}^{n} x_{i0}^k = 1, \quad k = 1,...,K \quad (3)$$

$$\sum_{j=1}^{n} x_{0j}^k = 1, \quad k = 1,...,K \quad (4)$$

$$\sum_{i=0}^{n} x_{ip}^k = \sum_{j=0}^{n} x_{pj}^k, \quad p = 1,...,n, \;\; k = 1,...,K \quad (5)$$

$$\sum_{k=1}^{K}\sum_{m=1}^{DM_i} d_i^m y_{im}^k = D_i, \quad i = 1,...,n \quad (6)$$

$$\sum_{k=1}^{K}\sum_{m=1}^{PM_i} p_i^m y_{im}^k = P_i, \quad i = 1,...,n \quad (7)$$

$$d_{ij}^k + p_{ij}^k \leq Q, \quad i, j = 0,...,n; \;\; k = 1,...,K \quad (8)$$

$$x_{ij}^k \in \{0,1\}, \quad i, j = 0,...,n; \;\; k = 1,...,K \quad (9)$$

$$y_{im}^k \in \{0,1\}, \quad i = 1,...,n; \;\; m = 1,...,DM_i; \;\; k = 1,...,K \quad (10)$$

$$z_{im}^k \in \{0,1\}, \quad i = 1,...,n; \;\; m = 1,...,PM_i; \;\; k = 1,...,K \quad (11)$$

We present next a brief line-by-line explanation of this formulation.

(1) The objective is to minimize the total distance travelled by the vehicles;

(2) Each customer is served at least once;

(3) – (4) Vehicles leave from the depot and return to the depot;

(5) Vehicles visit a customer and then leave;

(6) – (7) Each customer's demand is satisfied;

(8) Maximum capacity constraint;

(9) – (11) Set $x_{ij}^k, y_{im}^k, z_{im}^k$ as 0 – 1.

3.2. Problem Properties

Each batch (order) of a customer is an absolute object, which is independent from another. Because of the same coordinate position, the distance between a pair of batches of the same customer is zero.

Before discussing the properties of the VRPDSPDP, the expression of paths is explained. As stated in the above section, the delivery and pickup demands of customer *i* are composed of DM_i and PM_i batches (orders), respectively. Thus, a path is expressed as a sequence composed of vertex numbers representing the depot or visits to a customer, together with the demands of batches (delivery and/or pickup) dealt with at that visit. For instance, there is a path expressed as follows:

$$0 — 6\ (d_6^9\ ,\ p_6^5\ ) — 8\ (\ 0\ ,\ p_8^1 + p_8^8\ ) — 0$$

A vehicle departs from the depot, visits customer 6 first, unloads the ninth delivery batch (order) and collects the fifth pickup batch with demands of d_6^9 and p_6^5, respectively. After leaving customer 6, the vehicle makes a trip to visit customer 8. Since the amount of delivery load of customer 8 is zero, the vehicle does not have a delivery plan at customer 8. However, the vehicle conveys the first and eighth pickup batches of customer 8 back to the depot.

To study the properties of the VRPSPDP, we discuss an example illustrated in Fig. 1, there are 3 customers in the network (the figures in brackets represent the total demands of delivery and pickup batches, separated by a comma). The figure beside each edge denotes the corresponding cost (distance), and the cost matrix satisfies the triangular inequality. The vehicle capacity is 10, so the minimum number of vehicles required is 2.

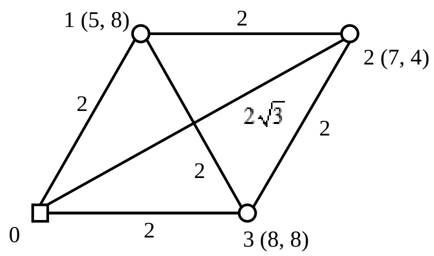


Fig. 1 Total demands and the corresponding cost (distance) of the example discussed

While the demands in the VRPSPDP are continuous and can be split in any way to fit the available capacity of a vehicle, those in the VRPDSPDP are discrete and cannot be split as arbitrarily as in the VRPSPDP. To clarify the customer delivery and pickup values for both the problems compared. We assume the demands in the VRPSPDP are split in terms of units as shown in Fig. 2, and those in the VRPDSPDP are split into several independent batches illustrated in Figure 3.

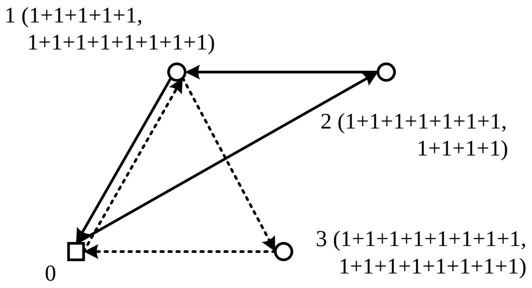


Fig. 2 Optimal solution to the VRPSPDP

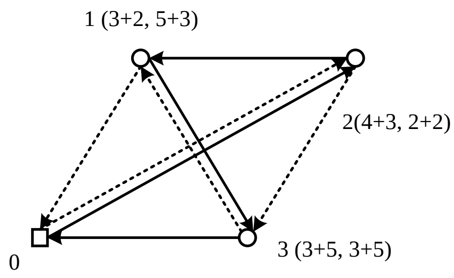


Fig. 3 Optimal solution to the VRPDSPDP

Path 1: 0 – 2(7, 4) – **1(3, 6)** – 0

Path 2: 0 – **1(2, 2)** – 3(8, 8) – 0

Path 1: 0 – **2(4, 2)** – **1(3, 5)** – **3(3, 3)** – 0

Path 2: 0 – **2(3, 2)** – **3(5, 5)** – **1(2, 3)** – 0

The paths of the optimal solutions are presented below the corresponding problem illustrations. The figures in bold are split points, and the two numbers in the brackets beside each point, separated by a comma, indicate the amount of delivery goods and the amount of pickups, respectively.

Three points are split in the above solution to the given VRPDSPDP, more than in the VRPSPDP (two split points). In addition, the total distance of the VRPSPDP ($10+2\sqrt{3}=13.46$) is shorter than that of the VRPDSPDP (18.93).

We can obtain the properties of the VRPDSPDP as follows:

(1) If *L* and *L'* correspond to the numbers of split points of the VRPSPDP and the VRPDSPDP, respectively, there always exists $L \le L'$ with the same cost matrix, customer demands and other restrictions.

(2) If *Z* and *Z'* correspond to the objective function values of the optimal solution to the VRPSPDP and the VRPDSPDP, respectively, there always exists $Z \le Z'$ with the same cost matrix, customer demands and other restrictions.

(3) The VRPDSPDP is a special version of the VRPSPDP, which is an NP-hard problem (Yin et al., 2013); the VRPDSPDP is also an NP-hard problem.

4. A Tabu Search Algorithm for the VRPDSPDP

Previous research on similar problems suggests that large instances may not be solved by exact algorithms and that meta-heuristics will provide a more effective solution approach. Our algorithm is based on Tabu Search introduced by Glover in 1986, and since then it has been used to solve many practical applications. TS is a memory-based search strategy to guide the local search method to continue its search beyond a local optimum. One way to achieve this is to keep track of attributes of recent moves or solutions made in the past in a tabu list. Whenever the algorithm attempts to make a move in the tabu list, the move is banned. Consequently, other solutions must be explored. However, this feature is not strict; it can be overridden when some aspiration criteria are satisfied. A popular aspiration criterion is for the target function value to be the best ever seen. If this requirement is met, it is obvious that this solution is encountered for the first time and can be accepted.

4.1. Initial Solution

The tabu search algorithm has some dependence on the initial solution. A good initial solution can help the TS find a good final solution in the solution space, while a poor one can reduce the convergence speed of the TS. In general, when solving a specific problem, other algorithms can be used to generate a high-quality initial solution and then TS is used to further improve the quality of the solution. The solution from a well-designed TS should not depend strongly on the quality of the initial solution and if multiple starts are used for the TS, then there may be some advantages in the diversity gained from different initial solutions. Therefore, it is not rare to find initial solutions generated randomly. A survey on initial solution generation methods for the TS in recent years (from 2000), which focus on the problems with split loads is shown in Table 3.

Table 3

Initial solution generation methods for the TS focusing on the problems with split loads

Method	Literature
Random generating	Xia et al. (2018); Avci (2017); Fu et al. (2017)
Savings-based procedure	Polat et al. (2015)
Sweep-based procedure	Nagy et al. (2015); Nagy et al. (2013)
GENIUS algorithm	Lai et al. (2015)
Nearest Neighborhood Heuristic	Yin et al. (2013)

In this paper, each batch (order) of a customer is an absolute object as a separate item in the beginning. That is, there exist $DM_i + PM_i$ co-located fictitious customers. Nagy et al.. (2015) only created two co-located customers and have already run into difficulties solving even small problems to optimality. Due to the heavy problem complexity, the running time of the TS is required to be as short as possible. Since the TS is more important than the quality of the initial solution, we prefer to generate an initial solution randomly with less time-consuming process and leave more opportunities to the TS for further solution quality improvement. Thus, the initial solution is generated by building up successive routes on which the next item is chosen at random and added to the end of the route unless this violates the capacity constraint, in which case the route back to the depot is completed and a new route is started. The number of vehicle routes used in the initial solution may be more than the minimum required, but the tabu search will aim to reduce the number of vehicle routes if possible.

4.2. Batch Combination

Since each discrete batch (order) is regarded as an independent object, the situation exists in which the same customer is visited more than once in one trip. Taking Fig. 4 as an example, vertices *i* and *j* correspond to two different batches of the same customer. Visiting a customer twice on the same route (once for batch *i* and once for batch *j*) will normally increase the route distance compared to visiting the customer once. To avoid this situation, we design a special operation to move batches of the same customer into one visit, where it is feasible to do so, by assigning vertex *j* next to vertex *i*. As shown in Fig. 4, besides moving vertex j behind vertex i, relocating j in front of i ($\mathrm{L} \to a \to j \to i \to b \to \mathrm{L}$) or putting i next to j ($\mathrm{L} \to c \to i \to j \to d \to \mathrm{L}$ or $\mathrm{L} \to c \to j \to i \to d \to \mathrm{L}$) may gain different results. Preventing from exploring multiple solutions with the same cost and reasonably shorten the operation time, we only consider the movement illustrated in Figure 4. This move is a basic operator in the path improvement of the VRPDSPDP, laying the foundation for a further neighbourhood structure.

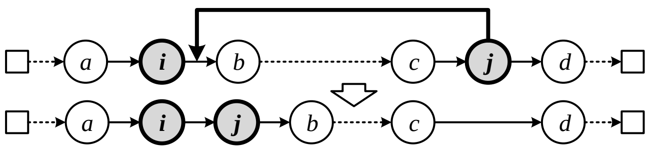


Fig. 4 Batch combination

4.3. Item Creation

In traditional neighbourhood searching, there is an underlying assumption that the move object is an individual customer. However, the operator object of the VRPDSPDP is actually each batch (order); thus, simply using the customer whose batches in the route are all included cannot reflect the characteristic of discrete splits. However, taking a single batch as the operator object results in a computation-intensive and time-consuming process, because the number of batches is much larger than that of customers. Considering the above situation, we reach a compromise and design an individual operator aimed to create a neighbourhood move object, which corresponds to an "item" in this paper. The operation of item creation first selects a batch vertex randomly from a selected path and then searches both forward and backward to check whether other batches exist that have the same customer origin. After that, one or more adjacent vertices in a path representing batches for a customer dealt with in a single visit are selected at random to create an item. The item created through the designed operation is composed of one, several or all batches of a customer on the route. As illustrated in Fig. 5, the picked-out vertex is *i*, and, by checking forwards and backwards, *t*, *i*, *j* and *k* are shown to relate to the same customer. It is assumed that *i*, *j* and *k* are the selected adjacent batch vertices to create an item expressed as vertex *i**. In the rest of this paper, we refer to each vertex of a route path as an item for the sake of conciseness.

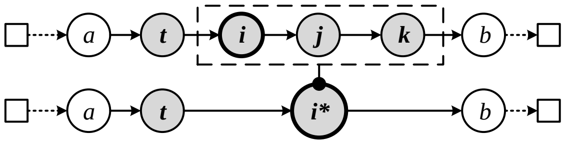


Fig. 5 Item creation

The “splitshift” operator proposed by Nagy et al. (2015) for the VRPDDP is worth mentioning at this point. The “splitshift operator duplicates a customer where the delivery and pickup happen simultaneously on the current route and inserts either its linehaul or backhaul into another route. Merge operators often accompany split operators, however, if a delivery and pickup entity of a customer should find themselves next to each other in subsequent moves, Nagy et al. (2015) think that a separate operator is not required to bind them together again. The Item Creation can be seen as related to the “splitshift”. The new operator considers all demands of one customer, including *t*, *i*, *j* and *k*, which are currently served in a single visit. As shown in Fig. 5, Assume that *t* denotes the delivery batch, and *i*, *j* and *k* are pickup batches, then the Item Creation splits the customer into a delivery item (*t*) and a pickup item (*i*, *j* and *k*), respectively, which implements the same as “splitshift” in this case.

4.4. Neighbourhood Structure

Local search methods have been proved effective in solving the VRPSDP (Subramanian et al., 2010; Zhang et al., 2012). Several operators from the local search method are 2-opt, 2-opt*, 3-opt, or-opt, swap, shift, reverse, cross, relocate and relocate split, which are applied to find the solution to the VRPSPDP or VRPDDP and further illustrated in Table 4.

Except the Relocate operator and Relocate split operator proposed by Ho and Haugland (2004) which are designed specially to focus on split loads, all the local search methods listed in Table 4 consider the moves on customers, no matter whose demands are split or not. In our heuristic, due to the operation of Item Creation, the difference between a customer and the items for delivery or pickup can be ignored for the operation of these moves. In our implementation the neighbourhood moves of five classes are applied to the current solution. These moves include two intra-route operations and three inter-route ones. Table 5 provides a brief introduction.

Table 4

Local search methods for the VRPSPDP or VRPDDP

Type	Operator	Detail	Literature
Intra-route	2-opt	Replace a non-adjacent arc pair with a new one, which reverses the location of nodes lying between these new arcs. (Croes, 1958)	Polat (2017); Lai et al. (2015); Wang et al. (2014, 2013); Yin et al. (2013).
	3-opt	Delete three edges in a route and reconnect them in the same route. (Lin, 1965)	Polat (2017).
	Or-opt	Remove one, two or three adjacent customers and insert in another position of the route. (Or, 1976)	Wang et al. (2014, 2013).
	Shift	Randomly select a customer and move it to a random position on the same route.	Polat (2017); Nagy et al. (2015); Wassan and Nagy (2013).
	Swap	Randomly swap positions of two customers on the same route.	Polat (2017); Yin et al. (2013).
	Reverse	Simply reversing the direction of a route.	Nagy et al. (2015); Wassan and Nagy (2013).
Inter-route	2-opt* 2-Exchange	Remove two links from two different routes and introduce two new ones by connecting the first customer on the first link to the last customer on the second link and connecting the first customer on the second link to the last customer on the first link. (Potvin et al., 1992)	Wang et al. (2014, 2013).
	Exchange (*m*, *n*)	Transfer *m* sequential customers from one route to another and in turn transfer *n* sequential customers from the second route to the first. (Osman, 1993)	Polat (2017); Yin et al. (2013).
	Cross	Exchange two segments of different routes.	Polat (2017).
	Shift (1, 0)	Move a customer from a route to the best possible position on another route.	Polat (2017); Nagy et al. (2015); Wassan and Nagy (2013).
	Replace (1, 1) Swap (1, 1)	Reallocate two customers currently on different routes.	Polat (2017); Nagy et al. (2015); Lai et al. (2015); Wassan and Nagy (2013).
	Swap (*m*, *n*)	Interchange *m* customers and *n* customers between two different routes	Wang et al. (2014, 2013).

Relocate	Reallocate two one-unit demand of customers currently on different routes. (Ho and Haugland, 2004)	Wang et al. (2014, 2013); Yin et al. (2013).
Relocate split	Delete a split node on a route in price of creating a new split node in another route. (Ho and Haugland, 2004)	Wang et al. (2014, 2013); Yin et al. (2013).

Table 5

Neighbourhood moves

Neighbourhood move	Operation object (item)	Effect of possible solution improvement		
		Combine batches of the same customer	Short route distance	Reduce the number of vehicles used
Intra-swap	Two different customers (all their batches)	Impossible	Possible	Impossible
Intra-reverse	Two different customers (all their batches)	Impossible	Possible	Impossible
Inter-reassignment	Fractional and all batches of two customers	Possible	Possible	Possible
Inter-swap	Fractional and all batches of two customers	Possible	Possible	Impossible
Tail-swap	Two tails (each begins with a batch to the end)	Possible	Possible	Impossible

Detailed explanations of the above moves are provided in the rest of this section. For intra-swap and intra-reverse, one route is selected at random from the current solution and then two items of different customers are created and selected for this route. Regarding the inter moves, inter-reassignment, inter-swap and tail-swap, two different routes are chosen randomly and one item is generated for each path.

4.4.1. Intra-swap

In this move the positions of two items of different customers (*i* and *j*) are swapped and a new route is generated, which is indicated by Fig. 6.

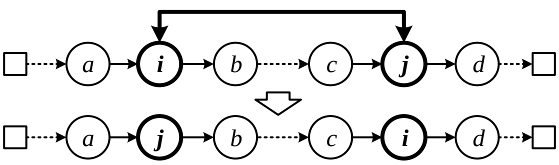


Fig. 6 Intra-swap

4.4.2. Intra-reverse

Each item in this move is a sub-path beginning with *i* and ending with *j*. The neighbourhood move of intra-reverse inverts the orders of all the batches in the sub-path and produces a new route, which is illustrated in Fig. 7.

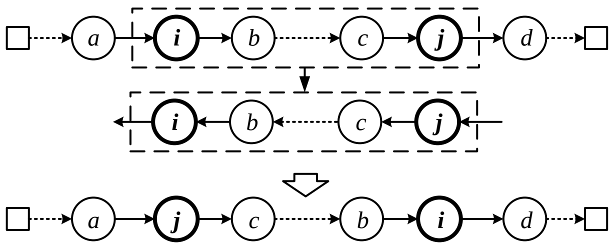


Fig. 7 Intra-reverse

4.4.3. Inter-reassignment

Assume that the randomly chosen routes are Route 1 and Route 2. Taking the example shown in Fig. 8, we remove item *i* from Route 1 and assign it after *j* to Route 2 in this move. Fig. 9 illustrates a special case: there is only one customer on Route 1, so, after moving *i* into Route 2, no vertex exists in Route 1. For this situation we design an operation named elimination. When the number of vehicles used in the current solution is more than the minimum required, elimination is implemented and generates a new solution, saving one vehicle.

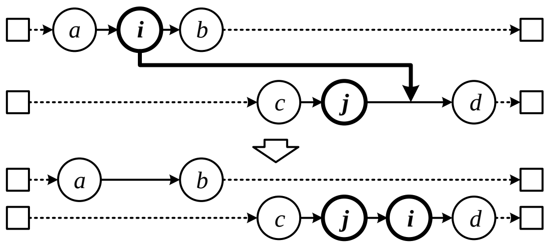


Fig. 8 Inter-reassignment

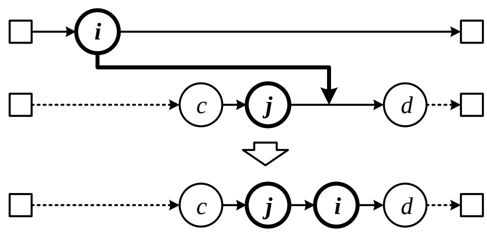


Fig. 9 Elimination

4.4.4. Inter-swap

In this move the positions of items *i* and *j* are exchanged and two new paths are obtained, as illustrated in Fig. 10.

4.4.5. Tail-swap

The operation of tail-swap is shown in Fig. 11; the tail $(i \rightarrow b \rightarrow \cdots)$ in Route 1 and the tail $(j \rightarrow d \rightarrow \cdots)$ in Route 2 are exchanged, and two new routes are generated.

This neighbourhood structure is the one that allows moves to infeasible solutions in terms of the vehicle capacity. This structure is able to enhance the TS algorithm.

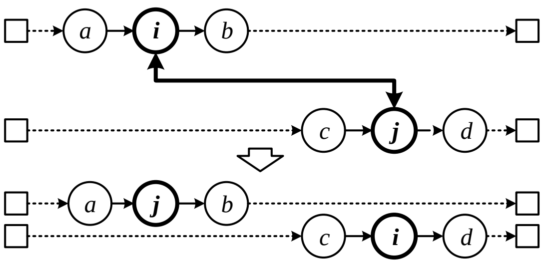


Fig. 10 Inter-swap

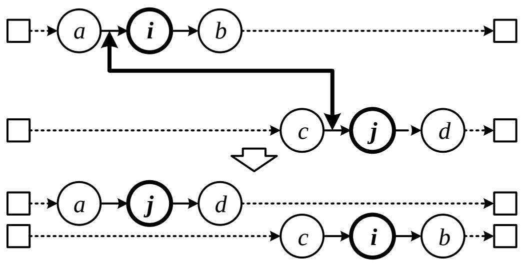


Fig. 11 Tail-swap

4.5. Evaluation of the Solutions

The VRPDSPDP has two objectives to be optimized: the total travelling cost (distance) and the fixed cost (the number of vehicles used). As mentioned in relation to the VRPDSPDP, the priority is given to the fixed cost. Therefore, a feasible solution with a certain number of vehicles always dominates any other feasible solutions requiring more vehicles. For those solutions with the same number of vehicles, the one with the minimum total travelling cost is selected.

To facilitate the exploration of the search space, a move is allowed even if it results in an infeasible solution. The extent of the infeasibility can be measured by incorporating the vehicle capacity into the objective function by adding a penalty if the constraints are broken. Gendreau et al. (2002) introduced penalties, produced a mix of feasible and infeasible solutions and avoided the possibility of being trapped in a local minimum in a tabu route algorithm for the VRP. We adopt the mechanism and use the equation given below:

$$\sum_{k=1}^{K}\left[E(r)+p\cdot A(r)\right]$$

where K is the total number of routes in the solution, $E(r)$ is the travelling cost of route r, $A(r)$ is the number of excess edges along route r and p is the penalty coefficient. $A(r)$ equals zero for all the routes if a solution is feasible. $p\in[0.000001, 200000]$ equals 1 at the beginning and is weighted by a self-adjusting parameter: every 10 iterations, it is divided by 2 if all 10 previous solutions were feasible or multiplied by 2 if they were all infeasible.

4.6. Tabu List

We create five independent tabu lists for different classes of moves. The idea of each tabu list is learned and adjusted from Fu et al. (2005). Each tabu list contains the move attributes of solutions during the last five to eight (randomly selected) iterations. A set of $n\times n$ matrices can be constructed for the record of tabu status. For instance, if vertices i and j are selected for the intra-swap move, the tabu status is saved in the elements (i, j) of the intra-swap matrix. At each iteration the tabu status of the last move performed is added to the list while the others are decreased by one until they equal zero.

4.7. Stopping Criterion

The search is terminated if a specified number of iterations has elapsed since the last best solution was found. The variables presented in Table 6 are used in the description of the TS algorithm:

Table 6

Variables and their explanation used in the description of the TS algorithm

Variable	Explanation
Iter	Current number of iterations.
ConsIter	Current number of consecutive iterations without any improvement to the best solution so far.
MaxConsIter	Maximum number of consecutive iterations without any improvement to the best solution so far.
CandList	Current number of candidate moves on the list.
MaxCandList	Maximum number of candidate moves on the list.

4.8. TS Algorithm

The pseudo-code of the heuristic is given below:

1. **Initialize**
2. Input the data and parameters;
3. Generate an initial feasible solution randomly and set it as the current solution and the best solution so far;
4. Construct five separate tabu lists of neighbourhood searching;
5. **While** (*ConsIter* < *MaxConsIter*) **do begin**
6. **While** (*CandList* < *MaxCandList*) **do begin**
7. Select one of the five types of neighbourhood move randomly and create corresponding items;
8. Perform a corresponding operation on the current solution;
9. If the condition of elimination is satisfied, then implement elimination to remove one route.
10. Conduct batch combination and generate a new candidate solution;
11. Add the solution produced by the selected move to the candidate list;
12. **End;**
13. Select the best solution in the candidate list if not tabu or a solution better than the best one so far;

14. Set the new solution as the current solution, update the tabu list and increment *Iter*;

15. If the new solution improves the best solution so far, update the best solution so far and set *ConsIter* to 0; otherwise, increment *ConsIter*;

16. Update the corresponding tabu list;

17. **End.**

5. Computational Results and Comparisons

5.1. A Priori Split Strategy

To the best of our knowledge, there is neither any benchmark problem designed for the VRPDSPDP, nor any split strategy to generate discrete demands for a corresponding VRPSPDP. Chen et al. (2017) proposed two *a priori* split strategies to split deliveries in the SDVRP in advance and not during the algorithm procedure, aiming to split each customer's demands into several parts of goods (discrete batches or orders) so that it is possible to make full use of a vehicle's capacity. We propose two similar split strategies, adjusted from Chen's and named 20/10/5/1/*x* and 25/10/5/1/*x*, to discrete demands in the VRPSPDP. Taking 20/10/5/1/*x* as an example, we assume that each customer's demands can only be split and assigned to five separate groups, each of which has a different quantity from another. Batches of four groups have fixed demands of 0.2*Q*, 0.1*Q*, 0.05*Q* and 0.01*Q*, respectively. The fifth group includes loads of quantity less than 0.01*Q*. D_i, the deliveries of customer *i*, will be split into five distinct groups $W_s\left(s=20,10,5,1,x\right)$. Each batch in W_s has a quantity demand of T_s, and the number of batches belonging to W_s is denoted as H_s. The strategy of 20/10/5/1/*x* is shown in Table 7 ($\lfloor u \rfloor$ is the maximum integer not bigger than *u*).

Table 7

Strategy of 20/10/5/1/*x*

W_s	T_s	H_s
W_{20}	$T_{20}=0.2Q$	$H_{20}=\lfloor d_i/T_{20} \rfloor$
W_{10}	$T_{10}=0.1Q$	$H_{10}=\lfloor \left(d_i-T_{20}H_{20}\right)/T_{10} \rfloor$
W_5	$T_5=0.05Q$	$H_5=\lfloor \left(d_i-T_{20}H_{20}-T_{10}H_{10}\right)/T_5 \rfloor$
W_1	$T_1=0.01Q$	$H_1=\lfloor \left(d_i-T_{20}H_{20}-T_{10}H_{10}-T_5H_5\right)/T_1 \rfloor$
W_x	$T_x=d_i-T_{20}H_{20}-T_{10}H_{10}-T_5H_5-T_1H_1$	$H_x=\begin{cases}1 & T_x>0\\ 0 & T_x=0\end{cases}$

We also propose a 25/10/5/1/*x* strategy, which has the same basic principle as 20/10/5/1/*x*. For example, if $Q=1000$ and $D_i=566$, the demand is split into 200, 200, 100, 50, 10 and 6 by the 20/10/5/1/*x* strategy and into 250, 250, 50, 10 and 6 by the 25/10/5/1/*x* strategy. We use both strategies in our computational experiments.

5.2. Performance of the a Priori Split Strategy

We apply our TS algorithm with 20/10/5/1/*x* and 25/10/5/1/*x* split strategies to different classes of VRPSPDP instances and report the results in this section. All the experiments were performed on a personal computer with an Intel i7-4500U CPU 2.40 GHz and 12 GB RAM. In our TS algorithm, we set variables *MaxConsIter* and *MaxCandList* equal to $4500+10*n$ and $150+2*n$, respectively.

5.2.1. Experiment 1

The data for this experiment were randomly generated by Mitra (2008). Before applying the TS algorithm in this paper to the instances, we discretized the delivery and pickup loads of customers through the strategies of 20/10/5/1/*x* and 25/10/5/1/*x*, separately. For the sake of exposition, discretized batches for delivery and pickup for each customer for each strategy are presented in Appendix A. To show the solution performance, a comparison including the corresponding routes, delivery and pickup loads of customers along routes and the total route distances was performed, as shown in Table 8. The best-known result, a total distance of 60, was given by Mitra (2008). Our problem solution in both situations (20/10/5/1/*x* and 25/10/5/1/*x*) is 56, which is better than Mitra's with a 6.67% distance reduction. As presented in Table 8, customer 7 is split by two routes in our solution instead of customer 4 in Mitra's solution.

Table 8

Solution comparison for Experiment 1

	Route	Path	Total distance
Mitra (2008)	1	0 – 2(0.40, 0.40) – 6(0.50, 0.40) – **4(0.10, 0.20)** – 0	60
	2	0 – 1(0.30, 0.30) – 8(0.30, 0.20) – 5(0.20, 0.20) – 0	
	3	0 – 7(0.30, 0.30) – 3(0.30, 0.40) – **4(0.30, 0.30)** – 0	
20/10/5/1/*x*	1	0 – 5(0.20, 0.20) – 8(0.30, 0.20) – 1(0.30, 0.30) – **7(0.20, 0.00)** – 0	**56***
	2	0 – 4(0.40, 0.50) – 3(0.30, 0.40) – **7(0.10, 0.30)** – 0	
	3	0 – 2(0.40, 0.40) – 6(0.50, 0.40) – 0	
25/10/5/1/*x*	1	0 – 5(0.20, 0.20) – 8(0.30, 0.20) – 1(0.30, 0.30) – **7(0.05, 0.29)** – 0	**56***
	2	0 – **7(0.25, 0.01)** – 3(0.30, 0.40) – 4(0.40, 0.50) – 0	
	3	0 – 2(0.40, 0.40) – 6(0.50, 0.40) – 0	

Figures 12 and 13 present the fluctuations of the total loads on each trip and the illustrations of each demand. It is obvious that the deliveries decrease while the pickups increase. Experiment 1 shows the proposed heuristic is capable of producing a good solution to a problem studied in the literature.

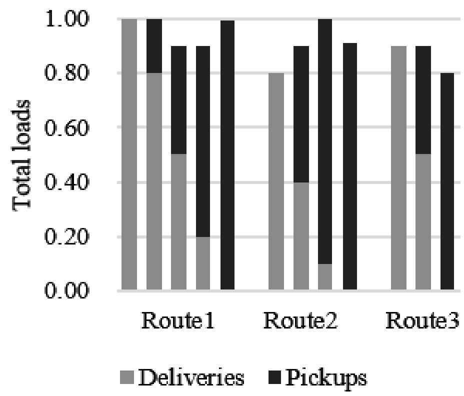


Fig. 12 Load changes in the solution obtained by 20/10/5/1/*x*

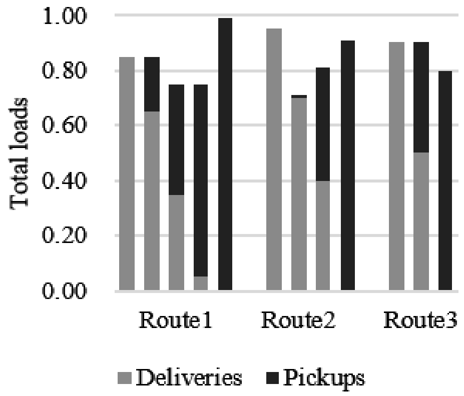


Fig. 13 Load changes in the solution obtained by 25/10/5/1/*x*

5.2.2. Experiment 2

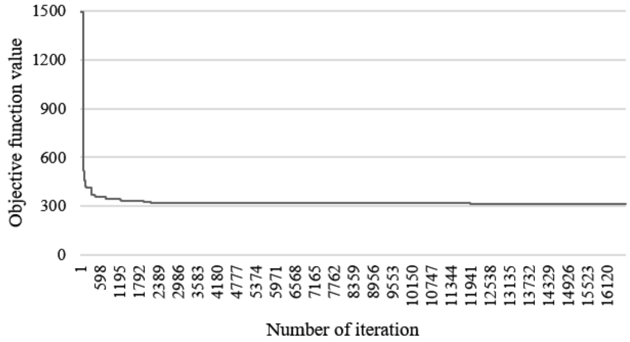
The data provided by Yin et al. (2013) were used in this experiment. The solution comparison is presented in Table 9. The results obtained by our algorithm (both strategy 20/10/5/1/*x* and strategy 25/10/5/1/*x*) are better than the best-known one (Yin et al., 2013). The solution achieved by strategy 20/10/5/1/*x* (308.74) is the best one so far, which reduces by 3.55% the total distance of Yin et al. (320.11). The solution of 25/10/5/1/*x* (311.87) is also better than Yin's solution.

Table 9

Solution comparison for Experiment 2

Route	Yin et al. (2013)	20/10/5/1/*x*	25/10/5/1/*x*
1	0-20-**22(2.50,0.60)**-4-3-0	0-5-8-19-**24(1.00,0.00)**-0	0-21-**24(2.50,0.00)**-9-0
2	0-21-11-10-**24(1.00,0.00)**-9-0	0-13-17-16-15-14-7-0	0-**3(0.50,0.00)-22(3.75,5.00)**-0
3	0-8-9-0	0-1-**22(3.00,4.00)**-0	0-2-5-19-8-0
4	0-**23(0.00,0.30)**-18-17-13-15-16-12-0	0-3-**22(0.00,1.60)**-4-2-0	0-**24(3.75,0.00)**-6-4-**3(0.00,0.16)**-0
5	0-5-7-14-1-6-2-0	0-20-**22(4.50,5.00)**-0	0-13-15-14-1-7-**22(0.00,1.25)**-0
6	0-**22(5.00,5.00)**-0	0-**24(5.00,0.00)-23(0.00,4.25)**-10-0	0-11-**23(0.00,0.80)**-18-16-17-12-0
7	0-**24(5.00,0.00)-22(0.00,5.00)**-0	0-**24(5.00,0.00)**-9-0	0-20-**22(3.75,4.35)-3(0.50,0.00)**-0
8	0-**24(5.00,0.00)-23(0.00,5.00)**-0	0-21-11-18-**23(0.00,1.05)**-12-0	0-**24(4.75,0.00)-23(0.00,4.50)**-10-0
Distance	320.11	**308.74***	**311.87**

To check further the convergence of our TS heuristic, we provide two illustrations (Fig. 14 and Fig.15) of the convergence process, which both show a good performance.



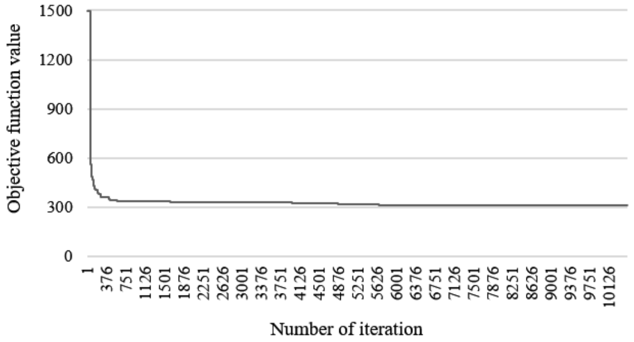


Fig. 14 Convergence process of the results of 20/10/5/1/*x* **Fig. 15** Convergence process of the results of 25/10/5/1/*x*

In addition to convergence, the solution quality is another vital criterion on which to judge an algorithm. Yin et al. (2013) used a fluctuation coefficient of the solution to evaluate the solution quality, which is calculated by the equation below:

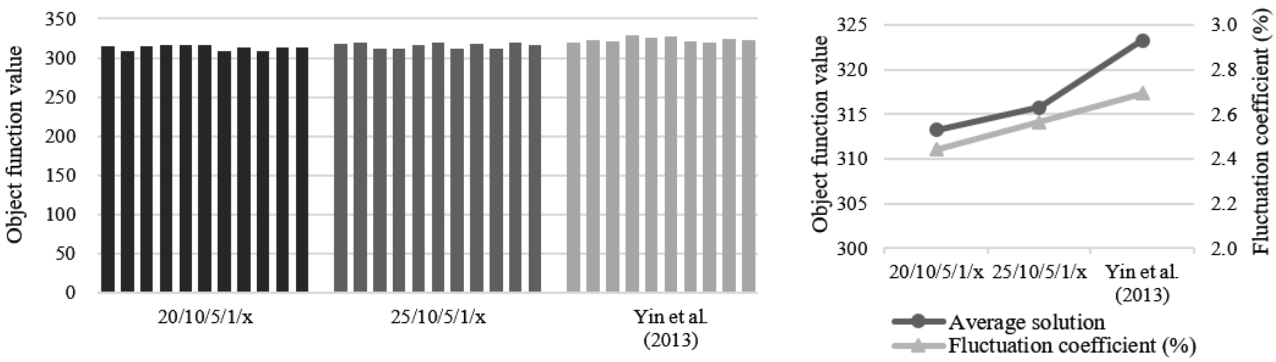
$$fluctuation\ \ coefficient = \frac{\max value - \min value}{average\ \ value} \times 100\%$$

We list ten results compared with Yin et al. (2013) in Table 10. Fig. 16 further illustrates the quality comparison over ten runs. The fluctuation coefficients of our algorithm (2.4% of 20/10/5/1/*x* and 2.57% of 25/10/5/1/*x*) are both better than those of Yin et al. (2013) (2.69%). In addition, the average solution values from our proposed algorithm are lower than the average solution in Yin et al. (2013).

Table 10

Comparison of ten results

	Yin et al. (2013)	20/10/5/1/*x*	25/10/5/1/*x*		Yin et al. (2013)	20/10/5/1/*x*	25/10/5/1/*x*
1	320.11	314.76	317.86	6	326.99	316.39	319.97
2	322.44	308.74	319.21	7	320.95	308.74	312.10
3	320.53	315.22	312.10	8	320.11	313.92	317.63
4	328.82	315.95	311.87	9	323.76	308.84	311.87
5	325.68	315.65	315.97	10	323.07	313.92	319.06
Average solution					323.25	313.21	315.76
Fluctuation coefficient					2.69%	**2.44%***	**2.57%**



(*a*) (*b*)

Fig. 16 Fluctuation comparison between results from the proposed method and results from Yin et al. (2013)

By analysing the convergence and solution quality, the algorithm indicates good convergence with a stable solution quality.

5.2.3. Experiment 3

Experiments 1 and 2 presented good performances of our heuristic. To confirm further the effective computation capability of our TS algorithm, more instances (problems of three sets given by Mitra (2005) for the VRPSPDP) are tested in this experiment. The distance between each pair of customers has two cases. (For both the cases, the route costs are symmetric). The delivery demand and pickup demand for each customer are listed in Appendix B.

Case 1: All the route costs are equal. $c_{ij} = 10, \ \forall i, j$ such that $j > i$ and $c_{ii} = \infty, \ \forall i$.

Case 2: All the route costs are not equal. $c_{ij} = 9 + j - i, \ \forall i, j$ such that $j > i$ and $c_{ii} = \infty, \ \forall i$.

For each instance our heuristic is run for 20 minutes (maximum), which is less than the 30 minutes of Mitra (2008). The same data were also used by Wang et al. (2015). However, in Wang et al. (2015) the objective of the VRPSPDP was to minimize the total travel distance and the number of assigned vehicles was not restricted. As a result, in Wang et al. (2015) a few shortest distances are found using more vehicles. Because of the much higher cost assumed for an extra vehicle compared with the distance related costs, the solution with the fewest vehicles and shortest route length is regarded as the best-known one in this paper. The comparison of our results for the VRPDSPDP and those for the VRPSPDP is presented in Table 11 and Table 12 in terms of both the same result and better ones. The column 3 in Table 12 shows the vehicle number of each problem. For each problem, the number of vehicles required are the same under *Case* 1 and *Case* 2. The minimum and actual number of assigned vehicles of optimal / upper bound solution, 20/20/5/1/*x*, 25/10/5/1/*x*, Mitra et al. (2008), Wang et al. (2015) and the best-known, separated by “ / ” , are given in column 3.

Table 11

Comparison between results from the proposed method and the best-known ones

Case	Set	Number of instances	Same result		Better result		Same and better results	
			20/10/5/1/*x*	25/10/5/1/*x*	20/10/5/1/*x*	25/10/5/1/*x*	20/10/5/1/*x*	25/10/5/1/*x*
Case 1	Set 1	25	17	20	0	0	17 (68.00%)	20 (80.00%)
	Set 2	9	2	1	6	5	8 (88.89%)	6 (66.67%)
	Set 3	9	0	1	7	6	7 (77.78%)	7 (77.78%)
Case 2	Set 1	25	9	20	0	0	9 (100.00%)	20 (80.00%)
	Set 2	9	0	0	8	6	8 (88.89%)	6 (66.67%)
	Set 3	9	0	0	9	9	9 (100.00%)	9 (100.00%)

Both Mitra (2008) and Wang et al. (2015) obtained the problem solutions with same (minimum) number of vehicles used for Set 1. 14 out of 25 best-known solutions for Set 1 are proved to be optimal (bold underlined data in columns named “Opt./UB”). Since the optimal solution to the VRPDSPDP cannot be better than that of the VRPSPDP, which means that the optimum of the VRPDSPDP may be worse than that of the VRPSPDP (as seen in *Problem Properties*), it is reasonable that the results of Set 1 of the VRPDSPDP are no better than those of the VRPSPDP. For Sets 2 and 3, we find the best-known solution or improve it in at least two thirds of the instances. The new best-known results are marked by “*” in Table 12. As seen in Set 2, 8 out of 9 instance solutions are better than or the same as the best-known ones by means of the strategy of 20/10/5/1/*x* and 6 out of 9 by that of 25/10/5/1/*x*. In terms of Set 3, we achieve 77.78% of the results of *Case* 1 that are no worse than those in the literature and 100% of those of *Case* 2 that are better than the best-known ones. We apply the hypothetical situation created by Mitra (2008), in which the fixed charge and mileage for owning or hiring a vehicle are taken as $100 and $0.10, separately. Because Wang et al. (2015) obtained all problem solutions of the minimum number of vehicles for Set 1, we only compare the problems of Sets 2 and 3. The result comparison for the above hypothetical situation is presented in Tables 12 and 13.

As shown in Table 13, over 88.89% of the results of the VRPDSPDP are better than or the same as those of the VRPSPDP.

5.3 Performance on the DSDVRP

To better judge the quality of the proposed algorithm, another set of computational experiment comparisons are made focusing on the DSDVRP. The data provided by Chen et al. (2017) were used in this experiment. The solution comparison is presented in Tables 14 and 15, and all the instance solutions are solved with the minimal numbers of

Table 12

Comparison of the results from the proposed method, Mitra (2008), Wang et al. (2015) and the best-known

Set	No.	Number of vehicles used	*Case* 1						*Case* 2					
			Opt./UB	20/10/5/1/*x*	25/10/5/1/*x*	Mitra (2008)	Wang (2015)	Best-known	Opt./UB	20/10/5/1/*x*	25/10/5/1/*x*	Mitra (2008)	Wang (2015)	Best-known
Set 1	1	2 / **2 / 2** / 2 / 2 / 2	240	**210**	**210**	210	210	210	566	**245**	**245**	263	245	245
	2	10/**10/10**/10/10/10	290	**290**	**290**	290	290	290	554	**461**	**461**	515	461	461
	3	19/**19/19**/19/19/19	**380**	**380**	**380**	380	380	380	**722**	**722**	**722**	722	722	722
	4	29/**29/29**/29/29/29	790	680	680	710	670	670	1303	1411	1185	1303	1183	1183
	5	38/**38/38**/38/38/38	**760**	780	**760**	760	760	760	**1444**	1475	**1444**	1444	1444	1444
	6	10/**10/10**/10/10/10	320	**290**	**290**	290	290	290	545	**461**	**461**	515	461	461
	7	19/**19/19**/19/19/19	**380**	**380**	**380**	380	380	380	**722**	**722**	**722**	722	722	722
	8	19/**19/19**/19/19/19	**380**	**380**	**380**	380	380	380	**722**	**722**	**722**	722	722	722
	9	29/**29/29**/29/29/29	740	**670**	**670**	750	670	670	1353	1187	**1183**	1365	1183	1183
	10	29/**29/29**/29/29/29	740	**670**	**670**	750	670	670	1296	1193	**1183**	1365	1183	1183
	11	29/**29/29**/29/29/29	710	**670**	**670**	710	670	670	1283	1202	**1183**	1321	1183	1183
	12	38/**38/38**/38/38/38	**760**	**760**	**770**	760	760	760	**1444**	1462	**1444**	1444	1444	1444
	13	38/**38/38**/38/38/38	**760**	**760**	770	760	760	760	**1444**	1453	1453	1444	1444	1444
	14	38/**38/38**/38/38/38	**760**	770	770	760	760	760	**1444**	1453	1462	1444	1444	1444
	15	38/**38/38**/38/38/38	**760**	770	**760**	760	760	760	**1444**	1462	1453	1444	1444	1444
	16	10/**10/10**/10/10/10	290	**290**	**290**	290	290	290	543	**461**	**461**	515	461	461
	17	19/**19/19**/19/19/19	**380**	**380**	**380**	380	380	380	**722**	**722**	**722**	722	722	722
	18	29/**29/29**/29/29/29	730	**670**	**670**	750	670	670	1364	1187	**1183**	1383	1183	1183
	19	38/**38/38**/38/38/38	**760**	780	780	760	760	760	**1444**	1466	**1444**	1444	1444	1444
	20	19/**19/19**/19/19/19	**380**	**380**	**380**	380	380	380	**722**	**722**	**722**	722	722	722
	21	29/**29/29**/29/29/29	740	**670**	**670**	750	670	670	1379	1206	**1183**	1383	1183	1183
	22	38/**38/38**/38/38/38	**760**	**760**	**760**	760	760	760	**1444**	1468	**1444**	1444	1444	1444
	23	29/**29/29**/29/29/29	710	680	**670**	710	670	670	1357	1203	**1183**	1303	1183	1183
	24	38/**38/38**/38/38/38	**760**	780	**760**	760	760	760	**1444**	1464	**1444**	1444	1444	1444

	25	38/**38/38**//38/38/38	**760**	780	780	760	760	760	**1444**	1462	1453	1444	1444	1444
Set 2	1	27/**27/27**/27/28/27	730	**630***	**630***	710	680	710	1648	**1268***	**1280**	1379	1295	1379
	2	27/**27/27**/27/28/27	740	**620***	**640**	710	680	710	1341	**1269**	**1263***	1379	1295	1379
	3	27/**27/27**/27/28/27	800	**640***	**660**	700	680	700	1491	**1291***	**1311**	1442	1259	1442
	4	37/**37/37**/37/37/37	990	**830***	**850**	890	850	850	1714	**1617***	**1624**	1773	1639	1639
	5	37/**37/37**/37/37/37	950	**860**	**840***	870	860	860	1731	**1615**	**1608***	1719	1639	1639
	6	37/**37/37**/37/37/37	920	**860**	870	900	860	860	1794	1674	1667	1818	1650	1650
	7	46/**46/46**/46/47/46	1090	**1070**	**1070**	1080	1060	1080	2107	**2005***	2443	2119	1981	2119
	8	46/**46/46**/46/47/46	1170	**1050***	1090	1080	1060	1080	2423	**2032***	2176	2146	1981	2146
	9	46/**46/46**/46/47/46	1130	1090	1200	1070	1060	1070	2948	**2089***	**2115**	2191	1981	2191
Set 3	1	27/**27/27**/27/28/27	810	**630***	**650**	710	680	710	1526	**1247***	**1256**	1379	1259	1379
	2	27/**27/27**/27/28/27	830	**640**	**630***	710	680	710	1646	**1259***	**1284**	1397	1259	1397
	3	27/**27/27**/27/28/27	800	**630***	**680**	700	680	700	1920	**1289***	**1315**	1397	1259	1397
	4	37/**37/37**/37/37/37	910	**840***	**840***	890	850	850	1834	**1617***	**1619**	1791	1639	1639
	5	37/**37/37**/37/37/37	900	**830**	**820***	870	850	850	1879	**1615***	**1636**	1719	1639	1639
	6	37/**37/37**/37/37/37	980	880	880	900	860	860	1857	**1639***	**1639***	1746	1650	1650
	7	46/**46/46**/46/47/46	1190	**1020***	**1070**	1080	1060	1080	2519	**1989***	**2107**	2119	1981	2119
	8	46/**46/46**/46/47/46	1170	**1050***	**1080**	1080	1060	1080	2136	**2036***	**2080**	2155	1981	2155
	9	46/**46/46**/46/47/46	1090	1090	1130	1070	1060	1070	2326	**2070***	**2121**	2137	1981	2137

Table 13

Comparison between the results from the proposed method and those from Wang et al. (2015)

Set	Strategy	Same number of vehicles				Fewer vehicles				Same and lower cost	
		Same distance/cost		Shorter distance/lower cost		Shorter distance		Lower cost			
		Case 1	*Case* 2	*Case* 1	*Case* 2	*Case* 1	*Case* 2	*Case* 1	*Case* 2	*Case* 1	*Case* 2
Set 2	20/10/5/1/*x*	2	0	1	2	4	0	6	6	9 (100.00%)	8 (88.89%)
	25/10/5/1/*x*	0	0	2	2	0	0	6	6	8 (88.89%)	8 (88.89%)
Set 3	20/10/5/1/*x*	0	0	2	2	2	2	6	6	8 (88.89%)	8 (88.89%)
	25/10/5/1/*x*	0	0	2	2	0	1	6	6	8 (88.89%)	8 (88.89%)

Table 14

Comparison between results from the proposed method and those from Chen et al. (2017)

	Number of instances	Same result	Better result			Same and better results
			Only fewer vehicles	Only shorter distance	Both fewer vehicles and shorter distance	
Same vehicles	13	3	\	4	\	7 (53.85%)
Fewer vehicles	12	\	10	\	2	10 (100.00%)
Sum	25	3	10	4	2	17 (68.00%)

Table 15

Instance results from Chen et al. (2017) and the proposed method

Instance	Minimal number of vehicles used	Chen et al. (2017)		20/10/5/1/*x*		25/10/5/1/*x*	
		Number of Vehicles used	Distance	Number of Vehicles used r	Distance	Number of Vehicles used	Distance
eil22	4	4	375.28	4	**375.28**	4	**375.28**
eil23	3	3	568.56	3	**568.56**	3	**568.56**
eil30	3	3	497.53	3	512.72	3	512.72
eil33	4	4	826.41	4	837.67	4	837.67
eil51	5	5	524.61	5	**524.61**	5	**524.61**
eilA76	10	11	849.60	**10**	860.11	**10**	853.83
eilB76	14	15	1024.44	**14**	1037.93	**14**	1047.26
eilC76	8	8	748.51	8	**745.92**	8	**744.71***
eilD76	7	7	684.53	7	719.05	7	699.34
eilA101	8	8	814.51	8	824.09	8	826.00
eilB101	14	14	1099.21	14	**1098.95***	14	1116.82
S51D1	3	3	459.50	3	**457.67***	3	**458.29**
S51D2	9	10	716.83	**9**	**714.05***	**9**	717.57
S51D3	15	16	964.83	**15**	971.46	**15**	975.76
S51D4	27	28	1592.23	**27**	1624.55	**27**	1673.90
S51D5	23	24	1371.41	**23**	1392.15	**23**	1399.96
S51D6	41	43	2240.46	**41**	2360.93	**41**	2310.32
S76D1	4	4	614.31	4	**600.19**	4	**599.41***
S76D2	15	16	1120.71	**15**	1413.73	**15**	1422.75
S76D3	23	24	1445.23	**23**	1702.57	**23**	1709.04
S76D4	37	37	2138.64	37	2171.96	37	2179.10
S101D1	5	6	746.08	**5**	**742.97**	**5**	**732.46***
S101D2	20	21	1412.98	**20**	1448.28	**20**	1437.42
S101D3	31	31	1924.39	31	1957.34	31	1989.13
S101D5	48	50	2874.86	**48**	3225.70	**48**	3228.09

vehicles by our TS algorithm. As shown in Table 14, 12 out of 25 problem solutions obtain less vehicles by the proposed algorithm, in which 16.67% improved with shorter distances. In the other 13 problems with the same number of vehicles, 53.85% are not worse than those in Chen et al. (2017). Overall, 68% (17 out of 25) instances have solutions from Chen et al. (2017) that have been improved (fewer vehicles or shorter distances) or reproduced the same solutions by our TS algorithm.

6. Conclusion and future research opportunities

The VRPDSPDP is not only of theoretical interest as it has practical applications in distribution. Allowing customers to be visited by more than one vehicle, travel cost savings can be obtained by taking full advantage of the vehicle capacity and reducing the route distances. In this work we describe the problem and establish a corresponding mathematical model. We analyse the features of the VRPDSPDP and present the optimal solution properties of the problem. A tabu search algorithm is proposed, in which two individual operations are designed to avoid unnecessary travel costs, speed up the search and enhance the algorithmic search ability. The computational results indicate that in most of cases the proposed algorithm can be used to find better solutions than those in the literature to the less constrained VRPSPDP.

Several opportunities for future research exist, including exploiting the special structure of the VRPDSPDP, which may lead to further improvements in the performance of the proposed method. Future studies may also include the influences of additional discrete demand restrictions on the solution costs. We believe that it will be worth exploring the new “Item Creation” further to focus on limitations of the proposed formulation due to its different alternative optima, which is capable of reducing the number of alternative optima that may be considered.

Acknowledgements — This research was supported by the National Natural Science Foundation of China (NSFC, 71271220)

Appendix A.

Table A1.1

Discretized batches for delivery for each customer for 20/10/5/1/*x* in experiment 1

Customer	Delivery demand	Number of delivery batches	Batch1	Batch2	Batch3	Batch4	Batch5	Batch6	Batch7	Batch8
1	0.30	7	0.20	0.05	0.01	0.01	0.01	0.01	0.01	

2	0.40	2	0.20	0.20						
3	0.30	7	0.20	0.05	0.01	0.01	0.01	0.01	0.01	
4	0.40	2	0.20	0.20						
5	0.20	1	0.20							
6	0.50	8	0.20	0.20	0.05	0.01	0.01	0.01	0.01	0.01
7	0.30	7	0.20	0.05	0.01	0.01	0.01	0.01	0.01	
8	0.20	1	0.20							
Amount	2.60	35								

Table A1.2

Discretized batches for pickup for each customer for 20/10/5/1/*x* in experiment 1

Customer	Pickup demand	Number of pickup batches	Batch1	Batch2	Batch3	Batch4	Batch5	Batch6	Batch7	Batch8
1	0.30	7	0.20	0.05	0.01	0.01	0.01	0.01	0.01	
2	0.40	2	0.20	0.20						
3	0.40	2	0.20	0.20						
4	0.50	8	0.20	0.20	0.05	0.01	0.01	0.01	0.01	0.01
5	0.20	1	0.20							
6	0.40	2	0.20	0.20						
7	0.30	7	0.20	0.05	0.01	0.01	0.01	0.01	0.01	
8	0.20	1	0.20							
Amount	2.70	30								

Table A2.1

Discretized batches for delivery for each customer for 25/10/5/1/*x* in experiment 1

Customer	Delivery demand	Number of delivery batches	Batch1	Batch2	Batch3	Batch4	Batch5	Batch6
1	0.30	6	0.25	0.01	0.01	0.01	0.01	0.01
2	0.40	3	0.25	0.10	0.05			
3	0.30	6	0.25	0.01	0.01	0.01	0.01	0.01
4	0.40	3	0.25	0.10	0.05			
5	0.20	2	0.10	0.10				
6	0.50	2	0.25	0.25				
7	0.30	6	0.25	0.01	0.01	0.01	0.01	0.01
8	0.20	2	0.10	0.10				
Amount	2.60	30						

Table A2.2

Discretized batches for pickup for each customer for 25/10/5/1/*x* in experiment 1

Customer	Pickup	Number of	Batch1	Batch2	Batch3	Batch4	Batch5	Batch6

	demand	pickup batches						
1	0.30	6	0.25	0.01	0.01	0.01	0.01	0.01
2	0.40	3	0.25	0.10	0.05			
3	0.40	3	0.25	0.10	0.05			
4	0.50	2	0.25	0.25				
5	0.20	2	0.10	0.10				
6	0.40	3	0.25	0.10	0.05			
7	0.30	6	0.25	0.01	0.01	0.01	0.01	0.01
8	0.20	2	0.10	0.10				
Amount	2.70	27						

Appendix B.

Table B

Delivery demand and pickup demand for each customer in experiment 3

Set	No.	Delivery demand	Pickup demand	Set	No.	Delivery demand	Pickup demand
Set 1	1	1	1	Set 2	1	$D_1 = 5$	1
	2	5	5		2	$D_j = D_{j-1} + 1$	5
	3	10	10		3	$\forall j \geq 2$	10
	4	15	15		4	$D_1 = 10$	5
	5	20	20		5	$D_j = D_{j-1} + 1$	10
	6	5	1		6	$\forall j \geq 2$	15
	7	10	1		7	$D_1 = 15$	10
	8	10	5		8	$D_j = D_{j-1} + 1$	15
	9	15	1		9	$\forall j \geq 2$	20
	10	15	5				
	11	15	10	Set 3	1	1	$R_1 = 5$
	12	20	1		2	5	$R_j = R_{j-1} + 1$
	13	20	5		3	10	$\forall j \geq 2$
	14	20	10		4	5	$R_1 = 10$
	15	20	15		5	10	$R_j = R_{j-1} + 1$
	16	1	5		6	15	$\forall j \geq 2$
	17	1	10		7	10	$R_1 = 15$
	18	1	15		8	15	$R_j = R_{j-1} + 1$
	19	1	20		9	20	$\forall j \geq 2$
	20	5	10				
	21	5	15				
	22	5	20				
	23	10	15				
	24	10	20				
	25	15	20				

References

Alfredo Tang Montané, F., Galvão, R. D., 2006. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. Comput. Oper. Res. 33(3), 595-619.

Anily, S., 1996. The vehicle-routing problem with delivery and back-haul options. Nav. Res. Log. (NRL). 43(3), 415-34.

Archetti, C., Bianchessi, N., Speranza, M. G., 2011. A column generation approach for the split delivery vehicle routing problem. Networks. 58(4), 241-54.

Archetti, C., Bianchessi, N., Speranza, M. G., 2014. Branch-and-cut algorithms for the split delivery vehicle routing problem. Eur. J. Oper. Res. 238(3), 685-98.

Archetti, C., Speranza, M. G., 2008. The split delivery vehicle routing problem: A survey, in: Golden, B., Raghavan, R., Wasil, E., (Eds.), The Vehicle Routing Problem: Latest Advances and New Challenges, New York: Springer, pp. 103-22.

Archetti, C., Speranza, M. G., 2012. Vehicle routing problems with split deliveries. Intl. Trans. in Op. Res. 19(1-2), 3-22.

Archetti, C., Savelsbergh, M. W. P., Speranza, M. G., 2006a. Worst-Case Analysis for Split Delivery Vehicle Routing Problems. Transp. Sci. 40(2), 226-34.

Archetti, C., Speranza, M. G., Hertz, A., 2006b. A tabu search algorithm for the split delivery vehicle routing problem. Transp. Sci. 41(1), 64-73.

Archetti C, Savelsbergh, M. W. P., Speranza, M. G., 2008a. To split or not to split: That is the question. Transp. Res. Part E. 44(1), 114-23.

Archetti, C., Speranza, M. G., Savelsbergh, M. W. P., 2008b. An optimization-based heuristic for the split delivery vehicle routing problem. Transp. Sci. 42(1), 22-31.

Avci, M., Topaloglu, S., 2016. A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery. Expert Syst. Appl. 53, 160-71.

Belenguer, J-M. M., Martinez, C., Mota, E., 2000. A lower bound for the split delivery vehicle routing problem. Oper. Res. 48(5), 801-10.

Berbotto, L., Garcia, S., Nogales, F. J., 2014. A Randomized Granular Tabu Search heuristic for the split delivery vehicle routing problem. Ann. of Oper. Res. 222(1), 153-73.

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

Brandão, J., 2006. A new tabu search algorithm for the vehicle routing problem with backhauls. Eur. J. Oper. Res. 173(2), 540-55.

Chen, J. F., Wu, T. H., 2006. Vehicle routing problem with simultaneous deliveries and pickups. J. Oper. Res. Soc. 57(5), 579-87.

Chen, P., Golden, B., Wang, X., Wasil, E., 2017. A novel approach to solve the split delivery vehicle routing problem. Int. T. Oper. Res. 24(1-2), 27-41.

Crispim, J., Brandão, J., 2005. Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls. J. Oper. Res. Soc. 56(11), 1296-302.

Croes, G. A., 1958. A method for solving traveling salesman problems. Oper. Res. 6, 791-812.

Derigs, U., Li, B., Vogel, U., 2010. Local search-based metaheuristics for the split delivery vehicle routing problem J. Oper. Res. Soc. 61(9), 1356-64.

Dror, M., Trudeau, P., 1989. Savings by split delivery routing. Transp. Sci. 23(3), 141-145.

Dror, M., Trudeau, P., 1990. Split Delivery Routing. Nav. Res. Log. (NRL). 37(3), 383-402.

Dueck, G., 1993. New Optimization Heuristics. J. Comput. Phys. 104(1), 86-92.

Fu, Z., Liu. W., Qiu, M., 2017. A tabu search algorithm for the Vehicle Routing Problem with soft time windows and split deliveries by order. Chinese Journal of Management Science. 25(5), 78-86 (in Chinese).

Gendreau, M., Laporte, G. Potvin, Y. J., 2002. Metaheuristics for the capacitated VRP, in: Toth, P., Vigo, D. (Eds.), The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, pp. 129-154.

Glover. F., 1986. Future paths for integer programming and links to artificial intelligence. Comput. Oper. Res. 13(5), 533-549.

Golden, B., Gulczynski, D., Wasil, E., 2008. Recent Developments in Modeling and Solving the Split Delivery Vehicle Routing Problem. Tutorials in Operations Research. 170-80.

Gulczynski, D., Golden, B. Wasil, E., 2010. The Split Delivery Vehicle Routing Problem with Minimum Delivery Amounts. Transp. Res. Part E. 46(5), 612–626.

Han, A., Chu, Y., 2016. A multi-start heuristic approach for the Split-Delivery Vehicle Routing Problem with Minimum Delivery Amounts. Transp. Res. Part E. 88, 11–31.

Ho, S. C., Haugland, D., 2004. A tabu search heuristic for the vehicle routing problem with time windows and split deliveries. Comput. Oper. Res. 31(12), 1947-64.

Hoff, A., Gribkovskaia, I., Laporte, G., Løkketangen, A., 2009. Lasso solution strategies for the vehicle routing problem with pickups and deliveries. Eur. J. Oper. Res. 192(3), 755-66.

Irnich, S., Schneider, M., Vigo, D., 2014. Four variants of the vehicle routing problem, in: Toth, P., Vigo, D. (Eds.), Vehicle Routing: Problems, Methods, and Applications. MOS-SIAM Series on Optimization, Philadelphia, pp. 241–271.

Khmelev, A., Kochetov, Y., 2015. A hybrid VND method for the split delivery vehicle routing problem. Electron Notes Discrete Math. 47, 5-12.

Koç, Ç., Laporte, G., 2018. Vehicle routing with backhauls: Review and research perspectives. Comput. Oper. Res. 91, 79-91.

Lai, M., Battarra, M., Di Francesco, M., Zuddas, P., 2015. An adaptive guidance meta-heuristic for the vehicle routing problem with splits and clustered backhauls. J. Oper. Res. Soc, 66(7), 1222-1235.

Lin, S., 1965. Computer solutions of the traveling salesman problems. Bell Syst. Tech. J. 44, 2245-2269.

Mitra, S., 2005. An algorithm for the generalized vehicle routing problem with backhauling. Asia-Pac. J. Oper. Res. 22(2), 153–169.

Mitra, S., 2008. A parallel clustering technique for the Vehicle Routing Problem with Split Deliveries and Pickups. J. Oper. Res. Soc. 59(11), 1532–1546.

Moreno, L., de Aragão, M. P., Uchoa, E., 2010. Improved lower bounds for the Split Delivery Vehicle Routing Problem. Oper. Res. Lett. 38(4), 302-6.

Mosheiov, G., 1998. Vehicle routing with pick-up and delivery: tour-partitioning heuristics. Comput. Ind. Eng. 34(3), 669-84.

Nagy, G., Salhi, S., 2005. Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. Eur. J. Oper. Res. 162(1), 126-41.

Nagy, G., Wassan, N. A., Salhi, S., 2013. The vehicle routing problem with restricted mixing of deliveries and pickups. J. Sched. 16(2), 199-213.

Nagy, G., Wassan, N. A., Speranza, M. G., Archetti, C., 2015. The Vehicle Routing Problem with Divisible Deliveries and Pickups. Transp. Sci. 49(2), 271–294.

Nakao, Y. Nagamochi, H., 2007. A DP-based heuristic algorithm for the Discrete Split Delivery Vehicle Routing Problem. J. Adv. Mech. Des. Syst. 1(2), 217–226.

Or, I., 1976. Traveling salesman-type combinational problems and their relation to the logistics of blood banking. PhD thesis, Northwestern University, USA.

Osman, I. H, Wassan, N. A., 2002. A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls. J. Sched. 5(4), 263-85.

Ozbaygin, G., Karasan, O., Yaman, H., 2018. New exact solution approaches for the split delivery vehicle routing problem. EURO. J. Comput. Optim. 6(1), 85-115.

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

Polat, O., 2017. A parallel variable neighborhood search for the Vehicle Routing Problem with Divisible Deliveries and Pickups. Comput. Oper. Res. 85, 71–86.

Polat, O., Kalayci, C. B., Kulak, O., Günther, H.-O., 2015. A perturbation based variable neighborhood search heuristic for solving the Vehicle Routing Problem with Simultaneous Pickup and Delivery with Time Limit. Eur. J. Oper. Res. 242(2), 369-382.

Potvin, J. Y., Kervahut, T., Garcia, B. L., Rousseau, J. M., 1992. A tabu search heuristic for the vehicle routing problem with time windows. Technical Report CRT-855, Universite de Montreal, Quebec, Canada.

Salani, M. Vacca, I., 2011. Branch and price for the Vehicle Routing Problem with Discrete Split Deliveries and Time Windows. Eur. J. Oper. Res. 213, 470-477.

Salhi, S., Nagy, G., 1999. A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. J. Oper. Res. Soc. 50(10), 1034-42.

Silva, M. M., Subramanian, A., Ochi, L. S., 2015. An iterated local search heuristic for the split delivery vehicle routing problem. Comput. Oper. Res. 53, 234-49.

Subramanian, A., Drummond, L. M. A., Bentes, C., Ochi, L. S., Farias, R., 2010. A parallel heuristic for the Vehicle Routing Problem with Simultaneous Pickup and Delivery. Comput. Oper. Res. 37(11), 1899-1911.

Wang, K., Ye, C. Ning, A., 2010. Competitive decision algorithm for the Split Vehicle Routing Problem with Simultaneous Pickup and Delivery and Time Windows. In Future Information Technology and Management Engineering (FITME), 2010 International Conference on (Vol. 2, pp. 371-375). IEEE.

Wang, K., Ye, C., Ning, A., 2015. Achieving better solutions for vehicle routing problem involving split deliveries and pickups using a competitive decision algorithm. Asia-Pac. J. Oper. Res. 32(04), 22.

Wang, Y., Ma, X., Lao, Y., Wang, Y., Mao, H., 2013. Vehicle routing problem simultaneous deliveries and pickups with split loads and time windows. Transportation Research Record: Journal of the Transportation Research Board. 2378, 120-128.

Wang, Y., Ma, X., Lao, Y., Yu, H., Liu, Y., 2014. A two-stage heuristic method for vehicle routing problem with split deliveries and pickups. Journal of Zhejiang University SCIENCE C. 15(3), 200-210.

Wassan, N. A., 2007. Reactive tabu adaptive memory programming search for the vehicle routing problem with backhauls. J. Oper. Res. Soc. 58(12), 1630-41.

Wassan, N. A., Wassan, A. H., Nagy, G., 2008. A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries. J. Comb. Optim. 15(4), 368-86.

Wassan, N. A., Nagy, G., 2013. A reactive tabu search meta-heuristic for the Vehicle Routing Problem with Divisible Deliveries and Pickups. Working paper. University of Kent.

Wassan, N., Nagy, G., 2014. Vehicle Routing Problem with Deliveries and Pickups: Modelling Issues and Meta-heuristics Solution Approaches. International Journal of Transportation. 2(1), 95-110.

Wilck IV, J. H., Cavalier, T. M., 2012a. A Construction Heuristic for the Split Delivery Vehicle Routing Problem. Am. J. Oper. Res. 2(02), 153-62.

Wilck IV, J. H., Cavalier, T. M., 2012b. A Genetic Algorithm for the Split Delivery Vehicle Routing Problem. Am. J. Oper. Res. 2(02), 207-16.

Xia, Y., Fu, Z., 2018. A tabu search algorithm for distribution network optimization with discrete split deliveries and soft time windows. Clust. Comput. doi: 10.1007/s10586-018-2635-8.

Yin, C., Bu, L. Gong, H., 2013. Mathematical model and algorithm of Split Load Vehicle Routing Problem with Simultaneous Delivery and Pickup. Int. J. Innov. Comput. I. 9(11), 4497-4508.

Yu, M., Qi, X., 2014. A vehicle routing problem with multiple overlapped batches. Transp. Res. Part E. 61, 40-55.

Zachariadis, E. E., Tarantilis, C. D., Kiranoudis, C. T., 2009. A hybrid metaheuristic algorithm for the vehicle routing problem with simultaneous delivery and pick-up service. Expert Syst. Appl. 36(2), 1070-1081.

Zhang, T., Chaovalitwongse, W. A., Zhang, Y., 2012. Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries. Comput. Oper. Res. 39(10), 2277-2290.