Split Demand One-to-One Pickup and Delivery Problems With the Shortest-Path Transport Along Real-Life Paths

JIAN XIONG$^{1,2}$, XIN QI$^2$, ZHUO FU$^1$, AND WEIXIONG ZHA$^2$

$^1$School of Traffic and Transportation Engineering, Central South University, Changsha 410075, China
$^2$School of Transportation and Logistics, East China Jiao Tong University, Nanchang 330013, China

Corresponding author: Xin Qi (qixin@ecjtu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 71271220, and in part by the Research Project of Humanities and Social Sciences in the Universities of Jiangxi Province under Grant JC18213 and Grant JC19207.

Abstract A variation of the One-to-one Pickup and Delivery Problem (OPDP) in connected graphs, the Split Demand One-to-one Pickup and Delivery Problem with the Shortest-path Transport along Real-life Paths (SDOPDPSTRP) is abstracted from passenger train operation plans based on networks. Unlike the classical OPDP, in the SDOPDPSTRP: the demands can be split and must be transported along the shortest path according to passengers requirements and vehicles should travel along a real-life path. A new kind of integer programming model is formulated for the SDOPDPSTRP based on the connection relationship between pickup-delivery demands (pd-pairs). Two different categories of splitting strategies are proposed to solve the SDOPDPSTRP: split the demands before the calculation and split the demands during the calculation. Two Multi-Start Variable Neighborhood Descent (a MS_VND originating from the other literature and a new MS_VND’ IN developed in this article) and seven neighborhood operators are proposed for these two splitting strategies to solve the SDOPDPSTRP. The results show that Approach III outperforms Approach I and Approach II in terms of average solutions with the same algorithm termination conditions and in terms of time efficiency, which has great practical significance for real-life transport organizations.

Index Terms One-to-one pickup and delivery problem, split demand, shortest-path transport, real-life connected graph, integer programming, multi-start, variable neighborhood descent, Gurobi solver.

I. INTRODUCTION

Travelling along the shortest path, an important requirement of passengers has no always been fully satisfied. As travel modes diversify, it is increasingly important to meet the needs of the passengers to increase the competitiveness of transport enterprises when formulating transportation schemes. Take Passenger Train Operation Plans (PTOP), which are based on lines, for instance, trains travel through real-life paths, and passengers’ demands between every two stations are transported along the shortest path by one or more trains.

Currently, a Chinese high-speed rail network has been formed, it has become an urgent problem to design the PTOP based on networks, which is different from the general PTOP based on lines. Therefore, a new One-to-one Pickup and Delivery Problem (OPDP) is abstracted to solve this new PTOP, which can be formed as follows: There are several pickup-delivery demands (pd-pairs) and trains in a real-life connected graph. The pd-pairs that are chosen must be transported along the shortest path according to passengers’ requirements. Each pd-pair can be split into different trains. Trains cannot visit (stop at or pass through) any station more than once, namely, each train should travel along a real-life path. Constraints, such as train capacity, train travel distances, and train stops, need to be considered.

This new problem can be addressed by introducing a set of maximum-income routes to be traversed by a fleet of vehicles to serve a group of known pd-pairs, which is referred to as the Split Demand One-to-one Pickup and Delivery Problems with the Shortest-path Transport along Real-life Paths (SDOPDPSTRP) in this article. Since each pd-pair must be transported along the shortest path and vehicle stops...
need to be considered, the SDOPDPSTRP will be studied based on connected graphs, which should not be abstracted into complete graphs.

The SDOPDPSTRP, which has rarely been studied in the literature, is studied in this article. The key contributions of this work are as follows:

(i) A new OPDP, the SDOPDPSTRP, is refined from the Passenger Train Operation Plans based on real-life connected graph, and a new model for the SDOPDPSTRP is studied.

(ii) A new Multi-Start Variable Neighborhood Descent (MS_VND’ in this article) with seven neighborhood operators are developed to solve the SDOPDPSTRP based on the second splitting methods in this article: splitting demands during the calculation.

(iii) New instances for the SDOPDPSTRP.

The remainder of this article is organized as follows. Section 2 presents related studies. Section 3 presents the model for the SDOPDPSTRP. Section 4 presents the solution approach. Section 5 presents the computation results. Finally, conclusions and future work are presented in Section 6.

II. RELATED LITERATURES
A. GPDP, OPDP, AND OPDPSTRP
Many scholars have carried out research on the PDP over the past few years. References [1], [2] reviewed current GPDP research and divided studies into two categories. The first category comprises the transportation of goods from a depot to line-haul customers and from back-haul customers to the depot, and this is denoted as the Vehicle Routing Problem with Back-hauls (VRPB). Research on the VRPB was reviewed by [3]. The second category considers all problems that occur when goods are transported between pickup and delivery locations, which is denoted as the General Vehicle Routing Problem with Pickups and Deliveries (GVRPPD). References [4], [5] divided the GPDP into three categories: the One-to-Many-to-One PDP (OMOPDP; [6], [7]), the Many-to-many PDP (MMPDP; [8]–[13]) and the One-to-one PDP (OPDP; [14]–[16]).

Most classical OPDPs are studied using complete graphs, and pickup points must be visited prior to delivery points (e.g. [17]–[23]). The classical OPDP can be easily formulated as a Mixed-Integer Program (MIP), such as those reported in [14]–[16]. Reference [24] classified the solution methods for the Dial-A-Ride Problem (DARP, an important category of the OPDP). Reference [25] proposed a combination of cutting planes to find feasible solutions for the OPDP with incompatibility constraints. Reference [26] studied a new kind of OPDP, the One-to-one Pickup and Delivery Problem with the Shortest-path Transport along Real-life Paths (OPDPSTRP), in which each pd-pair must be transported along the shortest path and each vehicle should travel along real-life paths in connected graphs. A new kind of modeling method was proposed for the OPDPSTRP according to its new route constructions.

Unlike the classical OPDP in a complete graph, the OPDPSTRP studied in this article is described based on connected graphs since the number of vehicle stops needs to be considered. Some research conducted on the route structure of the classical OPDP can provide references for the OPDPSTRP. Reference [27] listed four kinds of ride-sharing patterns for the Ride-sharing Problem. Reference [21] noted out that the cheapest path is not always the quickest path, and a comparison of multiple paths between every two points was necessary. Reference [28] proposed a method for relocating a pd-pair by considering four cases, and the shortest path was chosen as the optimal routing scheme in each local search move. References [29], [30] studied the Ride-sharing Problem (a kind of OPDP) in real-life networks.

Additionally, as in the OPDPSTRP, each vehicle starts at its location (regarded as a depot) and ends at the final delivery point of the contents transported by the vehicle; therefore, it can be considered to be a multi-depot (vehicles) problem. Most OPDP research is based on a single depot, such as that reviewed by [8], [9], [21], [22], [28], [29], [31]–[34]. Some OPDP research is based on multiple depots (vehicles), which is mainly concerned with the Taxi-sharing Problem and Ride-sharing Problem. For example, there is a starting point and an ending point for each vehicle in [35]–[38] while only the starting point is considered for each vehicle in [39].

B. SDVRP AND SDPDP
Since [40] 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. Reference [41] provided a survey on the SDVRP that overviews its variants and, in general, all routing problems that consider split deliveries.

Splitting demands into different vehicles may result in better schedules, so another feature of the SDOPDPSTRP is studied in this article, one kind of Split Demand Pickup and Delivery Problem (SDPDP). There are many categories of the SDPDPs, which can provide some reference for the SDOPDPSTRP. References [42], [43] first proposed the vehicle routing problem with split deliveries and pickups (VRPSPDP). The one-commodity SDPDTSP is discussed by [44], and the OMOPDP with split demands has been discussed by [45]–[47], and [48]. The many-to-many SDPDP is studied by [15], [49], [50]. References [51], [52] proposed a kind of multi-vehicle One-to-one SDPDP.

C. NEIGHBORHOOD AND ALGORITHM
Reference [50] note that when demands are from 51% to 60% of the capacity of the vehicle, up to 30% of the transportation costs can be saved. They find that the PDP with split demand can perform better when the vehicle capacity is approximately twice that of the average demand. Therefore, the keys to solving the SDOPDPSTRP are “splitting or not?” and “how to split?”. Some studies can provide reference to solve this problem.

As for splitting strategies, demands are split when routes are overloaded in [53]. Reference [48] split the demands for a Vehicle Routing Problem into discrete Split Deliveries and
TABLE 1. Differences between the OPDPSTRP in [26] and the SDOPDPSTRP in this article.

| Studies               | Reference [26]                  | This paper                                      |
|-----------------------|---------------------------------|-------------------------------------------------|
| Problems              | OPDPSTRP                        | SDOPDPSTRP: Split Demand OPDPSTRP               |
| Algorithms            | -                               | Insert, Spread, Point-delete, Rout-delete, and Perturbation |
| Instances             | 84 benchmark instances of the OPDPSTRP | 63 benchmark instances of the SDOPDPSTRP         |

Pickups (VRPSPDP) according to the ratios of 25/10/5/1/x and 20/10/5/1/x before the calculation, which are adjusted from [54].

As for neighborhoods, [55] presented eight kinds of local search moves for the OPDP: couple-exchange, block-exchange, relocate-couple, relocate-block, multi-relocate, 2-opt-L, double-bridge and shake. References [56], [57] modified three large neighborhood removal heuristics and two large neighborhood insertion heuristics from [58]–[60] for the OPDP. Additionally, the studies of [24], [61] showed that the solution feasibility of the OPDP is an important issue to the neighborhood efficiency of the algorithm. Reference [53] used four operators (relocation, exchange, 2-opt and split-point reposition) for a Simultaneous Delivery and Pick up Vehicle Routing Problem with Split Loads (SDPVRPSL). Reference [48] proposed five operators for the VRPSPDP: intra-swap, intra-reverse, inter-reassignment, inter-swap and tail swap. Reference [52] proposed 6 intra-route neighborhoods and 4 inter-route neighborhoods in randomized variable neighborhood descent for a SDOPDP.

As for the algorithms, [62] solved the OPDPTSP via the GRASP and the VND. Reference [15] proposed an efficient heuristic that combines the strengths of tabu search and simulated annealing. Tabu search is used in [48], [53]. A VNS is used by [51]. A branch-and-cut algorithm is used by [44], [45]. A branch-and-price approach is proposed by [46], [47]. Reference [52] introduced a hybrid meta-heuristic based on the Iterated Local Search (ILS) and split loads with a new larger dynamic programming-based neighborhoods.

In summation, there is far more research on the classical OPDP and SDPDP than on the SDOPDPSTRP, but there is no research focusing directly on the SDOPDPSTRP proposed in this article. A new kind of OPDP studied by [26], the OPDPSTRP, can provide a reference. This study extends the work of [26] by introducing “split demand” to the OPDPSTRP. The main differences between these two works are listed in TABLE 1.

### III. PROBLEM DEFINITION AND MATHEMATICAL MODEL

#### A. PROBLEM DEFINITION

To define the proposed SDOPDPSTRP in mathematical terms, we specify a connected graph, \( G \equiv (N, E, P, K) \), where \( N = \{1, \ldots, n_0\} \) for vertices, \( E = \{1, \ldots, e_0\} \) for edges, \( P = \{1, \ldots, P\} \) for pd-pairs, and \( K = \{1, \ldots, m\} \) for vehicles. Each pd-pair \( i \) with demand \( q_i \) yields income \( \pi_i \times q_i \). Each vehicle \( K \) has a maximum capacity \( Q_k \) and a fixed cost \( v_c \). The transportation cost per unit length of vehicle \( k \) is \( tc^k \). Each vehicle \( k \) has a stop cost \( S_{kn} \) at node \( n \).

The system also obeys the following assumptions.

(i) Each pd-pair can be split (different from OPDPSTRP in [1]), and must be transported through the shortest path according to passengers’ requirements, with the pickup point being visited prior to the delivery point.

(ii) Each vehicle must travel along a real-life path beginning with the first pickup point and ending at the last delivery point, namely each point cannot be accessed multiple times by one vehicle, a common practice in the Passenger Train Operation Plans and other similar plans.

(iii) For each vehicle, the travel distance limit (from the first pickup point to the last delivery point) is \( D \), and the
maximum number of stops is \( M_0 \). No vehicle can be overloaded.

(iv) The total cost of each vehicle consists of the constant cost, travel cost, and stop cost. To maximize income, not all pd-pairs need to be transported.

(v) There is only one shortest path between any two nodes in the graph.

By defining the aforementioned problem, we hope to identify a suitable scheme to help optimize the benefits.

**B. PARAMETERS AND VARIABLES**

(i) Parameters

- \( q_i \): Demand of pd-pair \( i \).
- \( \pi_i \): Revenue of pd-pair \( i \).
- \( Q^k \): Capacity of vehicle \( k \).
- \( v_c^k \): Fixed cost of vehicle \( k \).
- \( t_c^k \): Transportation cost per unit length of vehicle \( k \).
- \( s_c^k \): Cost of vehicle \( k \) at node \( n \).
- \( l_e \): Length of edge \( e \).
- \( c_i, j \): Judgment parameter for whether pd-pair \( j \) connects to vehicle/pd-pair \( i \), where \( i \in P \) for pd-pairs and \( i = \{ p + 1 \} \) for vehicles.
- \( c_{ai,j} \): Judgment parameter for whether pd-pair \( j \) can connect to vehicle/pd-pair \( i \), where \( i \in P \) for pd-pairs and \( i = \{ p + 1 \} \) for vehicles.
- \( s_d, n \): Judgment parameter for whether pd-pair \( i \) can (or cannot) be picked up/delivered at node \( n \).

All the above parameters can be set as in [26].

(ii) Variables

- \( x^k_{i,j} \): pd-pair \( j \) connects to vehicle/pd-pair \( i \) in vehicle \( k \) or not, where \( i \in P \) for pd-pairs and \( i = \{ p + 1 \} \) for vehicles.
- \( y^e \): Vehicle \( k \) travels by way of edge \( e \) with pd-pairs or not.
- \( q_s^k \): Vehicle \( k \) travels by way of edge \( e \) with pd-pairs or not.
- \( u^k_{i,j} \): Sequence number of pd-pair \( i \) transported by vehicle \( k \), namely \( u^k_{i,j} < u^k_{i,j'} \) when \( x^k_{i,j} = 1 \).
- \( s_n^k \): Vehicle \( k \) stops at node \( n \) or not.

**C. MATHEMATICAL MODEL**

The route structure of the SDOPDPSTRP is actually similar to that of the OPDPSTRP described in [26]. The SDOPDPSTRP can be formulated as an integer programming (IP) model. It should be noted that in this IP model, the decision variables are based on the relationships between pd-pairs, which is totally different from the classical OPDP. In the OPDP mathematical model, the values of the decision variables are based on the relationships between nodes. Take the variable \( x^k_{i,j} \) for instance, it means that node \( j \) come after node \( i \) in the classic OPDP, while it means that PD-pair \( j \) come after PD-pair \( i \) in the OPDPSTRP and the SDOPDPSTRP. For details of the new model methods and the new route construction rules, referring to [26].

1) OBJECTIVE FUNCTIONS

The objective function of the proposed IP model mainly consists of four components, i.e., the total income, the total fixed cost of using vehicle, the total transportation cost and the total stop cost.

(i) Total income

\[
\sum_{k \in K} \sum_{i \in P} \pi_i \cdot q_s^k
\]

(ii) Total fixed cost of using a vehicle

\[
\sum_{k \in K} \sum_{j \in P} v_c^k \cdot x_{p+1,j}^k
\]

(iii) Total transportation cost

\[
\sum_{k \in K} \left( \sum_{e \in E} l_e \cdot y_e^k + \sum_{i \in P \cup \{p + 1\}} \sum_{j \in P} l_{i,j} \cdot x_{i,j}^k \right)
\]

(iv) Total stop cost

\[
\sum_{k \in K} \sum_{n \in N} s_c^k \cdot s_n^k
\]

It is hoped that we can identify a suitable vehicle routing scheme to maximize the benefits:

\[
\sum_{k \in K} \sum_{i \in P} \pi_i \cdot q_s^k - \left( \sum_{k \in K} \sum_{j \in P} v_c^k \cdot x_{p+1,j}^k \right)
\]

\[
+ \sum_{k \in K} \left( \sum_{e \in E} l_e \cdot y_e^k + \sum_{i \in P \cup \{p + 1\}} \sum_{j \in P} l_{i,j} \cdot x_{i,j}^k \right)
\]

\[
+ \sum_{k \in K} \sum_{n \in N} s_c^k \cdot s_n^k
\]

2) CONSTRAINTS

(i) The constraints of determining the order between pd-pairs/vehicle are

\[
x_{i,j}^k \leq c_{i,j} \quad \forall k \in K, \quad i \in P \cup \{p + 1\}, \quad j \in P
\]

\[
x_{i,j}^k \leq \sum_{i \in P \cup \{p + 1\}} c_{a,i,j} \cdot x_{a,i}^k \quad \forall k \in K, \quad i \in P, \quad j \in P
\]

\[
\sum_{i \in P} c_{a,i,j} \cdot x_{i,j}^k \leq 1 \quad \forall k \in K, \quad i \in P \cup \{p + 1\}
\]

\[
x_{i,j}^k = 0 \quad \forall k \in K, \quad i \in P
\]

\[
u^k_i - u^k_j + p \cdot x_{i,j}^k \leq p - 1 \quad \forall k \in K, \quad i, j \in P
\]

\[
\sum_{k \in P \cup \{p + 1\}} x_{i,j}^k \leq 1 \quad \forall k \in K, \quad i, j \in P
\]

(ii) The constraints of the splitting demands are

\[
\sum_{k \in K} q_s^k \leq q_i \quad \forall i \in P
\]

\[
\sum_{j \in P \cup \{p + 1\}} x_{j,i}^k \leq q_s^k \quad \forall k \in K, \quad i \in P
\]
The capacity constraints are
\[
\sum_{i \in P} (ld_{i,e} \cdot q_{i}^{k} \cdot \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k}) \leq Q^{k} \quad \forall k \in K, \ e \in E \quad (13)
\]

The stop constraints are
\[
s_{n}^{k} \geq sod_{i,n} \cdot \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k} \quad \forall k \in K, \ i \in P, \ n \in N
\]
\[
\sum_{n \in N} s_{n}^{k} \leq M_{0} \quad \forall k \in K \quad (14)
\]

The constraints of whether a vehicle is traveling along edge e or not are
\[
ld_{i,e} \cdot \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k} \leq y_{e}^{k} \quad \forall k \in K, \ i \in P, \ e \in E \quad (16)
\]

The constraints of whether a vehicle is assigned vehicle or not are
\[
y_{e}^{k} \leq \sum_{j \in P} x_{j,i}^{k+1,j} \quad \forall k \in K, \ e \in E \quad (17)
\]

The route length constraints are
\[
\sum_{e \in E} le_{e} \cdot y_{e}^{k} + \sum_{i \in K} \sum_{j \in P} lc_{i,j} \cdot x_{j,i}^{k} \leq D \quad \forall k \in K \quad (18)
\]

The domains of the variables are
\[
x_{j,i}^{k} \in \{0, 1\} \quad \forall k \in K, \ i \in P \cup \{p+1\}, \ j \in P \quad (19)
\]
\[
q_{i}^{k} \in \{0, 1, 2, \ldots\} \quad \forall k \in K, \ i \in P \quad (20)
\]
\[
y_{e}^{k} \in \{0, 1\} \quad \forall k \in K, \ e \in E \quad (21)
\]
\[
u_{k}^{i} \in \{1, 2, 3, \ldots\} \quad \forall k \in K, \ i \in P \quad (22)
\]
\[
s_{n}^{k} \in \{0, 1\} \quad \forall k \in K, \ n \in N \quad (23)
\]

3) LINEARIZATION OF THE IP MODEL

The above model is an IP model because the constraint (13) is nonlinear. Reference [67] proposed methods to convert nonlinear formulas into linear formulas. For example, the nonlinear formula \( r = zy \) can be replaced by the linear formulas (24) and (25); where \( r \) is an 0-1 variable and \( M \) is a positive constant with a sufficiently large value.
\[
y - (1 - z) \cdot M \leq r \leq y + (1 - z) \cdot M \quad (24)
\]
\[
-\frac{r}{M} \leq -z \leq M \quad (25)
\]

Formula (13) is nonlinear, and \( \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k} \) is 0/1 variables obviously.

Let:
\[
q_{i}^{k} = q_{i}^{k} \cdot \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k} \quad \forall k \in K, \ i \in P \quad (26)
\]

After replacing the nonlinear constrain (13) by formula (27), (28) and (29) according to the methods mentioned in [67], the integer programming (IP) model for the SDOPDPSTRP is converted into a new integer linear programming (ILP) model that can be solved by the Gurobi solver. \( M \) is a constant with a sufficiently large value.
\[
q_{i}^{k} - (1 - \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k}) \cdot M \leq q_{i}^{k} \leq q_{i}^{k} + (1 - \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k}) \cdot M \quad \forall k \in K, \ i \in P \quad (27)
\]
\[
- (1 - \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k}) \cdot M \leq q_{i}^{k} \leq (1 - \sum_{j \in P \cup \{p+1\}} x_{j,i}^{k}) \cdot M \quad k \in K, \ i \in P \quad (28)
\]
\[
\sum_{k \in K} (ld_{i,e} \cdot q_{i}^{k}) \leq Q^{k} \quad \forall k \in K, \ e \in E \quad (29)
\]

D. A FEASIBLE SOLUTION FOR A SMALL INSTANCE

For a better introduction to the SDOPDPSTRP model, a small instance is given as follows. FIGURE 1 is a connected graph, and the edge lengths are shown in the figure. In a feasible schedule, 5 pd-pairs (demands: 2, 2, 6, 2 and 2) are transported by two vehicles (capacity: 5, maximum distance: 30, and maximum stops: 6) along two routes (paths). Points 1 and 14 are the stop nodes, and points 2, 3, 4, 15, 12, 6, 8 and 10 are the vehicles locations.
TABLE 2. Values of $c_{i,j}$ and $ca_{i,j}$.

|     | $j=1$ | $j=2$ | $j=3$ | $j=4$ | $j=5$ |
|-----|-------|-------|-------|-------|-------|
| $p_{i,j}$ | $-1$  | $0/0$ | $1/0/0$ | $1/0/0$ | $1/0/0$ |
| $q_{i,j}$ | $0/0/0$ | $0/0/0$ | $0/0/0$ | $0/0/0$ | $0/0/0$ |
| $r_{i,j}$ | $1/0/0$ | $1/0/0$ | $1/0/0$ | $1/0/0$ | $1/0/0$ |
| $s_{i,j}$ | $1/0/0$ | $1/0/0$ | $1/0/0$ | $1/0/0$ | $1/0/0$ |

TABLE 3. Decision variables for the schedule.

| Variables | Route 1 | Route 2 |
|-----------|---------|---------|
| $x_{i,j}$ | $x_{i,k}^{1} = 1, x_{i,k}^{2} = 1, x_{i,k}^{3} = 1$ | $x_{i,k}^{4} = 1, x_{i,k}^{5} = 1, x_{i,k}^{6} = 1$ |
| $y_{i,j}$ | $y_{i,j}^{1} = 1, y_{i,j}^{2} = 1, y_{i,j}^{3} = 1, y_{i,j}^{4} = 1, y_{i,j}^{5} = 1, y_{i,j}^{6} = 1$ | $y_{i,j}^{7} = 1, y_{i,j}^{8} = 1, y_{i,j}^{9} = 1, y_{i,j}^{10} = 1, y_{i,j}^{11} = 1$ |
| $s_{i,j}$ | $s_{i,j}^{1} = 1, s_{i,j}^{2} = 1, s_{i,j}^{3} = 1, s_{i,j}^{4} = 1, s_{i,j}^{5} = 1, s_{i,j}^{6} = 1$ | $s_{i,j}^{7} = 1, s_{i,j}^{8} = 1, s_{i,j}^{9} = 1, s_{i,j}^{10} = 1, s_{i,j}^{11} = 1$ |
| $q_{i,j}$ | $q_{i,j}^{1} = 2, q_{i,j}^{2} = 2, q_{i,j}^{3} = 5$ | $q_{i,j}^{4} = 2, q_{i,j}^{5} = 2, q_{i,j}^{6} = 1$ |
| $t_{i,j}$ | $t_{i,j}^{1} < t_{i,j}^{2} < t_{i,j}^{3} < t_{i,j}^{4}$ | $t_{i,j}^{5} < t_{i,j}^{6} < t_{i,j}^{7} < t_{i,j}^{8}$ |

\[
\begin{align*}
\text{FIGURE 2.} & \quad \text{Construction methods for Split.} \\
\text{FIGURE 3.} & \quad \text{Construction methods for Insert.}
\end{align*}
\]

IV. SOLUTION APPROACH

Two Multi-Start Variable Neighborhood Descent (MS_VND and MS_VND') and seven neighborhood operators are proposed to solve the SDOPDPSTRP based on two categories of strategies: splitting demands before the calculation and splitting demands during the calculation.

A. NEIGHBORHOODS

Reference [53] used four operators (relocation, exchange, 2-opt and split-point re-positioning) for a Simultaneous Delivery and Pick up Vehicle Routing Problem with Split Loads (SDPVRPSL). Reference [48] proposed five operators for the VRPSPDP: intra-swap, intra-reverse, inter-reassignment, inter-swap and tail swap for the VRPSPDP. Reference [52] proposed 6 intra-route neighborhoods and 4 inter-route neighborhoods in randomized variable neighborhood descent for a SDOPDP.

The solution for a small instance was provided in the last part of Section III, FIGURE 1 shows the route structure of the solution, TABLE 3 shows the variables. Since the route structure of the SDOPDPSTRP is quite different from that of the classical OPDP, there are not as many pd-pairs that can be reinserted into a new route in the SDOPDPSTRP as in the classical OPDP. Therefore seven neighborhoods, which are modified from [26], are presented for the SDOPDPSTRP.

1) SPLIT

As in FIGURE 2, pd-pair $i$ is selected randomly and inserted into a new route $j$ chosen according to the route structure feasibility strategy studied in [26]. If route $j_2$ is overloaded after being inserted, split pd-pair $i$ and leave the overload part in route $j_1$.

2) INSERT

As in FIGURE 3, pd-pair $i$ is selected randomly and inserted into a new route $j_2$ chosen according to the route structure feasibility strategy.

IV. SOLUTION APPROACH

Two Multi-Start Variable Neighborhood Descent (MS_VND and MS_VND') and seven neighborhood operators are proposed to solve the SDOPDPSTRP based on two categories of strategies: splitting demands before the calculation and splitting demands during the calculation.

A. NEIGHBORHOODS

Reference [53] used four operators (relocation, exchange, 2-opt and split-point re-positioning) for a Simultaneous Delivery and Pick up Vehicle Routing Problem with Split Loads (SDPVRPSL). Reference [48] proposed five operators for the VRPSPDP: intra-swap, intra-reverse, inter-reassignment, inter-swap and tail swap for the VRPSPDP. Reference [52] proposed 6 intra-route neighborhoods and 4 inter-route neighborhoods in randomized variable neighborhood descent for a SDOPDP.

The solution for a small instance was provided in the last part of Section III, FIGURE 1 shows the route structure of the solution, TABLE 3 shows the variables. Since the route structure of the SDOPDPSTRP is quite different from that of the classical OPDP, there are not as many pd-pairs that can be reinserted into a new route in the SDOPDPSTRP as in the classical OPDP. Therefore seven neighborhoods, which are modified from [26], are presented for the SDOPDPSTRP.

1) SPLIT

As in FIGURE 2, pd-pair $i$ is selected randomly and inserted into a new route $j_2$ chosen according to the route structure feasibility strategy.
3) SWAP
In the Swap, route $i$ is chosen randomly and pd-pairs are inserted into the other routes from route $i$ until the number of “insert” are more than the number of iterations $K$ or the solution is improved or there is no pd-pair on route $i$. Then, we choose pd-pairs according to the route structure feasibility strategy, and insert them into route $i$ until the number of “insert” are more than the numbers of iterations $K$ or the solution is improved.

4) SPREAD
As in FIGURE 4, a pd-pair is selected and inserted into a new route $j_2$ as an Insert operation. If the vehicle is overloaded, the success rate can be improved by choosing a new pd-pair $i$ from the route $j_2$ and transferring $i$ into a new route $j_3$ selected according to the route structure feasibility strategy. This cycle will continue until the vehicle is no longer overloaded, or if the number cyclic $k$ exceeds the preset number of iterations $K$.

5) POINT-DELETE
As in FIGURE 5, Point-delete starts by choosing a route at random. Then, the point with the minimal number of picking stops and delivery stops on the route is isolated, and these pd-pairs $i \in P$ are subsequently inserted into different routes selected according to the route structure feasibility strategy, thus making it possible to delete the point from the first route.

6) ROUTE-DELETE
In Route-delete, the net income $ne_i$ of each route $i$ is computed. Route $k$ with the max negative income is selected. All pd-pairs transported by route $k$ are changed to the state of being non-carried.

7) PERTURBATION
The key of the Perturbation is Reassign-vehicle, which is an Assignment Problem (AP). In Reassign-vehicle, vehicles are reassigned to routes to achieve the best scheme by the Gurobi solver in Matlab.

Since Reassign-vehicle may make the most significant change to the solution but cannot always improve the solution and requires more CPU time, it is better to choose to perturb the local best solution according to a low probability. Therefore, a kind of Perturbation is proposed to shock the local best solution instead of conducting Reassign-vehicle. In Perturbation, Split, Insert, Swap, Spread, Point-delete and Reassign-vehicle are chosen according to the operator choosing probabilities $p_1, p_2, p_3, p_4, p_5$ and $p_6$ respectively.

B. APPROACHES WITH DIFFERENT DEMAND SPLITTING STRATEGIES
As in Table 4, 3 kinds of approaches with two different categories of splitting strategies are proposed to solve the SDOPDPSTRP in this article: (i) demands are split before the calculation, and then, the new problems is solved as the OPDPSTRP; (ii) demands are split during the calculation.

| Approaches          | Splitting strategies       | Pre-splitting methods | Algorithms         |
|---------------------|---------------------------|-----------------------|--------------------|
| Approach I          | Split demands before the calculation | 20/10/5/1/x          | MS YND in [24]     |
| Approach II         | Split demands during the calculation | 25/10/5/1/x          |                   |
| Approach III        | Split demands before the calculation | -                    | MS YND in this paper |

1) SPLIT DEMANDS BEFORE THE CALCULATION
(i) Pre-treatment of the demands
In the strategy that splits demands before the calculation, the demands are split according to the ratios of 20/10/5/1/x and 25/10/5/1/x before the calculation as in [48], [54].

20/10/5/1/x Splitting Strategy: Each demand is split into to 5 separate groups, each of which has a different quantity. The first four demands are set as $0.2Q, 0.1Q, 0.05Q$ and $0.01Q$. The fifth demand is set as the load of the quantity less than 0.01$Q$. For example, if $Q = 300$ and $q = 205$, then we split $q$ into $q_1 = 60, q_2 = 60, q_3 = 60, q_4 = 15, q_5 = 3, q_6 = 3, q_7 = 3$ and $q_8 = 1$.****
25/10/5/1/x Splitting Strategy: Each demand is split into to 5 separate groups, each of which has a different quantity. The first four demands are set as $0.25Q$, $0.1Q$, $0.05Q$ and $0.01Q$. The fifth demand is set as the load of the quantity less than $0.01Q$. For example, if $Q = 300$ and $q = 205$, then we split $q$ into $q_1 = 75$, $q_2 = 75$, $q_3 = 30$, $q_4 = 15$, $q_5 = 3$, $q_6 = 3$, $q_7 = 3$ and $q_8 = 1$.

Then, the SDOPDPSTRP is converted to the OPDPSTRP and can be solved by a Multi-Start Variable Neighborhood Descent (MS_VND), which was proposed in [26] and shown as in Algorithm 2.

(ii) Generation steps of the initial solution
Generation steps of the initial solution are presented in Algorithm 1.

(iii) Algorithm
The steps of the MS_VND are presented in Algorithm 2, which were mentioned in [26].

To improve the search, the evaluation value of the algorithm is set as:

$$s = (z - z_1 \times M) \times (M/1000) + (z_0 - z_1 \times M) \quad (31)$$
Algorithm 1 Pseudo-Code of Generation Steps of Initial Solution

1. **Input:** let \( P = \{1, \ldots, p\} \) is set of pd-pairs set; let routes set \( R = \{R_i | R_i = pd\_pair \ i\} (i \in P) \) without vehicles; Input Multi-start candidate solution set size \( n \); let \( i = 0 \); 
2. Reassign vehicles to routes by neighborhood Reassign-vehicle and solution \( s^0 \) is obtained; 
3. let \( S = \{s_i = s^0, i = 1, \ldots, n\} \) be Multi-Start candidate solution set; 
4. for \( i = 1: n \) 
5. select pd\_pair \( j \) and route \( R_k \) randomly in solution \( s^0 \); try to insert pd-pair \( j \) into route \( R_k \); 
6. if succeed 
7. \( s_i \leftarrow s^0 \); Update Multi-Start candidate solution set \( S \); 
8. end if 
9. end for

FIGURE 10. Splitting Pd-pair and Avg Solution.

FIGURE 11. Average efficiency of the 6 operators for parameter \( p_k \).

V. INSTANCES AND COMPUTATIONAL RESULTS

A. GENERATION OF INSTANCES

Sixty three instances are provided to test the methods for the SDOPDPSTRP, which are generated as follows.

Each instance name has the format \( m0 \times m1-m2-m3-m4-m5-m6-m7 \), where \( m0 \times m1 \) is the size of a connected graph, the distance between any two node is randomly set as \([0.5, 1.5]\), \( 1/m2 \) is the probability that each edge in this graph is deleted, \( 1/m3 \) is the pd-pair generation probability between every two nodes, \( 1/m4 \) is the vehicle generation percentage for each node, \( m5 \) is ratio of the income/cost, the capacity is set as \( m6 \) times the average demands, and the vehicles are randomly chosen as \( 1/m7 \) times the demands. Additionally, [50] noted that when demands are from 51% to 60% of the capacity of the vehicle, up to 30% of the transportation costs can be saved. Therefore, \( m6 \) is set as \( 2 \sim 3 \), namely, the capacity is set as \( 2 \sim 3 \) times of the average demands in this article.

Consider the instance \( 3 \times 4.10-10-2-1-2-2 \) as an example. The size of the incomplete digraph is \( 3 \times 4 \) (12 nodes and 144 node-pairs), the distance between any two node is set as \([0.5, 1.5]\), the deletion probability of each edge is 1/10, the pd-pair generation probability between every two nodes is 1/10, the vehicle generation percentage for each node is 1/2, the ratio of income/cost is set as 1, the capacity is set as twice the average demands, and the number of vehicles is 1/2 the demand. It has been checked that there is only one shortest path between any two nodes in each graph. For each vehicle, the maximum travel distance limit is set as \( D = (m0+1) \times 2 \), and the maximum number of stops is \( M = (m0+1) \times 2 \).
**Algorithm 2 Pseudo-Code of the MS_VND Meta-Heuristic**

1. **Input:** let bestsofar_s=max[s_i], let Insert, Spread, Point-delete, Rout-delete and Perturbation be operators opt(k), (k = 1, 2, 3, 4 and 5); input K for Spread, Selection controlling value T0 and replacing proportion m, input p1, p2, p3, p4 and p5 for Perturbation, Multi-start candidate solution set size n, algorithm termination iterations constant_T, total_iteration; Input Multi-Start candidate solution set S, let constant=0, constant0=0, iteration=0.

2. while constant<constant_T or iteration<total_iteration

   for every S_i ∈ S

   if constant0 < T0

      k = 1;

   else if T0 ≤ constant0 < T0 * 2

      k = 2;

   else if T0 * 2 ≤ constant0 < T0 * 3

      k = 3;

   else if T0 * 3 ≤ constant0 < T0 * 4

      k = 4;

   else if T0 * 4 ≤ constant0 < T0 * 5

      k = 5;

   else

      k = 6;(Perturbation)

      if Reassign-vehicle is chosen in Perturbation

         constant 0=0;

      end if

   end if

   s′_i ← opt(k,s_i);

   if s′_i is not inferior than s_i

      let s_i = s′_i; update S;

   else

      if constant>constant_T/2 and f(s′_i)-f(s_i) ≥ f(s_i)/2

         let s_i = s′_i according to the probability 50%; update S;

      end if

   end if

end for

find the local best solution localbest_s in S; iteration=iteration+1;

if localbest_s is better than bestsofar_s

   let bestsofar_s=localbest_s; constant=0;

else

   constant=constant+1;constant 0=constant 0+1;

end if

if constant<constant_T/2

   replace the worst m solutions in S with bestsofar_s;

end if

end while

The demand of each pd-pair is randomly set as [10, 20], namely, the average demand q is set as 15.

All the data of the 63 instances can be found in APPENDIX A.

**B. COMPUTING ENVIRONMENT**

All experiments were conducted on a desktop equipped with an Intel(R) Core(TM) i7-4510U 2.00 GHz processor and 8 GB of RAM. The operating system of this PC was 64-bit Windows 8. The new integer linear programming (ILP) model was solved using the Gurobi solver 7.5.2, which was embedded into Matlab R2015a by the Yalmip toolbox. All the algorithms in this article were also programmed in Matlab.

**C. PARAMETER SETTING**

The ILP model is solved by the Gurobi solver with the termination conditions set for a computing time over 1000 seconds or the gap is less than 5%. The long preset time aims to ensure that the Gurobi solver can obtain at least one feasible solution served as a comparison indicator with the proposed approaches, although in some cases it failed to
Algorithm 3 Pseudo-Code of the MS_VND’ Meta-Heuristic

1. **Input:** let bestsofar_s=max{s_i}; let Split, Insert, Swap, Spread, Point-delete, Rout-delete and Perturbation be operators opt(k), (k = 1, 2, 3, 4, 5, 6 and 7); input K for Spread, Selection controlling value T0 and replacing proportion m, input p1, p2, p3, p4, p5, p7 and p8 for Perturbation, Multi-start candidate solution set size n, algorithm termination iterations constant_T, total_iteration; Input Multi-Start candidate solution set S, let constant=0, constant0=0, iteration=0.

2. **while** constant<constant_T or iteration<total_iteration **do**
   3. **for** every s_i in S **do**
      4. if constant0 < T0 **then**
         5. k = 1;
      6. **else if** T0 ≤ constant0 < T0 * 2 **then**
         7. k = 2;
      8. **else if** T0 * 2 ≤ constant0 < T0 * 3 **then**
         9. k = 3;
      10. **else if** T0 * 3 ≤ constant0 < T0 * 4 **then**
         11. k = 4;
      12. **else if** T0 * 4 ≤ constant0 < T0 * 5 **then**
         13. k = 5;
      14. **else if** T0 * 5 ≤ constant0 < T0 * 6 **then**
         15. k = 6;
      16. **else**
         17. k = 7; (Perturbation)
         18. if Reassign-vehicle is chosen in Perturbation **then**
         19. constant0=0;
      20. **end if**
      21. **end if**
      22. s_i’ ← opt(k,s_i);
      23. if s_i’ is not inferior than s_i **then**
         24. let s_i = s_i’; update S;
      25. **else**
         26. if constant>constant_T/2 and f(s_i’)−f(s_i) ≥ −f(s_i)/2 **then**
         27. let s_i = s_i’ according to the probability 50%; update S;
         28. **end if**
      29. **end if**
   30. **end for**
   31. find the local best solution localbest_s in S; iteration=iteration+1;
   32. if localbest_s is better than bestsofar_s **then**
      33. let bestsofar_s=localbest_s; constant=0;
   34. **else**
      35. constant=constant+1; constant0=constant0+1;
   36. **end if**
   37. if constant<=constant_T/2 **then**
      38. replace the worst m solutions in S with bestsofar_s;
   39. **end if**
   40. **end while**

achieve this goal. We provided the upper bounds found by the Gurobi solver as well a more in-depth reference to evaluate the performances of the proposed approaches: Approach I, Approach II and Approach III.

Almost all of numbers of the pd-pairs of the above 63 instances are not more than 230 after being pre-treated. Therefore, the parameters values (n, constant_T, total_iteration, T0, K and m) of MS_VND can be set the same as in [26], where the numbers of pd-pairs are not more than 236. Then, the parameters values (n, constant_T, total_iteration, T0, K and m) of MS_VND’ are set the same as for MS_VND to better compare the efficiency of these two splitting strategies. Furthermore, the operator sequence opt(k) and the operator choosing probability p_k in MS_VND’ are reanalyzed in APPENDIX B because two new operators, Split and Swap, are added to MS_VND’.
TABLE 5. Parameter setting for MS\_VND and MS\_VND'.

| Symbol       | Definition                                                                 | Value                        |
|--------------|-----------------------------------------------------------------------------|------------------------------|
| n            | Size of the Multi-Start candidate solution                                  | 90                           |
| constant_T   | Algorithm termination condition 1: limit on the number of iterations         | \(\text{constant}_T = \exp(-20(2-\text{num\_pd\_pairs})^700, \text{where num\_pd\_pairs is the number of pd-pairs})\) |
| total\_iteration | Algorithm termination condition 2: limit on the total number of iterations | \(\text{num\_pd\_pairs}^700, \text{where num\_pd\_pairs is the number of pd-pairs}\) |
| T_0          | Selected controlling value                                                  | 20                           |
| K            | Number of iterations for Spread                                            | 3                            |
| opt(k)       | Operators sequences                                                        | MS\_VND: Insert, Spread, Point-delete, Rout-delete, and Perturbation        |
|              |                                                                             | MS\_VND': Split, Insert, Swap, Spread, Point-delete, Rout-delete, and Perturbation |
|              |                                                                             | 9/24, 7/24, 1/24, 1/24, 6/24 for Insert, Spread, Point-delete, Rout-delete, and Reassign-vehicle, respectively |
| p_n          | Operator choosing probabilities in Perturbation                            | 6/35, 7/35, 10/35, 43/55, 1/35, 1/35, 6/35 for Split, Insert, Swap, Spread, Point-delete, Rout-delete, and Reassign-vehicle, respectively |
| m            | Replacement proportion for Multi-Start solution set                         | 1/8                          |

The parameters values of MS\_VND and MS\_VND' are given in TABLE 5.

TABLE 6. Abbreviation of the experiment indicators and definitions.

| Abbreviation | Definition |
|--------------|------------|
| UB           | The upper bound of the ILP model obtained by the Gurobi solver in a preset running time. |
| LB           | The best feasible objective value found by the Gurobi solver in a preset running time. |
| Gap          | The gap between UB and LB: (UB-LB)/UB. |
| Gap\_Avg     | The average feasible objective value obtained by Approach I, Approach II and Approach III after a preset number of iterations. |
| Gap\_Best    | The best feasible objective value obtained by Approach I, Approach II and Approach III after a preset number of iterations. |
| Time         | Average CPU time for solving the ILP model by Approach I, Approach II and Approach III (second). |
| Initial\_pd\_pairs | Number of initial pd-pairs of each instance. |
| Spitting\_pd\_pairs | Number of pd-pairs in each instance of the solution. |

TABLE 7. Computational results for small size graphs.

| Instances | Initial pd-pairs | Volumes | Approach I \((\text{split demands during calculation})\) | Approach II \((\text{split demands during calculation})\) | Approach III \((\text{split demands during calculation})\) |
|-----------|------------------|---------|--------------------------------------------------------|--------------------------------------------------------|--------------------------------------------------------|
|           |                  |         | UB          | Gap          | Time (seconds) | UB          | Gap          | Time (seconds) | UB          | Gap          | Time (seconds) |
| 3,4,10,11,12 | 2, 6, 12         | 1, 3, 7 | 2513     | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 |
| 3,4,10,11,12 | 2, 6, 12         | 1, 3, 7 | 2513     | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 |
| 3,4,10,11,12 | 2, 6, 12         | 1, 3, 7 | 2513     | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 |
| 3,4,10,11,12 | 2, 6, 12         | 1, 3, 7 | 2513     | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 |
| 3,4,10,11,12 | 2, 6, 12         | 1, 3, 7 | 2513     | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 | 21.718      | 19.3535      | 1, 3, 7, 21, 3, 7 |

The test results of Gurobi solver, Approach I, Approach II and Approach III are shown in TABLE 7 (small size graphs), TABLE 8 (medium size graphs) and TABLE 9 (large size graphs). Each instance has been solved 10 times by each method.

According to the results, the following can be found:

1) Approach III can always obtain better solutions than Approach I and Approach II with the same algorithm termination conditions. FIGURE 6 and FIGURE 7
show the gaps between the LB and LB_Avg and LB_Best solved by Approach I, Approach III and Approach III. The Gap_Avg and Gap_Best of the Approach III are the lowest.

2) Approach III outperforms Approach I and Approach II in terms of time efficiency. As shown in FIGURE 8, the average time efficiency (Average Solution/Time) of Approach III is better than that of Approach I and Approach II.

3) There is a strong link between the number of splitting pd-pairs and the time for solving the SDOPDPSTRP. As in FIGURE 9, Approach I and Approach II take more time than that of Approach III to solve the SDOPDPSTRP with the same algorithm termination conditions, because they split more pd-pairs than Approach III.

(iv) In theory, the fewer the demands that are split, the better the solutions that can be obtained. However, this phenomenon is not immutable. As shown in FIGURE 10, the average number of splitting pd-pairs is 45 in Approach III, but Approach III outperforms Approach I and Approach II in terms of solution quality with more than 500 on average. Namely, “how to split” is worthy of further study in the future.

VI. CONCLUSION

A new Pickup and Delivery Problem with a new route structure and split demand, the Split Demand One-to-one

VOLUME 8, 2020

TABLE 8. Computational results for medium size graphs.

| Instances | Initial pd-pairs | Vehicles | Graphs | Approach I | Approach II | Approach III |
|-----------|-----------------|----------|--------|------------|-------------|-------------|
|           | U                | L        | B      | GB         | GB         | GB         |
|           | [pre-split demand according to 25/09/16] |           |        |            |             |             |
| 64-10 20k | 7                | 3566      | 4600   | 1.000     | 0.999      | 0.887      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |

TABLE 9. Computational results for large size graphs.

| Instances | Initial pd-pairs | Vehicles | Graphs | Approach I | Approach II | Approach III |
|-----------|-----------------|----------|--------|------------|-------------|-------------|
|           | U                | L        | B      | GB         | GB         | GB         |
|           | [pre-split demand according to 25/09/16] |           |        |            |             |             |
| 64-10 20k | 7                | 3566      | 4600   | 1.000     | 0.999      | 0.887      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |
| 64-10 20k | 14               | 3566      | 5551   | 1.000     | 1.000      | 0.999      |

TABLE 10. Parameters setting of OPT(K) and P_K for the MS_VND.

| Symbol | Definition | Value
|--------|-----------|------|
| opt(K) | Operators sequences | MS_VND: Split, Insert, Swap, Spread, Point-deliver, Read-deliver, and Perturbation
| P_K   | Operator choosing probabilities in Perturbation | MS_VND: 0.35, 0.35, 0.35, 0.35, 0.35, 0.35

Note: The indexes in this table are introduced in TABLE VI.
Pickup and Delivery Problems with the Shortest-path Transport along Real-life Paths (SDOPDPSTRP), which is proposed using connected graphs, is introduced and formulated in a new way. Three methods are proposed to solve the SDOPDPSTRP: two methods (Approach I and Approach II) split demands before the calculation, and the other method (Approach III) splits demands during the calculation.

The results show that Approach III outperform Approach I and Approach II in terms of the average solutions and time efficiency under the same algorithm termination conditions, which has great practical significance for real-life transport organizations. It is also found that “how to split” is the key to solving the SDOPDPSTRP and that the time it takes to solve the SDOPDPSTRP is closely related to the number of split pd-pairs, especially for large instances. Therefore, the splitting methods for large instances are worthy of further study in the future.

**APPENDIX**

**A. INSTANCES**

The relative data for the instances can be found online at the following link: https://www.researchgate.net/publication/343415212_Instances_of_the_SDOPDPSTRP.

**B. PARAMETERS SETTING OF \opt(k) AND \(p_k\) FOR THE \(MD_{VND}'\)**

The parameter setting, such as the operator sequence \(\opt(k)\) and the operator choosing probability \(p_j\) in Perturbation will be revised in this section because two new operators, Split and Swap, are proposed in the \(MD_{VND}'\).

These two parameters have also been tested over 9 instances (including small size connected graphs, medium size connected graphs and large size connected graphs, chosen from the 64 instances in APPENDIX A), as in the parameter setting is tuned by determining the trade-off between the solution quality and CPU time after numerous experiments.

FIGURE 11 shows the performance of the six operators (Split, Insert, Swap, Spread, Point-delete and Route-delete) which are chosen separately in the \(MD_{VND}'\) (the other operators are removed).

According to FIGURE 11, the operator choosing ratios between neighborhoods is set as \(6:7:10:4:1:1\) (according to the average improvement efficiency) in the Perturbation for \(MD_{VND}'\).

FIGURE 12 shows the performance of the seven operators in the \(MD_{VND}'\) algorithm with the empirical values given in TABLE 10. The results show that Reassign-vehicle will take a large amount of CPU time. Therefore, the sequence of choosing the operators is determined as Split, Insert, Swap, Spread, Point-delete, Route-delete and Perturbation for the \(MS_{VND}'\).

For a better comparison, the final parameters are shown in the TABLE 10.

**REFERENCES**

[1] S. Parragh, K. Doerner, and R. Hartl, “A survey on pickup and delivery problems I,” J. Für Betriebswirtschaft, vol. 58, pp. 21–51, Oct. 2008.

[2] S. N. Parragh, K. F. Doerner, and R. F. Hartl, “A survey on pickup and delivery problems: Part II: Transportation between pickup and delivery locations,” J. Für Betriebswirtschaft, vol. 58, no. 2, pp. 81–117, Jun. 2008.

[3] Ç. Koç and G. Laporte, “Vehicle routing with backhauls: Review and research perspectives,” Comput. Oper. Res., vol. 91, pp. 79–91, Mar. 2018.

[4] G. Berbeglia, J. Cordeau, I. Gribkovskaya, and G. Laporte, “Static pickup and delivery problems a classification scheme and survey,” Top, vol. 15, pp. 1–31, Apr. 2007.

[5] G. Berbeglia, J.-F. Cordeau, and G. Laporte, “Dynamic pickup and delivery problems,” Eur. J. Oper. Res., vol. 202, no. 1, pp. 8–15, Apr. 2010.

[6] I. Gribkovskaya and G. Laporte, “One-to-many-to-one vehicle pickup and delivery problems,” Oper. Res. Comput. Sci. Interface, vol. 43, pp. 359–377, 2008.

[7] Z. Zhu, J. Xiao, S. He, Z. Ji, and Y. Sun, “A multi-objective memetic algorithm based on locality-sensitive hashing for one-to-many-to-one dynamic pickup-and-delivery problem,” Inf. Sci., vol. 329, pp. 73–89, Feb. 2016.

[8] H. N. Psaraftis, “An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows,” Transp. Sci., vol. 17, no. 3, pp. 351–357, Aug. 1983.

[9] H. N. Psaraftis, “Analysis of an O(N^2) heuristic for the single vehicle many-to-many Euclidean dial-a-ride problem,” Transp. Res. B, Methodol., vol. 17, no. 2, pp. 133–145, Apr. 1983.

[10] T. R. Sexton and L. D. Bodin, “Optimizing single vehicle many-to-many operations with desired delivery times: I. Scheduling,” Transp. Sci., vol. 19, no. 4, pp. 378–410, Nov. 1985.

[11] T. R. Sexton and L. D. Bodin, “Optimizing single vehicle many-to-many operations with desired delivery times: II. Routing,” Transp. Sci., vol. 19, no. 4, pp. 411–435, Nov. 1985.

[12] R. Camargo, G. Miranda, and A. Løkketangen, “A new formulation and an exact approach for the many-to-many hub location-routing problem,” Appl. Math. Model., vol. 37, pp. 7465–7480, May 2013.

[13] J. Rieck, C. Ehrenberg, and J. Zimmermann, “Many-to-many location-routing with inter-hub transport and multi-commodity pickup-and-delivery,” Eur. J. Oper. Res., vol. 236, no. 3, pp. 863–878, Aug. 2014.

[14] H. Hernández-Pérez and J.-J. Salazar-González, “The multi-commodity one-to-one pickup-and-delivery traveling salesman problem,” Eur. J. Oper. Res., vol. 196, no. 3, pp. 987–995, Aug. 2009.

[15] M. Āhīn, G. Čašušić, G. Čašušić, and D. T. Aksu, “An efficient heuristic for the multi-vehicle One-to-one pickup and delivery problem with split loads,” Transp. Res. C, Emerg. Technol., vol. 27, pp. 169–188, Feb. 2013.

[16] M. Soysal, M. Çimen, and E. Demir, “On the mathematical modeling of green one-to-one pickup and delivery problem with split load,” J. Cleaner Prod., vol. 174, pp. 1664–1678, Feb. 2018.

[17] B. Fleischmann, “A cutting plane procedure for the traveling salesman problem on a road network,” Eur. J. Oper. Res., vol. 21, pp. 307–317, Apr. 1985.

[18] G. Cornejo, J. Fonlupt, and D. Naddef, “The traveling salesman problem on a graph and some related integer polyhedra,” Math. Program., vol. 89, pp. 1–27, Sep. 1991.

[19] H. Longo, M. P. de Aragəo, and E. Uchoa, “Solving capacitated arc routing problems using a transformation to the CVRP,” Comput. Oper. Res., vol. 33, no. 6, pp. 1823–1837, Jun. 2006.

[20] E. Demir and T. Van Woensel, “Mathematical modeling of CO2e emissions in one-to-one pickup and delivery problems,” in Proc. IEEE Int. Conf. Ind. Eng. Eng. Manage., Dec. 2013, Art. no. 14762859.

[21] A. N. Letchford, S. D. Nasiri, and A. Oukil, “Pricing routines for vehicle routing with time windows on road networks,” Comput. Oper. Res., vol. 51, pp. 331–337, Nov. 2014.

[22] X. Li, J. Ma, J. Cui, A. Ghiasi, and F. Zhou, “Design framework of large-scale one-way electric vehicle sharing systems: A continuum approximation model,” Transp. Res. B, Methodol., vol. 88, pp. 21–45, Jun. 2016.

[23] M. Mahmoudi and X. Zhou, “Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state–space–time network representations,” Transp. Res. B, Methodol., vol. 89, pp. 19–42, Jul. 2016.

[24] S. C. Ho, W. Y. Szeto, Y.-H. Kuo, J. M. Y. Leung, M. Petering, and T. W. H. Tou, “A survey of dial-a-ride problems: Literature review and recent developments,” Transp. Res. B, Methodol., vol. 111, pp. 395–421, May 2018.
XIN QI received the B.S. degree in transportation management from Central South University, in 2004, and the M.S. degree in transportation management from East China Jiao Tong University, in 2009. He is currently pursuing the Ph.D. degree in transportation planning and management with Central South University. He is also an Associate Professor with East China Jiao Tong University. His research interests include railway transportation organization and vehicle routing problem.

ZHUO FU received the B.S., M.S., and Ph.D. degrees in transportation management from Central South University, in 1982, 1985, and 2004, respectively. He is currently a Professor with Central South University. His research interests include railway transportation organization and vehicle routing problem.

WEIXIONG ZHA received the B.S. degree in mathematics from Wuhan University, in 1983, the M.S. degree in mathematics from Central South University, in 1987, and the Ph.D. degree in transportation management from Beijing Jiaotong University, in 1998. He is currently a Professor with East China Jiao Tong University. His research interests include railway transportation operation management optimization and transportation economy.