Proceedings of the Federated Conference on Computer Science and Information Systems pp. 307–315
DOI: 10.15439/2020F127
ISSN 2300-5963 ACSIS, Vol. 21

Stochastic multi-depot vehicle routing problem with pickup and delivery: an ILS approach

Brenner H. O. Rios, Eduardo C. Xavier, Flávio K. Miyazawa
Institute of Computing
University of Campinas
São Paulo, Brazil
Email: brenner@students.ic.unicamp.br, {ecx,fkm}@ic.unicamp.br

Pedro Amorim
INESC TEC
Faculty of Engineering
University of Porto
Porto, Portugal
Email: amorim.pedro@fe.up.pt

***Abstract*—We present a natural probabilistic variation of the multi-depot vehicle routing problem with pickup and delivery (MDVRPPD). In this paper, we present a variation of this deterministic problem, where each pair of pickup and delivery points are present with some probability, and their realization are only known after the routes are computed. We denote this stochastic version by S-MDVRPPD. One route for each depot must be computed satisfying precedence constraints, where each pickup point must appear before its delivery pair in the route. The objective is to find a solution with minimum expected traveling distance. We present a closed-form expression to compute the expected length of an *a priori* route under general probabilistic assumptions. To solve the S-MDVRPPD we propose an Iterated Local Search (ILS) that uses the Variable Neighborhood Descent (VND) as local search procedure. The proposed heuristic was compared with a Tabu Search (TS) algorithm based on a previous work. We evaluate the performance of these heuristics on a data set adapted from TSPLIB instances. The results show that the ILS proposed is efficient and effective to solve S-MDVRPPD.**

I. Introduction

VEHICLE routing problems (VRPs) have been extensively studied over the last three decades, mainly due to their economic importance and their theoretical challenges. The diversity of applications has motivated the study of several variants of VRPs. One of its more challenging variants is the multi-depot vehicle routing problem (MDVRP), where the well know TSP is a particular case of this problem. On the other hand, uncertainty is a characteristic of many real VRPs. Some common stochastic elements are customer requests, travel time and service time. The stochastic VRP (SVRP) is basically any VRP where one or more parameters are stochastic.

The VRP variant we consider is the stochastic version of MDVRP with pickup and delivery (MDVRPPD). The MDVRPPD is closely related to the problem proposed in [1]. In this work, the authors introduced the multi-depots pick-up and delivery problem with time windows and multi-vehicles. The principle of MDVRPPD is to design an optimal set of routes for a fleet of vehicles, each one located in a different depot. The set of routes allows serving a set of pickup and delivery points geographically dispersed. The number of vehicles is equal to the number of depots. Each vehicle must start and end the route in its assigned depot. Vehicles must visit once and only once each node. In Figure 1 we show the tour of three vehicles belonging each one to a different depot.

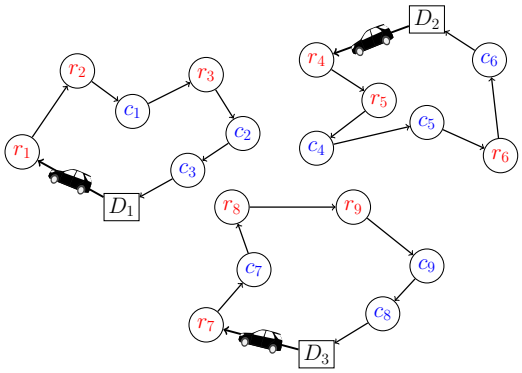

Fig. 1. Example of a solution for the MDVRPPD with three vehicles. Each vertex r_i represents a pickup point while c_i represents its corresponding delivery point.

In the stochastic version of the MDVRPPD the pickup and delivery points are uncertain. Consider for example an online marketplace provider, which is an e-commerce platform owned and operated by the provider, where third-parties can sell their products. It is common for the provider to be responsible for the gather and delivery of sold products (specially in the case of food delivery for example). If, based on past data, the provider has access to a probability distribution of the chance of a request from costumer A from seller B to occur, better routes can be constructed.

We propose an Iterated Local Search (ILS) heuristic to solve this problem that uses the Variable Neighborhood Descent (VND) heuristic as a local search. We denote the proposed algorithm by ILS-VND. The ILS-VND is based on the ILS heuristics presented by Subramanian et. al. [2] for the VRP with simultaneous pickup and delivery. To evaluate its performance we compare it with an adaptation of the TABUSTOCH algorithm proposed by Gendreau et. al. [3]. The tabu search algorithm is one of the main methods to deal with SVRPs [4].

The remainder of this paper is organized as follows. Section II-A presents the description and mathematical formulation of the MDVRPPD. Section III presents the closed-form expression to compute the expected length of an *a priori*

route. Section IV introduces the proposed heuristic ILS-VND. Section V presents the adapted tabu search heuristic. Section VI describes the computational experiments, and Section VII presents the conclusions of this work.

A. Related work

The proposed problem has a close connection with the well known multi-depot traveling salesman problem (mTSP) [5]. Specifically, with the special case of multi-depot multiple travelling salesman problem (MmTSP). In the MmTSP each salesman starts from a unique city, travels to a set of cities and completes the route by returning to his original city with each city visited once [6]. Kara and Bektas [7] presents a mTSP review and explores connections with VRPs. The MDVRPPD is also closely related to the steiner multi cycle problem (SMCP), recently introduced in [8]. The SMCP arises in the scenario where a company has to periodically exchange goods between two different locations, and different companies can collaborate to create a route that visits all its pairs of locations sharing the total cost of the route [8]. The MDVRPPD can be seen as a version of the SMCP with depots. There are several heuristic approaches to solving VRPs and its stochastic variant. State of the art solutions include: particle swarm optimization approach [1], VNS [9], adaptive large neighbourhood search [10], ant colony optimization [11], genetic approach [12], tabu search [13], simulated annealing [14] and hybrid heuristic with exact methods [15] [16]. A review of the solution methods used in the past 20 years for the SVRP is presented in [17].

II. Problem description and model

In this section, we present the MDVRPPD and model it as an integer linear program first, then we define the S-MDVRPPD.

A. MDVRPPD

In this work, MDVRPPD is defined as follows. Let $G = (V, E)$ be a complete undirected graph, where $V = \{v_1, \ldots, v_n\}$ is the vertex set and $E = \{(v_i, v_j) : v_i, v_j \in V, i < j\}$ is the edge set. With each edge (v_i, v_j), it is associated a non-negative cost or distance d_{ij}. A subset of vertices $D = \{v_1, \ldots, v_m\}$ represents the depots, and the remaining vertices $V' = \{v_{m+1}, \ldots, v_n\}$ corresponds to pickup and delivery points. Let $w = |V'|/2$, then w vertices are pickup points and w vertices are delivery points. Each pickup point v_i is associated with a unique delivery point v_{i+w}, and vice versa, for $m + 1 \leq i \leq m + w$. There are m identical vehicles of unlimited capacity such that each one is located in a single depot. Each vehicle leaves its depot, serves a subset of pickup and delivery vertices and returns to its depot, forming a cycle (or route). The problem consists in determining a set of m vehicle cycles of minimal total cost considering the following constraints: a) each cycle starts and ends at the corresponding vehicles depot; b) each $v \in V'$ is visited exactly once by one vehicle c) each pair of pickup and delivery points, e.g $\{v_i, v_{i+w}\}$ for $m + 1 \leq i \leq m + w$, must belong to the same cycle and d) each cycle has an orientation where each pickup vertex in this cycle appears before its delivery pair.

The MDVRPPD is NP-hard since it includes the Traveling Salesman Problem (TSP) as a special case (e.g. if each pair of pickup and delivery are in the same location and there is only one depot).

We adapt the mathematical formulation proposed in [5] for the deterministic static version of MDVRPPD. In this formulation we assume G is a complete symmetric directed graph. The parameters and variables of the formulation are defined in Table I.

TABLE I
Parameters and decision variables for the MDVRPPD

D	Set of depots, $\{v_1, ..., v_m\}$.
V'	Set of nodes (pickup and delivery), $\{v_{m+1}, ..., v_n\}$
H^+	Set of pickup nodes, $\|H^+\| = w$.
u_i^k	A positive integer variable that indicates the order vertex i is visited by vehicle k, and $u_i^k = 0$ if i is not visited by k, $i \in V'$, $k \in D$.
d_{ij}	Distance between vertices i and j.
x_{ij}^k	If the vehicle from depot k travel along arc (i, j), then $x_{ij}^k = 1$, otherwise $x_{ij}^k = 0$.

$$\text{minimize} \sum_{k\in D}\sum_{j\in V'}(d_{kj}x_{kj}^k + d_{jk}x_{jk}^k) + \sum_{k\in D}\sum_{i\in V'}\sum_{j\in V'} d_{ij}x_{ij}^k \tag{1}$$

$$s.t. \quad \sum_{j\in V'} x_{kj}^k = 1, \quad k \in D, \tag{2}$$

$$\sum_{j\in V'} x_{jk}^k = 1, \quad k \in D, \tag{3}$$

$$\sum_{k\in D} x_{kj}^k + \sum_{k\in D}\sum_{i\in V'} x_{ij}^k = 1, \quad \forall j \in V', \tag{4}$$

$$x_{kj}^k + \sum_{i\in V'} x_{ij}^k = x_{jk}^k + \sum_{i\in V'} x_{ji}^k, \quad \forall k \in D, j \in V', \tag{5}$$

$$u_i^k \leq n(\sum_{j\in V'} x_{ij}^k + x_{ik}^k), \quad i \in V', \quad k \in D, \tag{6}$$

$$x_{ki}^k \leq u_i^k, \quad i \in V', \quad k \in D, \tag{7}$$

$$u_i^k + 1 \leq u_j^k + (1 - x_{ij}^k)n, \quad i, j \in V', \quad k \in D, \tag{8}$$

$$u_i^k + 1 \leq u_{i+w}^k + (1 - \sum_{j\in V'} x_{ij}^k)n, \quad i \in H^+, \quad k \in D, \tag{9}$$

$$x_{ij}^k \in \{0, 1\}, \quad i, j \in V, \tag{10}$$

$$u_i^k \in \mathcal{Z}^+, \quad i \in V, \quad k \in D, \tag{11}$$

In this formulation, constraint (2) ensures that exactly one vehicle depart from each depot $k \in D$, while (3) assures the vehicle returns to the depot. Constraint (4) ensures that each node is visited exactly once. Route continuity is ensured by the flow conservation constraints (5). Constraints (6) assures that the order of client i is 0 if it is not in the route of vehicle k. Constraints (7) impose that if i is the first vertex visited in route k, than its order in this route is at least 1. Constraint (8) is a subtour elimination constraint, since if j is visited after i in route k, then the visit order of j must be larger than the one of i in this route. Constraint (9), ensures that each pick-up node (i) must be visited before the corresponding delivery node ($i + w$). Finally we have the integrality constraints (10) and (11) of the variables in the model.

B. S-MDVRPPD

Now, we define the particular S-MDVRPPD considered in this work. This problem has one type of uncertainty: stochastic pickup and delivery points. Each pair $\{v_i, v_{i+w}\} \in V'$, for $m + 1 \leq i \leq m + w$, has a probability p_i of being present when traveling along the route. When pickup point v_i is absent, delivery point v_{i+w} is also absent. We consider the S-MDVRPPD as a two stage stochastic problem. In the *first stage*, a set of cycles satisfying constraints (a) - (d) of the MDVRPPD are computed. The presence or absence of $\{v_i, v_{i+w}\}$ is revealed at the latest time upon leaving the preceding vertex of v_i. We suppose that the demand of every delivery point v_{i+w} is the same e.g. one unit. In the *second stage*, the first stage routes are followed as planned, with the following exception: any absent node is skipped. The S-MDVRPPD consists of designing a first stage solution that *minimizes the expected cost of the second stage solution*.

The S-MDVRPPD can be formulated as a stochastic integer program. We will use the parameters and variables defined in Table I. Let $T(x, \xi)$ be the cost of second stage solution if $x = (x_{ij}^k)$ is the first stage solution, and $\xi = (\xi_i)$ is the vector of non-negative random variables associated with the vertices of V'. The S-MDVRPPD is then formulated as

$$\min_x E_\xi[T(x, \xi)] \tag{12}$$

subject to equations (2)-(11).

III. The expected cost of an *a priori* route

Given a priori computed route $s = (v_0, v_1, \ldots, v_{2q}, v_0)$, where v_0 is a depot, let l_s be the cost/length of s. Our goal is to compute efficiently the expected length $E[l_s]$ of route s, given that during its execution, each pair $\{v_i, v_{i+w}\}$ of pickup and delivery points in this route have a probability of occurring during s's execution. We may also refer to node v_i as r_i, and v_{i+w} as c_i. Let $P(v_i)$ be the probability that node v_i appears in s. Note that we have the following relationship for a pair of pickup and delivery points r_i and c_i: $P(r_i) = P(c_i)$, $P(c_i|r_i \text{ appears}) = P(r_i|c_i \text{ appears}) = 1$, and $P(c_i|r_i \text{ not appear}) = P(r_i|c_i \text{ not appear}) = 0$.

In this theorem we assume that R is the set of pickup points that appear in s ($|R| = q$). The pickup points are numbered in the superscript, in the order they appear in s, from r^1 until r^q. Likewise, C is the set of the corresponding delivery points and are also numbered in the superscript, from c^1 until c^q in the order they appear in s. We also use the following notation. If v_i is a pickup point we denote this by writing $r(v_i)$, and its corresponding delivery point as v_i^-, and if v_i is a delivery node, we denote this by writing $c(v_i)$ and denote its corresponding pickup point as v_i^+. Finally, given a subsequence $s_i^j = (v_i, v_{i+1}, \ldots, v_j)$ of s, let $R(s_i^j)$ denote the set containing the pickup points that appear in s_i^j, and also containing the pickup points of the delivery vertices that appear in s_i^j. Notice that $|R(s_i^j)| \leq j - i + 1$, and it is strictly small only when a pickup point appears in s_i^j and its delivery point also appears. Then we can compute $E[l_s]$ as follows.

Theorem 1. *Given a priori route $s = (v_0, v_1, \ldots, v_{2q}, v_0)$, then:*

$$\begin{aligned} E[l_s] = &\sum_{i=1}^{q} d_{v_0,r^i} P(r^i) \prod_{k=1}^{i-1} (1 - P(r^k)) \\ &+ \sum_{i=1}^{q} d_{c^i,v_0} P(c^i) \prod_{k=i+1}^{q} (1 - P(r^k)) \\ &+ \sum_{i=1}^{2q} \sum_{j=i+1}^{2q} f(v_i, v_j) \end{aligned} \tag{13}$$

where

$$f(v_i, v_j) = \begin{cases} 0 & \text{, (a) or (b)} \\ d_{v_i,v_j} P(v_i) \prod_{v \in R(s_{i+1}^{j-1})} (1 - P(v)) & \text{, (c)} \\ d_{v_i,v_j} P(v_i) P(v_j) \prod_{v \in R(s_{i+1}^{j-1})} (1 - P(v)) & \text{, (d)} \end{cases} \tag{14}$$

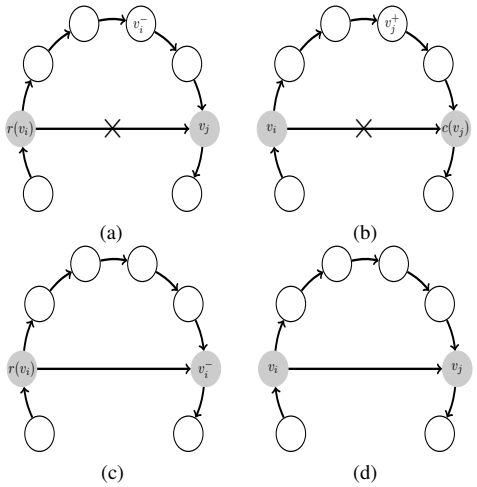

Fig. 2. In (a) v_i is a pickup node and v_j appears after v_i's corresponding delivery node v_i^-. In (b) v_i is any vertex, v_j is a delivery node and v_j^+ appears after v_i. Both situations, (a) and (b) do not occur, since in (a), if $r(v_i)$ is present in the route then v_i^- must appear as well, and in (b) if $c(v_j)$ is present then v_j^+ must be present as well, so going from v_i directly to v_j skipping vertices in between is not a valid route. In (c) we have the case where v_i is a pickup point and v_j is its corresponding delivery point. In (d) we have all other cases that do not belong to one of the previous cases.

Proof: In equation (13) we are basically computing the probability of each edge between vertices in s to appear, in the execution of s, and multiplying this probability by the edge's cost. We have three terms in this equation.

In the first term, we are computing the expected cost of each possible initial edge of the route, such that if the route starts with (v_0, r^i) then all previous pickup points r^k, $k = 1, \ldots, i-1$ must not be present in the route. In the second term, we are computing the expected cost of each possible final edge in the route, similar to the first term.

In the last term we compute the expected cost of each edge from any pair of vertices in the route. Figure 2 represents all the possible cases between vertices v_i and v_j. Since cases (a) or (b) do not occur in practice the expected cost of an edge (v_i, v_j) in any one of these cases is zero. In case (c), the probability of going directly from $r(v_i)$ to $v_j = v_i^-$ is the probability of request of pickup point v_i to occur times the probability of requests of points in between v_i and v_j to not occur. Likewise, in case (d), if v_i and v_j are not related in any of the previous cases, then the probability of edge (v_i, v_j) to occur, is equal to the probability of v_i and v_j to occur times the probability of none of the requests of vertices in between them to occur. ■

We can compute the expected cost of an *a priori* route, $E(l_s)$, with time complexity $O(q^2)$, where $2q+2$ is the size of the route.

IV. ILS-VND

The proposed heuristic (ILS-VND) for the S-MDVRPPD works as follows. The method is executed $MaxIter$ times. In each iteration an initial solution is generated by a greedy algorithm, then this solution is improved using ILS. Internally, the ILS procedure uses the VND heuristic for performing the local search and a refinement heuristic for the initial solution called Random Mix-Shift. The ILS-VND is presented in Algorithm 1, where s^* corresponds to the best solution found during any iteration.

Algorithm 1: ILS-VND

```
for k := 1, ..., MaxIter do
    s := GenerateInitialSolution(seed);
    s' := RandomMixShift(s) ;
    iterILS := 0;
    while iterILS < MaxIterILS do
        r = number of neighborhoods ;
        s := VND(N(.), r, s) ;
        if f(s) < f(s') then
            s' := s ;
            s := Perturb(s') ;
            iterILS := 0 ;
        else
            iterILS := iterILS + 1 ;
    if f(s') < f(s*) then
        s* := s' ;
```

A. Initial solution generation

The method employed for building a feasible initial solution is based in the work of [18], so it is generated by following two steps. The first step is called *nodes assignment*. Each pair of pickup and delivery is assignment to one of the depots. After all vertices have been assigned to depots, the second step, called *nodes sequencing*, decides the service sequence of the pickup and delivery nodes. The details of these two steps are introduced in the following paragraphs.

1) Nodes assignment: This step assigns nodes to the depot which is closer to them. In the same way as in [18], in order to make the initial solution more flexible, the assignments of pickup an delivery nodes to depots are based on a probability. Suppose that $d(D_a, r_i, c_i)$ indicates the sum of the distances between pickup point r_i and depot D_a, and the distance between the delivery point c_i and depot D_a. Let $\overline{d(D, r_i, c_i)}$ the average distance between the pair of pickup and delivery $\{r_i, c_i\}$ and all depots. The probability of the set $\{r_i, c_i\}$ being assigned to depot D_a, is calculated by Eq. (15).

$$P(D_a, r_i, c_i) = \frac{\max\left\{\overline{d(D, r_i, c_i)} - d(D_a, r_i, c_i), 0\right\}}{\sum_{a=1}^{|D|} \max\left\{\overline{d(D, r_i, c_i)} - d(D_a, r_i, c_i), 0\right\}} \tag{15}$$

2) Nodes sequencing: Once we have assigned all pickup and delivery nodes to depots, we sequence the nodes to create cycles. Let D_i be an arbitrary depot, v_i a pickup or delivery point associated with D_i and t_i a tour containing D_i. The procedure begins by looking for a node v_i nearest to the last node of t_i, which initially contains only D_i, such that all constraints of the S-MDVRPPD are satisfied when v_i is appended to t_i. Then v_i is appended to t_i and the procedure is repeated until all the pickup and delivery nodes are inserted into t_i. The Figure 3 shows an example of sequencing one depot and four pairs of pickup and delivery points. The complexity time of this procedure is $O(n^2)$.

B. Local Search

The local search is based on the VND heuristic introduced by Mladenović and Hansen [19]. In the variable neighborhood descent method a change of neighborhoods is performed in a deterministic way. The proposed VND is presented in Algorithm 2.

A set $\{N^1, \ldots, N^6\}$ of six neighborhood operators were used by the proposed VND. All operators are exhaustively executed. These operators are adapted in such a way that they preserve feasibility. We divide these operators into three groups: inter-tour, intra-tour and inter&intra-tour operators. The inter-tour operators are: Shift(1,0) and Swap(1,1). The intra-tour neighborhood operators are: 2-opt, 3-opt and Reverse. Finally, Mix-shift(1,0) is the only operator that is inter&intra-tour operator. In the case of inter-route operators, to reduce the computational cost, each vertex removed of a

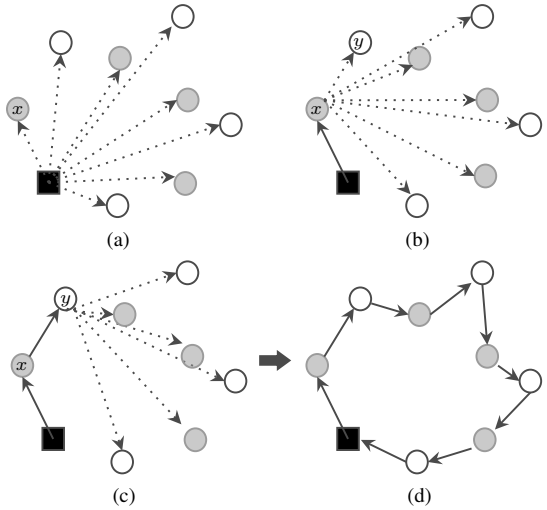

Fig. 3. Example of nodes sequencing in a route. Gray nodes are pickups and white nodes deliveries. In (a) distances from the depot to all vertices are calculated. In (b) node x is added to the route, since it is the closest to the depot and its addition does not break the constraints of the problem. Then, the distances from x to all the nodes that are not part of the route are computed. The node y is the closest to x, and its addition does not break the constraints of the problem. In (c) node y is added to the route. The process repeats until all nodes are added to the route. The obtained route is shown in (d).

route can only be inserted before or after one of its p closest neighbors in the other routes.

The list of neighborhoods considered are:

- **Shift(1,0)** – N^1 – A pickup and delivery pair r, c is removed from a route t_1 and each one is moved to the best position in route t_2 keeping the feasibility of the solution.
- **Swap(1,1)** – N^2 – An exchange between a pair r_1, v_1 from a route t_1 and another pair r_2, v_2 from route t_2. Each vertex of the pairs are inserted in the best possible position while maintaining the feasibility of the solution.
- **Mix-Shift(1,0)** – N^3 – This operator is similar to the Shift(1,0) operator with the difference that now it is allowed a movement within its own route.
- **2-opt** – N^4 – Two nonadjacent arcs are removed and other two are added to form a new route. We only consider movements that do not break the constraints of the problem.
- **3-opt** – N^5 – Three nonadjacent arcs are removed and other two are added to form a new route. We only consider movements that do not break the constraints of the problem.
- **Reverse** – N^6 – This operator reverses the direction of the route. Then swaps are performed between each pair of pickup and delivery.

In case of improvement of the current solution, the algorithm performs an intensification process on each route. The objective is to decrease the cost of each route. Therefore, the neighborhoods N^4, N^5 and N^6 are applied in this order in the current solution.

Given the versatility of the *Mix-Shift(1,0)* operator, and based on the neighborhood structure proposed by Gendreau et. al. [3], we present the Random Mix-Shift heuristic (Algorithm 3). In this heuristic a randomly selected pickup and delivery pair $\{r, c\}$ is removed, r is inserted immediately before or after one of its p closest neighbors. The vertex c is randomly inserted into the same route, without breaking the constraints of the problem. To avoid necessary iterations of Random Mix-Shift heuristic, we chose to use s' if it is *promising*. A solution s' is *promising* if it has the potential to become the new best solution, specifically, if its cost is at most $\alpha\%$ higher than the cost of the best solution so far, s. We use this heuristic in the ILS-VND as a refinement mechanism to improve the initial solution.

C. Perturbation Mechanism

A set P of two perturbation mechanisms were adopted in the ILS-VND heuristic. Every time the *perturb()* function is called one of the following operators is randomly selected and applied.

Algorithm 2: VND

```
Let r be the number of neighborhoods
 structures and s a current solution ;
k := 1; current neighborhood ;
while k ≤ r do
    Find the best neighbor s' of s ∈ N^k ;
    if f(s') < f(s) then
        s := s' ;
        k := 1 ;
        intensification in the modified
         routes ;
        s' := 2 − opt(s) ;
        s'' := 3 − opt(s') ;
        s''' := Reverse(s''') ;
        if f(s''') ≤ f(s) then
            s := s'''
    else k := k + 1 ;
```

Algorithm 3: Random Mix-Shift

```
for k := 1 ..., MaxIterShift do
    r, c := SelectRandomPair(r, c) ;
    s' := Mix-Shift(s, r, c) ;
    if f(s') < f(s*) then
        s := s'; s* := s'; k := 1 ;
    else
        if f(s') < αf(s*) then
            s := s' ;
        else
            s := s*;
```

Double-Swap – P^1 – Two Swap(1,1) operators are performed in sequence.

Depot Exchange – P^2 – The depot exchange operator select two depots at random, and exchange their routes.

V. Tabu search

We present an adaptation of the TABUSTOCH heuristic proposed by Gendreau et. at. [3] that was originally designed to the Vehicle Routing Problem (VRP) with stochastic demands. The algorithm solves a two stage stochastic VRP, where in the first stage a feasible solution is constructed including all vertices (clients). In the second stage recourse actions maybe taken, since the real demands of costumers are realized, capacity constraints may become violated. In traversing a route, once a vehicle becomes full it returns to the depot and resumes the route in the next client to be visited. All the parameters in the adapted algorithm, are the same used in the original TABUSTOCH. We will only present the modifications made to TABUSTOCH in order to deal with the S-MDVRPPD. Let x^k be a solution in the first stage in iteration k of the algorithm. Let $T(x^k)$ be the expected value in the second stage. Let $T(x^k) = \sum_{i=1}^{m^k} T^i(x^k)$, where m^k is the number of routes at iteration k.

The initial solution is built by assigning to each depot the closest *candidate* pair of pickup and delivery. A pair of pickup and delivery is a *candidate* if it has not been assigned to some depot. The selected pickup and delivery pair is appended to the solution (first pickup and then delivery). The initial solution is always feasible. The neighbourhood structure used by TABUSTOCH is the Mix-shift(1,0) operator presented in section IV-B. Thus, there is the possibility of inserting pairs of pickup and delivery in different cycles.

We consider the movement of nodes in a solution as elements of the tabu list. There are two ways to move a vertex: 1) change the position of the vertex in the same tour and 2) move the vertex (and its corresponding pair) to another tour. Either of these two movements is tabu for θ iterations, where θ is randomly selected from the interval $[|V|-5, |V|]$. The search of solutions in each iteration considers the current solution x^k and the best non-tabu solution x^{k+1} in the neighborhood structure Mix-Shift(1,0). However, a tabu solution can be selected if it improves the best solution T^* (aspiration criteria).

Note that computing the expected value of a solution is expensive. Moving a pickup and delivery pair not only affects the cost related to their immediate neighbors, but also affects the cost of each node in the tour. Notice also that the movement affects the costs of both previous and new tours where they where inserted.

Suppose we wish to insert a pair of pickup and delivery $\{x^+, x^-\}$ into a route. Figures (4a) and (4b) represents the possible positions of $\{x^+, x^-\}$ in a route before removing them while Figures (4c) and (4d) represents all possible positions of x^+ and x^- after their insertion into the new route. Dotted arrows represent arcs before insertion. Red lines represent subtours and black arrows arcs. We denote the approximations of the effect of inserting a pickup and delivery in a route with A_i and $\bar{A}_i$. A_i refers to the insertion of $\{x^+, x^-\}$ as shown in the Figure (4c). $\bar{A}_i$ refers to the insertion of $\{x^+, x^-\}$ as shown in Figure (4d). Note that the approximations of the effect of removing a pickup and delivery in a route can be represented with $-A_i$ and $-\bar{A}_i$.

We use three easy-computational approximations of insertion cost to speed up the search process. The first approximation, given by equations (16) and (17), completely detach the stochastic nature of the problem. The second approximation, given by equations (18) and (19), partially remediate the first approximation, but these equations give all the weight for e, f, g, h and x^-. Taking into account P_e, P_f, P_g, P_h and P_{x^-}, the third approximation, given by equations (20) and (21), seeks to remedy the second approximation. The problem with this last approximation happens when P_e, P_f, P_g, P_h are small and P_{x^+} (and so P_{x^-}) is large.

Tests conducted on 600 randomly generated instances involving between 10 and 100 vertices indicates that the third approximation yields the best correlation with the true cost increase ($r = 0.89$).

$$A_1(e,f,g,h,x^+,x^-) = d_{ex^+} + d_{x^+f} + d_{gx^-} + d_{x^-h} - d_{ef} - d_{gh} \quad (16)$$

$$\bar{A}_1(e,f,x^+,x^-) = d_{ex^+} + d_{x^+x^-} + d_{x^-f} - d_{ef} \quad (17)$$

$$A_2(e,f,g,h,x^+,x-) = (d_{ex^+} + d_{x^+f} + d_{gx^-} + d_{x^-h} - d_{ef} - d_{gh})P_{x^+} \quad (18)$$

$$\bar{A}_2(e,f,x^+,x^-) = (d_{ex^+} + d_{x^+x^-} + d_{x^-f} - d_{ef})P_{x^+} \quad (19)$$

$$A_3(e,f,g,h,x^+,x^-) = d_{ex^+}P_eP_f + d_{x^+f}P_{x^+}P_f + d_{gx^-}P_gP_{x^-} + d_{x^-h}P_{x^-}P_h - d_{ef}P_eP_f - d_{gh}P_gP_h \quad (20)$$

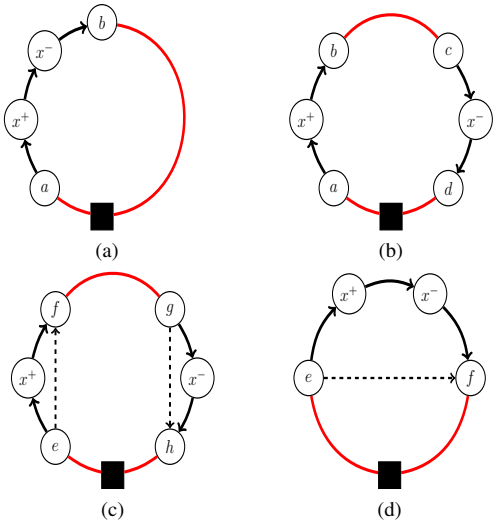

Fig. 4. Example of movements of nodes x^+ and x^-. Cases (a) and (b) represent different situations of nodes x^+ and x^- in a route, before the movement. Cases (c) and (d) represent possible insertion of x^+ and x^- in a route, after the movement.

$$\bar{A}_3(e,f,x^+,x^-) = d_{ex^+}P_eP_{x^+} + d_{x^+x^-}P_{x^+}P_{x^-} + d_{x^-f}P_{x^-}P_f - d_{ef}P_eP_f \quad (21)$$

We have the necessary terms to *approximate the cost of a movement*. The expressions (22)-(25) are used to evaluate the movement cost of x^+ and x^-. We will use the cases shown in Figure 3. If the movement of x^+ and x^- happens in the order: from case (4a) to case (4c) we use the equation (22); from case (4a) to case (4d) we use equation (23); from case (4b) to case (4c) we use (24); and, from case (4b) to case (4d) we use (25).

$$\Delta_1 = A_3(e,f,g,h,x^+,x^-) - \bar{A}_3(a,b,x^+,x^-) \quad (22)$$

$$\bar{\Delta}_1 = \bar{A}_3(e,f,x^+,x^-) - \bar{A}_3(a,b,x^+,x^-) \quad (23)$$

$$\Delta_2 = A_3(e,f,g,h,x^+,x^-) - A_3(a,b,c,d,x^+,x^-) \quad (24)$$

$$\bar{\Delta}_2 = \bar{A}_3(e,f,x^+,x^-) - A_3(a,b,c,d,x^+,x^-) \quad (25)$$

VI. Computational experiments

We conducted experiments using a data set derived from six TSPLIB instances (*ulysses16, bayg29, dantzig42, eil51, st70* and *st76*). For each of these instances, n vertices in the interval of [2, 10] were randomly selected to be depots. A random matching was performed among the other vertices to create pickup and delivery pairs. The probability of presence of each pickup and delivery pair was chosen uniformly in the interval $[0, 1]$. We generate 30 test instances.

The algorithms described above were coded in C++ and all experiments were run on a Linux operating system with 3 GB memory and Intel Core i5 2.54x4 Ghz processor. Computational times reported here are in CPU seconds on this machine. To evaluate our ILS-VND heuristic we compare it with an adaptation of TABUSTOCH algorithm. Ten independent runs of the algorithms were performed for each test case. The number of iterations (*MaxIter*) and perturbation allowed (*MaxIterILS*), was 10 and 15 respectively. The parameters α, $MaxIterShift$ and p were fixed to 1.05, 100 and 5, respectively. They were calibrated empirically after preliminary tests with different values.

Table II shows the results obtained by the ILS-VNS and the TABUSTOCH heuristics. The best solutions are in boldface. The columns related with the instances show: the instance name, *Name*, the number of vertices in the graph, $|V|$, and the number of depots $|D|$. Columns *Avg.* and *Best* show the average and the best solution costs found by the algorithms in their ten independent executions, respectively. Column *Time* presents the average processing time, in seconds, spend by each algorithm. The ILS-VND presented the best results for all instances.

Table III shows the percentage improvement of the best and average solutions obtained by the ILS-VND, against the TABUSTOCH. Negative values mean that ILS-VND was better than the TABUSTOCH. The formula $\frac{\text{ILS-VND}-\text{TABUSTOCH}}{\text{ILS-VND}}$ was used to generate the values presented in Table III. The results show that the ILS-VND was superior to the TABUSTOCH for all instances tested.

TABLE II
Results of the ILS-VND and TABUSTOCH to solve 30 instances of the S-MDVRPPD. Both heuristics were executed 10 times for each instance, and Avg. Cost and Time are the average solution cost and average time over these 10 executions.

Instance			ILS-VND			TS		
Name	$\|V\|$	$\|D\|$	**Best Cost**	**Avg. Cost**	**Time (s)**	**Best Cost**	**Avg. Cost**	**Time (s)**
ulysses16a	16	2	**66,44**	67,41	0,91	69,96	71,14	0,46
ulysses16b	16	2	**30,68**	30,69	0,72	31,94	31,94	0,41
ulysses16c	16	2	**44,92**	44,92	0,70	45,26	45,93	0,66
ulysses16d	16	4	**57,59**	57,59	0,26	65,05	65,55	0,53
ulysses16e	16	4	**53,21**	53,21	0,27	81,09	81,09	0,60
bayg29a	29	3	**7082,80**	7247,16	6,30	8010,40	9348,96	11,37
bayg29b	29	9	**11890,90**	11990,20	0,46	15745,10	15745,10	8,50
bayg29c	29	9	**10631,60**	10660,90	0,61	13856,80	13856,80	7,43
bayg29d	29	3	**7121,43**	7219,10	8,34	7705,80	8449,30	10,00
bayg29e	29	5	**8078,92**	8116,32	4,37	10800,10	11700,60	11,27
dantzig42a	42	8	**676,93**	696,76	5,11	985,70	1009,46	28,37
dantzig42b	42	4	**542,06**	557,25	28,43	676,76	723,88	69,13
dantzig42c	42	10	**714,80**	730,77	2,00	1130,56	1130,56	35,98
dantzig42d	42	2	**599,89**	628,23	55,22	617,35	696,75	41,13
dantzig42e	42	10	**671,76**	680,96	3,41	1244,39	1334,65	45,51
eil51a	51	3	**351,98**	369,15	90,39	418,03	447,26	122,13
eil51b	51	7	**368,78**	384,45	24,70	537,55	586,47	66,46
eil51c	51	9	**340,74**	354,02	11,94	579,64	596,45	45,63
eil51d	51	7	**323,60**	329,68	39,62	493,12	535,91	107,50
eil51e	51	5	**339,41**	344,22	22,97	409,80	484,96	93,16
st70a	70	4	**621,25**	690,76	224,80	810,95	875,85	408,45
st70b	70	6	**728,87**	760,81	88,89	871,67	945,15	765,90
st70c	70	8	**680,43**	710,06	115,43	1078,96	1249,14	459,02
st70d	70	8	**620,63**	669,17	87,50	1012,48	1071,10	398,99
st70e	70	6	**575,27**	616,96	200,63	960,09	1048,68	483,40
eil76a	76	6	**556,89**	599,37	171,47	703,79	790,67	480,91
eil76b	76	2	**464,73**	486,68	606,41	491,96	530,28	352,76
eil76c	76	4	**513,01**	564,59	309,40	559,31	636,99	381,38
eil76d	76	6	**522,01**	557,11	236,04	694,46	806,43	582,45
eil76e	76	6	**490,29**	534,55	176,54	642,77	723,63	404,50

TABLE III
PERCENTAGE DECREASE IN SOLUTIONS COST OBTAINED BY ILS-VND COMPARED TO TABUSTOCH.

	ILS-VND	
Instance	**Best (%)**	**Avg. (%)**
ulysses16a	-5,30	-5,25
ulysses16b	-4,12	-3,92
ulysses16c	-0,76	-2,20
ulysses16d	-12,96	-12,14
ulysses16e	-52,39	-34,38
bayg29a	-13,10	-22,48
bayg29b	-32,41	-23,85
bayg29c	-30,34	-23,06
bayg29d	-8,21	-14,56
bayg29e	-33,68	-30,63
dantzig42a	-45,61	-30,98
dantzig42b	-24,85	-23,02
dantzig42c	-58,17	-35,36
dantzig42d	-2,91	-9,83
dantzig42e	-85,24	-48,98

	ILS-VND	
Instance	**Best (%)**	**Avg. (%)**
eil51a	-18,76	-21,16
eil51b	-45,77	-52,55
eil51c	-70,11	-68,48
eil51d	-52,39	-62,56
eil51e	-20,74	-40,88
st70a	-30,53	-26,79
st70b	-19,59	-24,23
st70c	-58,57	-75,92
st70d	-63,14	-60,06
st70e	-66,89	-69,98
eil76a	-26,38	-31,92
eil76b	-5,86	-8,96
eil76c	-9,02	-12,82
eil76d	-33,04	-44,75
eil76e	-31,10	-35,37

VII. CONCLUSION

This article described a new and practical SVRP involving multiple depots, and pickup and delivery (S-MDVRPPD). Contrary to the deterministic case, it is not easy to compute the objective function associated with a solution [20]. We presented a closed-form expression to compute the expected length of an *a priori* sequence under general probabilistic assumptions. In order to dealt with S-MDVRPPD, an algorithm based on the Iterated Local Search metaheuristic, which uses a VND heuristic as local search procedure was proposed. We use six local search operators, *Shift(1,0), Swap(1,1), 2-opt, 3-opt, Reverse, and Mix-Shift*. Also, we use two perturbation mechanisms, *Double-Swap and Depot-exchange*. We propose a heuristic based on *Mix-shift* operator to refine the initial solution of the ILS-VND. The ILS-VND was compare with a tabu search algorithm (TABUSTOCH). We report the results for 30 instances. The results show that the ILS-VND was superior to the TABUSTOCH for all instances tested. Our approach can be used as benchmark for future research in this area. The S-MDVRPPD can be further generalized to handle more practical constraints, e.g., limited capacity vehicles, time windows and stochastic demands.

ACKNOWLEDGMENT

This work is financed by FAPESP (Proc. 2015/11937-9, 2016/01860-1, 2018/08879-5), CNPq (Proc. 314366/2018-0, 425340/2016-3, 304856/2017-7), ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia within project POCI-01-0145-FEDER-031908.

REFERENCES

[1] I. H. Dridi, E. B. Alaïa, P. Borne, and H. Bouchriha, "Optimisation of the multi-depots pick-up and delivery problems with time windows and multi-vehicles using PSO algorithm," *International Journal of Production Research*, pp. 1–14, sep 2019. doi: 10.1080/00207543.2019.1650975

[2] A. Subramanian, L. Drummond, C. Bentes, L. Ochi, and R. Farias, "A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery," *Computers & Operations Research*, vol. 37, no. 11, pp. 1899–1911, nov 2010. doi: 10.1016/j.cor.2009.10.011

[3] M. Gendreau, G. Laporte, and R. Séguin, "A tabu search heuristic for the vehicle routing problem with stochastic demands and customers," *Operations Research*, vol. 44, no. 3, pp. 469–477, jun 1996. doi: 10.1287/opre.44.3.469

[4] H. N. Psaraftis, M. Wen, and C. A. Kontovas, "Dynamic vehicle routing problems: Three decades and counting," *Networks*, vol. 67, no. 1, pp. 3–31, aug 2015. doi: 10.1002/net.21628

[5] I. Kara and T. Bektas, "Integer linear programming formulations of multiple salesman problems and its variations," *European Journal of Operational Research*, vol. 174, no. 3, pp. 1449–1458, nov 2006. doi: 10.1016/j.ejor.2005.03.008

[6] M. Assaf and M. Ndiaye, "Multi travelling salesman problem formulation," in *2017 4th International Conference on Industrial Engineering and Applications (ICIEA)*. IEEE, apr 2017. doi: 10.1109/iea.2017.7939224

[7] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, jun 2006. doi: 10.1016/j.omega.2004.10.004

[8] V. N. Pereira, M. C. San Felice, P. H. D. Hokama, and E. C. Xavier, "The steiner multi cycle problem with applications to a collaborative truckload problem," in *17th International Symposium on Experimental Algorithms (SEA 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018. doi: 10.4230/LIPICS.SEA.2018.26

[9] M. Polacek, R. F. Hartl, K. Doerner, and M. Reimann, "A variable neighborhood search for the multi depot vehicle routing problem with time windows," *Journal of Heuristics*, vol. 10, no. 6, pp. 613–627, dec 2004. doi: 10.1007/s10732-005-5432-5

[10] G. Laporte, R. Musmanno, and F. Vocaturo, "An adaptive large neighbourhood search heuristic for the capacitated arc-routing problem with stochastic demands," *Transportation Science*, vol. 44, no. 1, pp. 125–135, feb 2010. doi: 10.1287/trsc.1090.0290

[11] P. Stodola, "Hybrid ant colony optimization algorithm applied to the multi-depot vehicle routing problem," *Natural Computing*, vol. 19, no. 2, pp. 463–475, jan 2020. doi: 10.1007/s11047-020-09783-6

[12] J. E. Mendoza and J. G. Villegas, "A multi-space sampling heuristic for the vehicle routing problem with stochastic demands," *Optimization Letters*, vol. 7, no. 7, pp. 1503–1516, sep 2012. doi: 10.1007/s11590-012-0555-8

[13] A. L. Erera, M. Savelsbergh, and E. Uyar, "Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints," *Networks*, vol. 54, no. 4, pp. 270–283, dec 2009. doi: 10.1002/net.20338

[14] J. C. Goodson, "A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands," *European Journal of Operational Research*, vol. 241, no. 2, pp. 361–369, mar 2015. doi: 10.1016/j.ejor.2014.09.031

[15] B. H. O. Rios, E. F. G. Goldbarg, and G. Y. O. Quesquen, "A hybrid metaheuristic using a corrected formulation for the traveling

car renter salesman problem," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, jun 2017. doi: 10.1109/cec.2017.7969584

[16] B. H. O. Rios, E. F. G. Goldbarg, and M. C. Goldbarg, "A hybrid metaheuristic for the traveling car renter salesman problem," in *2017 Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE, oct 2017. doi: 10.1109/bracis.2017.20

[17] J. Oyola, H. Arntzen, and D. L. Woodruff, "The stochastic vehicle routing problem, a literature review, part II: solution methods," *EURO Journal on Transportation and Logistics*, vol. 6, no. 4, pp. 349–388, nov 2016. doi: 10.1007/s13676-016-0099-7

[18] Y. Kuo and C.-C. Wang, "A variable neighborhood search for the multi-depot vehicle routing problem with loading cost," *Expert Systems with Applications*, vol. 39, no. 8, pp. 6949–6954, jun 2012. doi: 10.1016/j.eswa.2012.01.024

[19] N. Mladenović and P. Hansen, "Variable neighborhood search," *Computers & Operations Research*, vol. 24, no. 11, pp. 1097–1100, nov 1997. doi: 10.1016/s0305-0548(97)00031-2

[20] D. J. Bertsimas, "A vehicle routing problem with stochastic demand," *Operations Research*, vol. 40, no. 3, pp. 574–585, jun 1992. doi: 10.1287/opre.40.3.574