A GRASP-based Solution Construction Approach for the Multi-Vehicle Profitable Pickup and Delivery Problem

Abeer I. Alhujaylan[1], Manar I. Hosny[2]
Computer Science Department
College of Computer and Information Sciences (CCIS)
King Saud University (KSU)
Riyadh, Saudi Arabia

***Abstract*—With the advancement of e-commerce and Internet shopping, the high competition between carriers has made many companies rethink their service mechanisms to customers, in order to ensure that they stay competitive in the market. Therefore, companies with limited resources focus on serving only customers who provide high profits at the lowest possible cost. The Multi-Vehicle Profitable Pickup and Delivery Problem (MVPPDP) is a vehicle routing problem and one variant of the Selective Pickup and Delivery Problem (SPDP) that is considered to plan the services for these types of companies. The MVPPDP aims to serve only the profitable customers, where the products are transformed from a selection of pickup customers to the corresponding delivery customers, within a given travel time limit. In this paper, we utilize the construction phase of the well-known Greedy Randomized Adaptive Search Procedure (GRASP) to build initial solutions for the MVPPDP. The performance of the proposed method is compared with two greedy construction heuristics that were previously used in the literature to build the initial solutions of the MVPPDP. The results proved the effectiveness of the proposed method, where eight new initial solutions are obtained for the problem. Our approach is especially beneficial for building a population of solutions that combine both diversity and quality, which can help to obtain good solutions in the improvement phase of the problem.**

Keywords*—Selective pickup and delivery problem; multi-vehicle profitable pickup and delivery problem; greedy randomized adaptive search procedure; metaheuristic algorithms

I. INTRODUCTION

Transportation management is considered one of the most difficult problems facing people and governments in different countries all over the world. In our daily life, millions of people use land, sea, or air transport means to commute from one place to another, raising the need to optimize the planning of these services, in order to reduce their cost as well as their negative environmental impacts. Therefore, a lot of research has been conducted recently to address these problems in the fields of computer science, operations research, and industrial engineering. Land transport, in particular, has received a great interest from researchers, due to its huge volume. Research efforts try to optimize the daily use of the means of transportation, such as cars, buses, trucks, trains, motorcycles, trams, etc. Among the most known land transport problems are: vehicle routing problems [1], pickup and delivery problems [2], bus scheduling problems [3], truck routing problems [4], cash transportation problems [5], railroad blocking problems [6], and others [7][8][9].

Researchers in this field generally aim at minimizing congestion and the environmental damage of the transportation services, caused by harmful emissions, such as carbon dioxide and other greenhouse gases, which cause pollution and global warming, and have a negative effect on people's health. In the shipment sector, for instance, the average fleet emission for delivery trucks and vans is 10.17 kg of CO_2 per gallon of diesel consumed [10]. Furthermore, many trucks are not exploited to their full capacity, where statistics indicate that 15% to 30% of pollution, traffic congestion, and accidents are caused by empty trucks [11]. Besides the environmental damage, the inefficient regulation of the trucks' paths and the non-exploitation of their full capacity has an economic effect on the companies that work in the transport sector. In order for these companies to remain and continue its activities in the market, many solutions are suggested to increase the companies' profits, as well as to reduce their costs.

The Vehicle Routing Problem (VRP) is a well-known combinatorial optimization problem that deals with transportation network management and the scheduling and distribution of vehicles and goods. The VRP is concerned with planning and organizing the distribution of goods to find the appropriate routes to transfer the customers' demands by using one or more homogeneous fleet of vehicles. Each vehicle has a limited capacity and it starts its tour from a distribution center (depot), then it transfers goods to customers, and returns to the distribution center. In the literature, several types of the VRP with different complex constraints have been presented and solved over the last 50 years which contributed to minimizing a lot of road transportation problems, such as, pollution and congestion[12].

The Pickup and Delivery Problem (PDP) is an important variant of the VRP, which aims to minimize the total transportation cost when distributing goods or people from one location (pickup node) to another location (delivery node). The PDP also has several important variants and applications, such as the transportation of raw materials from suppliers to

factories, the distribution of beverages and the collection of empty bottles and cans, shipping cargos, etc. One relatively new variant of the PDP is the Selective Pickup and Delivery Problem (SPDP), which has recently started to receive interest in the academic literature. The SPDP can be distinguished from the standard PDPs by relaxing the constraint that all nodes must be visited. The SPDP helps companies that have limited resources and wish to provide high level services to customers with minimum possible cost, by finding the best routes to deliver their products and pick up some of them. Also, the SPDP contributes positively to environmental and economic considerations; specifically, it helps to reduce the harmful impacts of transportation that result from pollution and congestion, by selecting only a subset of customers to be visited. As such, solving the SPDP helps to achieve the goals of green supply chain management [13][14][15]. There are two types of SPDPs: (1) SPDPs subject to minimizing the travelling cost only (e.g. [16][17][18]), and (2) SPDPs subject to minimizing the travelling cost and maximizing the profits collection (e.g. [19] [20][21]).

The Multi-Vehicle Profitable Pickup and Delivery Problem (MVPPDP), which has been presented by Gansterer et al. [22], belongs to the second type of the SPDP. It considers both travel cost and profit's collection in the planning process. The MVPPDP is a static problem with a central distribution (depot), predefined number of homogenous fleet of vehicles and a set of requests. Each request has a predefined pair of customers, a pickup customer and a delivery customer. Moreover, these requests include transferring a number of homogenous products from pickup customers (origins) to their corresponding delivery customers (destinations) with certain associated profits. The MVPPDP aims to plan perfect routs to serve some of the customers in a one-day planning horizon, with the aim of maximizing the total collected profits minus the total travel costs. Some real-life applications of this problem include: food delivery mobile apps, delivery companies that transfer goods from factories to related shops, etc.

The MVPPDP is an NP-hard problem [22], which means that exact algorithms can find an optimal solution for only a small number of input data. Therefore, metaheuristic algorithms have been often used to find good solutions that are not necessarily optimal, in a reasonable processing time.

In this paper, we present a new approach to construct initial solutions for the MVPPDP. We adapt the construction phase of the well-known Greedy Randomized Adaptive Search Procedure (GRASP) to the underlying problem. GRASP is a multi-start metaheuristic that is commonly applied to solve different combinatorial optimization problems. It was first introduced by Feo and Resende [23]. It consists of two phases: construction and improvement phase. The construction phase is used to build an initial feasible solution, while the second phase is a local search used to improve the initial solution found in the construction phase to get a local optimum [24]. There are several advantages of GRASP compared to other popular metaheuristics, like Genetic Algorithms, Simulated Annealing and Tabu Search. These include combining the advantages of random and greedy search, which helps the GRASP to be fast, competitive and able to find good solutions in a reasonable time. Furthermore, the number of parameters that need to be tuned is small, which is another advantage that makes the GRASP preferred over other metaheuristics [25]. Also, GRASP has successfully contributed to solving different variants of VRP [26] [27] [28].

The rest of this paper is organized as follows. In Section 2, a review of some related work is presented. The definition and mathematical formulation of the tackled problem is given in Section 3. In Section 4, the proposed method is described in detail. Experimental results are discussed in Section 5. Finally, conclusions and future work are presented in Section 6.

II. Literature Review

The MVPPDP belongs to the type of SPDP that is subject to minimizing the travelling costs and maximizing the profits collection. The MVPPDP aims to visit only the profitable customer pairs, in order to make the gathering operation as profitable as possible. Thus, the MVPPDP can be considered as a combination of two types of problems: the feature of selecting a subset of customers belongs to the SPDP, while the feature of selecting customers that have maximum revenue belongs to the Profitable Tour Problem (PTP). Therefore, the literature review of the MVPPDP will be classified into three parts: The MVRPPDP, the SPDP and the PTP.

A. The Multi-Vehicle Profitable Pickup and Delivery Problem (MVPPDP)

The MVPPDP was presented by Gansterer et al. [22]. Two versions of the General Variable Neighborhood Search (GVNS) were used to solve it: a sequential one (GVNSseq), and a self-adaptive one (GVNSsa). The initial solution was built by using a Greedy Construction Heuristic (C1) and a Two-Stage Cheapest Insertion Heuristic (C2). Then, one of three strategies had been selected randomly to make some changes on the initial solution. After that, 11 neighborhoods operators were used to improve the solution. They tested the performance of proposed algorithms against Guided Local Search (GLS) on a newly created dataset that is divided into three sizes: small, medium, and large. The performance of the C2 heuristic was found to be better than C1 for most instances. Also, the results indicated that both variants of GVNS with the C2 heuristic outperform GLS for all sizes of test data in terms of solution quality. However, GLS had an advantage regarding the average runtimes in both medium and large sized instances.

Haddad [29] presented a new method for the MVPPDP. The proposed algorithm combined Iterated Local Search (ILS) and Random Variable Neighborhood Descent (RVND). Also, the algorithm is not limited to accept only the feasible solutions during the search. The same greedy constructive heuristic that was used in [22] was adopted to construct the initial solution. In order to improve the solution, several neighborhood moves were applied. The proposed algorithm was tested on the benchmark instances proposed by [22]. It proved its efficiency in addressing the small and medium sized instances, where it was able to find new best solutions on 6 instances. However, the performance of the proposed algorithm was not good enough for large sized instances.

B. The Selective Pickup and Delivery Problem (SPDP) Subject to Travelling Cost and Profits Collection

The work in [13] is an application of the SPDP, where three heuristics were proposed to solve a complex real-life problem appearing in the soft drinks distribution and recyclable containers collection in a Quebec based company. The algorithms are the Nearest Neighbor Heuristic (NNH), First Petal Heuristic (FPH), and Second Petal Heuristic (SPH). The results on real-life instances showed a reduction in distance by 23 %. The authors in [15] proposed a Mixed Integer Liner Programming (MILP) and a Tabu Search (TS) to solve the Single Vehicle Routing Problem with Deliveries and Selective Pickups (SVRPDSP). Also, three heuristics were developed: shifting pickups (SP), optimization of the sequence of customers in the route (RC), and reducing the number of second visits (RV). The empirical results indicated that the proposed heuristics gave near-optimal solutions for 68 instances that were derived from the VRP library[1]. In [14], the Selective Multi-Depot Vehicle Routing Problem with Pricing (SMDVRPP) was introduced to tackle the reverse logistics problem of companies that aim to collect cores of durable goods from its merchants after re-buying or trade-in by customers to encourage them to buy. Two MILPs models and a Rich Neighborhood Tabu Search (TS-RN) were used to solve this problem. The proposed heuristic was tested on 40 randomly generated instances, showing promising results in terms of both accuracy and efficiency. In [20][21], two Hybrid metaheuristics were presented to solve the SVRPDSP: Hybrid General Variable Neighborhood Search (HGVNS) and a hybrid metaheuristic based on an Evolutionary Algorithm (EA). Both metaheuristics were tested on the same instances as in [15], where the experimental results proved the effectiveness and robustness of the proposed metaheuristics. For the interested readers, other applications of the SPDP can be found in [19] [30] [31] [32] and [33].

C. The Profitable Tour Problem (PTP)

The Profitable Tour Problem (PTP) shares with the SPDP the same goal of finding a route that maximizes the difference between the total gained profit and the total cost of traveling. The difference between them is that in the PTP no restrictions are imposed on the vehicle route (i.e., the vehicle's capacity, maximum time of trip and precedence constraints are not considered) [34]. The studies that address the PTP are so rare in the literature [35]. In [36], a new neighborhood search is used to solve the standard VRP. Then, an algorithm that relies on resource constrained shortest paths is used to find the optimal subsequence of visits. To evaluate the proposed method, three metaheuristics were used: local improvement method, a hybrid genetic search and an iterated local search. The proposed strategy performed well, where it contributed to finding new 52 best solutions. In [37] [38], an exact approach and three metaheuristics were proposed to solve the capacitated team orienteering and profitable tour problems: a VNS algorithm and two versions of Tabu Search (TS). The first version of TS explored only feasible solutions, while the other one explored the infeasible solutions as well. All proposed methods succeeded to find the optimal solution when compared on instances that were solved by a branch-and-price algorithm. In [39], a hybrid VNS was proposed to solve a rich variant of the PTP (RPTP). Instances of the Orienteering Problem with Time Windows (OPTW) were used to test the efficiency of the proposed algorithm, where it was able to get good solutions in a reasonable time.

[1] http://or.dei.unibo.it/library/vrplib-vehicle-routing-problem-library

III. Problem Definition and Mathematical Formulation

The MVPPDP is a static problem, where all problem constituents are known in advance. The problem is characterized by having a central distribution location (depot), a predefined number of homogenous vehicles and a set of customers' requests. Each request has a predefined pair of customers: a pickup customer and a delivery customer. Moreover, these requests include transferring a number of homogenous products (i.e., they have the same characteristics and quality) from pickup customers (origins) to their corresponding delivery customers (destinations) to get profits. Fig. 1 represents a simple example of the problem, where there are three vehicles that serve only the profitable customer pairs (i.e., those who make high revenues at the lowest possible cost) among 50 customers [22].

Several constraints should be taken into consideration when solving the MVPPDP:

- *Pair constraint:* Each product has a predefined pair of pickup and delivery customers.
- *Precedence constraint*: the pickup customer should be visited before its related delivery customer.
- *Time constraint*: Each vehicle has a certain daily travel time that should not be exceeded during serving the customers.
- *Capacity constraint*: The total amount of products gathered at any point in time should not exceed the capacity of the vehicle.
- Each vehicle journey should start and end from/at the depot with empty load.
- Each customer pair cannot be visited more than once.

The objective is to select a set of customer pairs to serve such that the revenue is maximized and its total cost is minimized.

The MVPPDP can be formally defined as follows [40]: Let $G = (V, A)$ be a graph in which $V = \{0, \dots, 2n + 1\}$ defines the vertex set, where the vertex $(0, 2n + 1)$ represents a central depot, while the remaining vertices represent the pickup customers $P = \{1, \dots, n\}$, and delivery customers $D = \{n + 1, \dots, 2n\}$. $A = \{(i, j): i, j \in V, i \neq j\}$ defines the arc set, where each arc is associated with a non-negative routing cost c_{ij}. Each delivery vertex i has a revenue r_i to be gained when visiting it. Also, each vertex i has a supply q_i (pickup, $q_i > 0$) or demand (delivery, $q_{n+i} = -q_i$). At start, there is no supply or demand in the depot ($q_0 = 0$). In addition, each vertex i has service duration d_i. There is a set of vehicles $K = \{1, \dots, m\}$, each vehicle has a load capacity C and maximum tour time T.

Some notations used in the following mathematical model are described as follows: $\boldsymbol{n}$: number of pickup customers and number of delivery customers, $\boldsymbol{m}$ **:** number of vehicles, **P:** set of pickup customers, $P = \{1, .., n\}$, $\boldsymbol{D}$: set of delivery vertices, $D = \{n+1, \ldots, 2n\}$, $\boldsymbol{V}$ **:** set of all vertices including the depot (starting point= 0 and ending point= $2n+1$), $V = P \cup D \cup \{0, 2n+1\}$, $\boldsymbol{K}$: set of available vehicles, $K = \{1, \ldots, m\}$, $\boldsymbol{r_i}$: revenue gained by serving a delivery customer at vertex i, $\boldsymbol{q_i}$: supply (pickup : $\boldsymbol{q_i} > 0$) or demand (delivery: $\boldsymbol{q_{n+i}} = -\boldsymbol{q_i}$) at vertex i, $\boldsymbol{d_i}$: duration of service at vertex i, $\boldsymbol{c_{ij}}$: transportation cost when traveling from i to j, $\boldsymbol{t_{ij}}$: travel time between vertex i and vertex j, $\boldsymbol{C}$ **:** loading capacity of a vehicle, **T:** maximum travel time of a vehicle, $\boldsymbol{x_{ijk}}$: binary decision variable equal to one if and only if arc ij is used by vehicle k, $\boldsymbol{Q_{ik}}$: decision variable giving the loading amount of vehicle k after visiting vertex i, $\boldsymbol{B_{ik}}$: decision variable for the beginning of service time of vehicle k at vertex i.

The mathematical model can be formulated as in [22] as follows:

$$Maximize \sum_{i\in V}\sum_{j\in V}\sum_{k\in K}(r_i - c_{ij})x_{ijk} \quad (1)$$

The constraints are

$$\sum_{i\in V}\sum_{k\in K} x_{ijk} \le 1 \qquad j \in V \quad (2)$$

$$\sum_{j\in V}\sum_{k\in K} x_{ijk} \le 1 \qquad i \in V \quad (3)$$

$$x_{i0k} = 0 \qquad i \in V, k \in K \quad (4)$$

$$x_{2n+1,jk} = 0 \qquad j \in V, k \in K \quad (5)$$

$$\sum_{i\in V}(x_{ijk} - x_{jik}) = 0 \qquad j \in V\ \{0, 2n+1\}, k \in K \quad (6)$$

$$\sum_{j\in V}(x_{ijk} - x_{n+i,jk}) = 0 \qquad i \in P, k \in K \quad (7)$$

$$\sum_{j=V} x_{0jk} = \sum_{i=V} x_{i,2n+1,k} = 1 \qquad k \in K \quad (8)$$

$$(x_{ijk} = 1) \Rightarrow Q_{jk} = Q_{ik} + q_j \qquad i \in V, j \in V \setminus \{0\}, k \in \quad (9)$$

$$Q_{ik} \le C \qquad i \in V, k \in K \quad (10)$$

$$Q_{0k} = 0 \qquad k \in K \quad (11)$$

$$B_{ik} \le B_{n+i,k} \qquad i \in P, k \in K \quad (12)$$

$$B_{jk} \ge x_{ijk}(B_{ik} + d_i + t_{ij}) \qquad i \in V, j \in V, k \in K \quad (13)$$

$$B_{2n+1,k} \le \mathrm{T} \qquad k \in \mathrm{K} \quad (14)$$

$$B_{0k} = 0 \qquad k \in K \quad (15)$$

$$x_{ijk} \in 0{,}1 \qquad i \in V, j \in V, k \in K \quad (16)$$

$$Q_{ik}, B_{ik} \ge 0 \qquad i \in V, k \in K \quad (17)$$

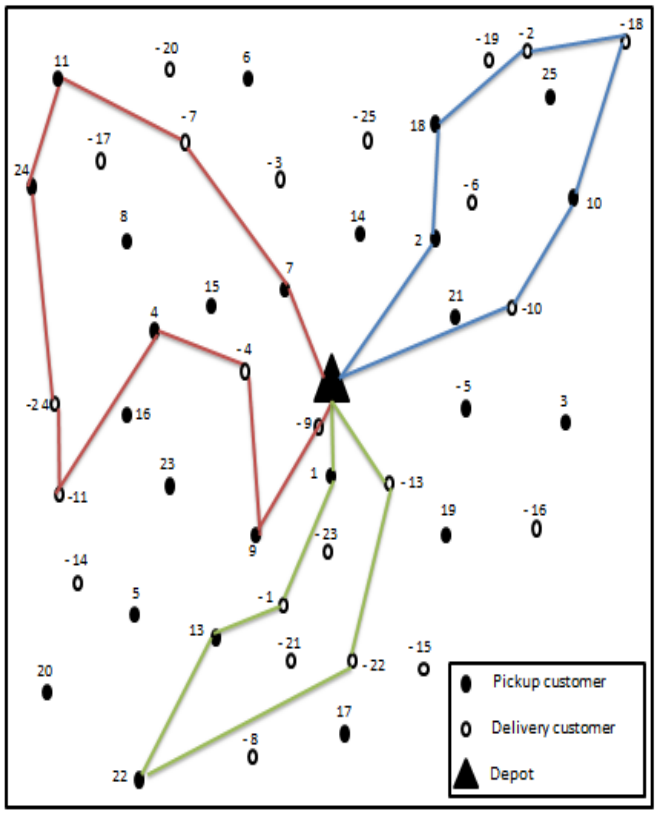

Fig. 1. An Example of the MVPPDP.

The objective function (1) tries to maximize the total profit by subtracting the total travel costs from the total revenues. Constraints (2) and (3) mean that each vertex is visited at most once. Constraints (4) and (5) mean that the origin depot (starting point) cannot be entered, and the destination depot (end point) cannot be departed by any vehicle k. Constraint (6) means flow conservation. Constraint (7) means that each customer pair (pickup customer and delivery customer) has to be served by the same vehicle. Constraint (8) means all vehicles must start from the depot and return to the depot. Constraint (9) means exceeding the vehicle capacity is not allowed. Constraint (10) means that the load of a vehicle is bounded by C. Constraint (11) means that all the vehicles start with empty load. Constraint (12) means that the delivery customer cannot be visited before its corresponding pickup customer. Constraint (13) means that the earliest time to start the service at vertex j is given by the beginning of service time at vertex i plus the service time at i and the travel time between i and j. Constraint (14) means that the maximum time for each vehicle is restricted by T. Constraint (15) means each vehicle starts at the depot at time 0. Finally, decision variables are defined by (16) and (17).

IV. PROPOSED METHOD

Most metaheuristic algorithms start solving a problem by generating one or more initial solutions and then they improve them using some local search method. The type of method used in generating the initial solution(s) plays an important role in the efficiency and effectiveness of those algorithms, regardless of the improvement method used. There are two main approaches that can be used to construct initial solutions: random and greedy. The random approach is simple, fast, and can create diverse solutions. However, the produced random solutions maybe of very low quality, which makes the job of the improvement procedure harder. In contrast, the greedy approach, which takes the objective function into consideration while constructing a solution, is often complicated and needs more computation time. In addition, although it overcomes the other approach with respect to the quality of solution, it risks getting stuck in local optima mainly due to lack of diversity [24]. To combine the advantages of the random and greedy approaches, Greedy-Randomized (GR) procedures can be used. The main idea behind these procedures in general is to select at random one of the best (greedy) decisions (instead of the absolute best as done in pure greedy methods). This helps in diversifying the search and is especially beneficial for use within population-based metaheuristics.

In our proposed method, we use a greedy-randomized procedure that is based on the well-known GRASP (Greedy Randomized Adaptive Search Procedure) to generate initial solutions for the MVPPDP. As previously mentioned, the GRASP consists of a construction phase (greedy-randomized) and an improvement phase (local search). In this paper we only utilize the construction phase to generate the initial solutions for the MVPPDP. Any local search method (e.g. hill climbing, simulated annealing, genetic algorithm, etc.) can be used later to improve the constructed solution quality. The framework of the greedy randomized procedure is shown in Algorithm 1 [24]. We adapted the GR procedure to our problem as explained next.

First, we select seed customers for each route based on a greedy approach. Then, we fill up the routes with the rest of the customers, based on combining two criteria: greediness and randomness.

Because of the importance of selecting seed customers and their clear impact on the performance of the algorithm in general, several heuristics have been proposed. The seed vertex is usually selected according to a specific criterion, often in relation to the depot (e.g., nearest, farthest, related, ..., etc.). In our method, we select the seed customer pairs for each route according to a certain measure that we call the *Customer Benefit (CB)*. The CB is calculated based on the revenue gained after delivering the demand to the delivery customer, with respect to the distance between the customer pair. Using the notations of Section 3, the customer benefit of a customer pair $(i, n+i)$ is calculated as shown in Equation (18):

$$CB_{i,n+i} = \frac{r_{i,n+i}}{c_{i,n+i}} \quad (18)$$

Algorithm 1: The greedy-randomized procedure[24]

$S = \{\}$; //Initialize the current solution to empty
Evaluate the fitness values of all candidate elements ;
Repeat
Construct the Restricted Candidate List RCL based on the fitness values of the candidate elements
// select one element randomly from the RCL
e = SelectAtRandom(RCL) ;
If $s \cup e$ is feasible **then**
$s \leftarrow s \cup e$;
Reevaluate the fitness values of candidate elements;
Until Complete solution is constructed

Where $r_{i,n+i}$ is the revenue gained after delivering the goods to the delivery customer, and $c_{i,n+i}$ is the cost of travelling from the pickup node to the delivery node. Thus, the CB tries to rank customers based on their relative benefit versus cost with respect to the operating company. The customer pair that has the highest benefit is selected to be a seed customer.

After selecting the seed customer in the route, several candidate sub-routes will be generated for each vehicle. The following steps will be repeated to fill up each vehicle until either the final allowed time of the vehicle has been reached or no more customer pairs need to be served. Firstly, insert all customer pairs that are not served yet, where each pair of customers is inserted individually into a route in the best possible position. That is, the position that leads to the lowest possible total distance. While doing this, in order to achieve the precedence constraint, the best position for the delivery customer will be selected after adding the pickup customer.

After that, we calculate what we call the *Insertion Ratio (IR)* for each candidate pair [40]. For example, suppose we have the nodes $(a, -a, b, -b, 0)$, where $a, b \in P$ (pickup) and $-a, -b \in D$ (delivery), and 0 represents the depot. Also assume that the current route is $\{0, a, -a, 0\}$, where $(a, -a)$ is a seed customer pair, while $(b, -b)$ is the candidate customer pair to be added in the route. After determining the best position of $(b, -b)$, assume that the route will be $\{0, b, a, -b, -a, 0\}$. Thus, the insertion ratio of $(b, -b)$ is computed by dividing its revenue by its insertion cost (in its best position) in the route, as shown in Equation 19:

$$IR_{b,-b} = \frac{r_{b,-b}}{(c_{0,b}+c_{b,a}-c_{0,a}) + (c_{a,-b}+c_{-b,-a}-c_{a,-a})} \quad (19)$$

Thus, the IR represents the greedy property of selecting the next customer to be inserted in the route. Then, all the candidate customer pairs will be sorted in descending order with respect to their IR values and assigned to the Candidate Solution Set (CSS). After this, the first half of the candidate solution set is assigned to the Restricted Candidate List (RCL), so that the opportunity of selecting a good pair of customers increases. Thus, the random property lies in choosing one customer pair randomly from the RCL. The previous process will be repeated to add the rest of the un-served pairs of customers, while considering the time limit and the vehicle capacity constraint. Once the time limit of the vehicle has been exhausted, a new route (vehicle) will be initiated. At the end, we will have an initial solution for the MVPPDP which contains a number of routs equal to the number of vehicles.

To increase the opportunity of getting better solutions, we decided to use the concept of the population-based metaheuristics, where a number of solutions are generated at the same time and then we select the best one according to a certain evaluation criterion. Therefore, the previous method of generating the initial solution will be repeated several times according to the population size to generate a number of candidate solutions. The quality of each candidate solution is evaluated in terms of the value of the objective function (OF). This objective function is defined as maximizing the total profit by subtracting the total travel cost from the revenues collected, as shown in Equation (1). After that, the best solution that has the highest OF value is saved in a matrix to compare it with other best solutions that are selected from different iterations. Finally, we select the best overall solution which represents the initial solution for the MVPPDP.

The outline of the construction phase of our GRASP is presented in Algorithm 2, where the meaning of each notation is as follows: ***Pop_Size****:* the size of the population, ***Max_Iter:*** maximum number of iterations, ***Num_Vehicles***: number of vehicles**,** ***CSS:*** Candidate Solutions Set, ***RCL:*** Restricted Candidate List, ***US:*** Un-Served pairs of customers and ***SM:*** Solutions Matrix that contains the best solutions in the population for each iteration.

V. Computational Experiments

The computational experiments aim to compare the performance of our proposed algorithm with the construction heuristics used in [40]: the Greedy Construction Heuristic and the Two-Stage Cheapest Insertion Heuristic. Before we present the results of the experiments, we describe the dataset that was used to test the algorithm and the values chosen for each parameter.

A. Test Instances

We used the same instances that are proposed in [22] and [40][2]. The data instances are 36 instances, which are classified into three groups: small size, which contains 20 and 50 customers, medium size, which contains 100 and 250 customers and large size which contains 500 and 1000 customers. The number of instances in each group is 12 instances. The customers are set to be pickup customers and delivery customers. There is only one central depot, There are at most 8 vehicles which are organized as follows: 2 vehicles to serve 20 customers, 3 vehicles to serve 50 customers, 4 vehicles to serve 100 customers, 5 vehicles to serve 250 customers, 6 vehicles to serve 500 customers and 8 vehicles to serve 1000 customers. The capacity of the vehicle is between [50, 120]. Each pickup customer has an integer demand value between [1, 50], which is transported from a pickup customer and delivered to the related delivery customer to get the associated revenue. The revenue amounts are set to be either fixed for all customers, proportional to the demands, or randomly. Also, the total time limit is set to be either small or large, where the range is within [2500, 15000] to generate short and long routes.

B. Parameter Tuning

Few number of parameters that needs to be tuned is one of the reasons that encouraged us to choose the GRASP metaheuristic. There are only three parameters: the size of the Candidate Solutions Set (CSS), the size of the Restricted Candidate List (RCL), and the number of iterations (Max_Iter). In our method, we added a fourth parameter which is the size of the population (Pop_size); that is the number of solutions that were generated using the greedy-randomized approach, and the best one is selected before moving to the next iteration. Empirical experiments were performed to select the suitable value for each parameter. Six medium-sized data instances have been used to tune the parameters. These are the data instances containing 100 customers that are served by 4 vehicles. All three revenue states were considered in the instances namely fixed revenue for all customers, proportional revenue to demand and randomly selected revenues. Also, each revenue case has been tested twice: when the vehicle capacity is 80 units and the total time limit is 6000 to generate a long route, and when the vehicle capacity is 50 and a total time limit is 4000 to generate a short route. The details of testing each parameter are as follow:

1) The size of the Restricted Candidate List (RCL): To increase the diversity of solutions, half of the solutions that are found in the CSS were assigned to the RCL; i.e., if the number of solutions in $CSS = n$, then $\lceil (n)/2 \rceil$ solutions are assigned to the RCL. This value has been chosen by trial, since choosing a smaller size for the RCL resulted in solutions that are similar to each other and thus lack diversity.

2) The number of iterations: Table I shows the objective function values after testing six datasets: 100 customers with fixed revenue to generate short route (100 - F -S), 100 customers with fixed revenue to generate long route (100-F-L), 100 customers with proportional revenue to generate short route (100-P-S), 100 customers with proportional revenue to generate long route (100-P-L), 100 customers with random revenue to generate short route (100-R-S), 100 customers with random revenue to generate long route (100-R-L). Each data instance was tested with different numbers of iterations: 10, 50,100 and 500. The results of testing show that there is no enhancement in the objective function values after 100 iterations. Thus, the maximum number of iterations was taken to be 100.

3) The size of the population: The same previous dataset instances were tested again to select the appropriate number of solutions in the population. The population size was increased from 10 to 20 solutions. Table II presents the values of the objective function after setting the number of iterations to 100. The results showed that increasing the population size did not lead to an improvement in the objective function value. Thus, the population size was taken to be 10.

[2] We note that [22] is the published paper of the thesis in [40].

Algorithm 2: The pseudocode of the construction phase of GRASP

For i=1 to Max_Iter
  For S=1 to Pop_Size
    For k=1 to *Num_Vehicles*
      Phase 1: Seed Vertex Selection
        Step 1: Compute the Customer Benefit (CB) for all pairs of customers
        Step 2: Select the customer pair that has the highest CB and insert it in the route as a seed customer
      Phase 2: Route Construction
        While Maximum Route Time is not violated
          Step 3: For each customer pair $\in US$, insert it in the best position in the route after checking the precedence, time, and capacity constraints
          Step 4: Compute the insertion Ratio (IR) for all candidate customer pairs, based on their best insertion position calculated in Step 3.
          Step 5: Put all the candidate customer pairs in the CSS in descending order of IR
          Step 6: Assign half of the candidate customer pairs in the CSS to the RCL
          Step 7: Pick one customer pair randomly from the RCL and insert it in its best position in the route, as calculated in Step 3.
        End While
    End For
    Step 8: Compute the objective function for the solution and assign the solution to the SM
  End For
  Step 9: Select the best solution that has the highest objective function in SM and assign it to Final-Best-Solutions matrix
End For
Step 10: Select the best solution that has the highest objective function in the Final-Best-Solutions matrix to be the initial solution for the $MVPPDP$

TABLE I. RESULTS OF TUNING THE NUMBER ITERATIONS

Number of Iterations	**Dataset Instances**					
	100-F-S	***100-F-L***	***100-P-S***	***100-P-L***	***100-R-S***	***100-R-L***
10	12826.14	25948.13	48647.27	80966.13	4336.17	74177.33
50	14053.64	29000.17	49305.84	80966.13	47310.47	75913.21
100	14053.64	29000.17	51323.10	80966.13	47310.47	76031.08
500	14053.64	29000.17	51323.10	80966.13	47310.47	76031.08

TABLE II. RESULTS OF TUNING THE POPULATION SIZE

Dataset instances	**Population size**	
	10	***20***
100-F-S	14053.64	14053.64
100-F-L	29000.17	29000.17
100-P-S	51323.10	51323.10
100-P-L	80966.13	80966.13
100-R-S	47310.47	47310.47
100-R-L	76031.08	76031.08

C. Experimental Results

In this experiment, the proposed algorithm was run 5 times for every dataset. The stopping criterion is the number of iterations which is 100 for all datasets except the large-sized data instances which was set to 20 iterations only, due to time limitation. Before presenting the results of our method, we describe briefly the main construction heuristics that were used in [40], since we are comparing our results with them.

In the *Greedy Construction Heuristic (C1)*, the customer pairs are added sequentially to the route based on the value of the Selection Ratio (SR). In each iteration, the SR is calculated for all candidate customer pairs $(i, n+i)$ as follows:

$$\mathrm{SR}_{i,n+i} = \frac{r_{i,n+i}}{c_{x,i} + c_{i,n+i} + c_{n+i,0}} \quad (20)$$

Where $r_{i,n+i}$ is a revenue of the candidate pair, $c_{x,i}$ is the distance from the last vertex (x) in the route to the candidate pickup customer, $c_{i,n+i}$ is the distance between the candidate customer pair nodes, and $c_{n+i,0}$ is the distance from the candidate delivery customer to the depot. Then, the customer pair that has the highest selection ratio is inserted into the route. This procedure is repeated until either the allowed time is consumed or no more customer pairs need to be served. In the C1 heuristic, there is no need to check the precedence and capacity constraints, because each delivery customer is added directly after the related pickup customer.

The *Two-Stage Cheapest Insertion Heuristic (C2)* works as follow: In the first stage, select the seed customer pair for each route, based on the Idle Distance (ID), which is computed for each candidate pair $(i, n+i)$ as follows:

$$\mathrm{ID}_{i,n+i} = c_{0,i} + c_{n+i,0} \quad (21)$$

Where $c_{0,i}$ is the distance from the depot to the candidate pickup customer, and $c_{n+i,0}$ is the distance from the candidate delivery customer to the depot. The pair with the shortest ID is selected to be a seed customer for the route. Then, the routes are constructed with the rest of unvisited customers by computing the Insertion Ration (IR) for each customer pair as

done in Equation (19). The customer pair with the highest IR is inserted in the route with respect to the precedence and capacity constraints. The second stage of the algorithm, tries to insert all customers that have not been inserted in the first stage, due to capacity violation.

Table III presents the results of constructing initial solutions for the MVRPPDP using our GRASP method. The results are calculated in terms of the objective function (OF) of the solution (i.e., the gained profit), which is equal to total revenue – total cost, as previously shown in Equation (1). Thus, the larger the OF value, the better is the solution obtained. The results of our GRASP were compared with the results of C1 and C2. In Table III, the first column presents the instance name. The following two columns show the results of GRASP in terms of the average and best objective value of 5 runs respectively. The third and fourth columns represent the results of C1 and C2 algorithms in terms of the best objective value. For each group of instances of a particular size, the average results are shown in the highlighted row.

TABLE III. COMPARISON OF THE RESULTS THE GRASP WITH C1 AND C2 IN TERMS OF PROFIT

Instance Name	GRASP (AVG)	GRASP (Best)	C1 (Best)	C2 (Best)
20-F-S	18097.3	18097.3	16185.4	21400
20-F-L	37937.2	**37937.3**	22006.2	32400.1
20-P-S	39957.7	39957.7	39646.5	43528
20-P-L	57763.9	**57763.9**	54849.9	55775.5
20-R-S	30039.9	**30039.9**	22954.4	26884
20-R-L	42845.9	**42845.8**	30486.7	41933
Average_20	37773.65	37773.65	31021.5167	36986.7667
50-F-S	18396.2	**18443.6**	15000.5	17958.2
50-F-L	32686.3	33259.2	34988.2	43008.3
50-P-S	53631.8	53631.8	53694.8	38796.9
50-P-L	91164.1	92502.1	99109	62731.2
50-R-S	24364.4	24364.5	19789.7	24619.2
50-R-L	52889.2	**54712.1**	45326.9	51723.3
Average_50	45522	46152.21667	44651.5167	39806.1833
100-F-S	13369.3	13749.3	19033.1	28818.1
100-F-L	27336.9	28692.9	31825.6	55322.2
100-P-S	51218.1	**54475.1**	44329.1	46547.4
100-P-L	80299.3	**86141.8**	69459.3	79541.3
100-R-S	46160.4	46450.7	49912,7	70214.6
100-R-L	76019.9	77136.6	63116.1	91663.1
Average_100	49067.3167	51107.73333	45552.64	62017.7833
250-F-S	10831.9	11231.9	27676.1	40906.1
250-F-L	49193.5	50980.9	44456	67845.1
250-P-S	33041.9	34606.8	63930	43247.3
250-P-L	102591	107171.1	112471	94126.8
250-R-S	40446.5	41914	79486.1	102873
250-R-L	128304.5	129344.6	130371	143886
Average_250	60734.8833	62541.55	71898.3667	83647.3833
500-F-S	21758.3	24012.2	49210	78580.2
500-F-L	84616.4	85889.1	73299.8	135652
500-P-S	49864.8	51297.9	124075	84513.6
500-P-L	138057.2	142740.2	179001	169098
500-R-S	54859.3	56901.7	108049	116568
500-R-L	163822.4	166232.5	170205	218965
Average_500	85496.4	87845.6	117306.633	133896.133
1000-F-S	9132.4	11045.9	27655.6	66345.3
1000-F-L	56276.1	58068.2	32078.4	112840
1000-P-S	134015.9	141167.4	264997	152307
1000-P-L	325103.1	340418.4	380514	318268
1000-R-S	99173.9	102051	193390	197083
1000-R-L	268887.4	271613.1	275805	362266
Average_1000	148764.8	154060.6667	195740	201518.217

D. Results and Discussion

Observing the results in Table III, the proposed algorithm demonstrated good performance on average in solving the small and some medium-sized data instances, where we got new best solutions for 8 data instances (bold values). Also, we got 8 solutions that were better than either C1 or C2 (underlined values). However, the proposed algorithm was not the best one for the large instances on average, but its performance was acceptable in some cases where we got two solutions that are better than one of the other two heuristics. This is probably due to decreasing the number of iterations for the large instances due to extensive time consumption. Also, we observed that when the GRASP was better than one of the other heuristics, its performance was better than C1 for instances with fixed and random profits (except 20-P-S). On the other hand, it was better than C2 on instances with proportional profits to its demand. Overall, we believe that our method is comparable with the other two methods and has the advantage of being able to construct a number of solutions (a population) that are characterized with the greedy-randomized feature, rather than just one greedy solution as done in the other two construction methods.

VI. Conclusion

In this paper, the construction phase of a Greedy Randomized Adaptive Search Procedure (GRASP) was used to build initial solutions for the Multi-Vehicle Profitable Pickup and Delivery Problem (MVPPDP). The algorithm uses the concept of a Restricted Candidate List (RCL) to combine between random and greedy properties, which can help in the diversification of the search, and is especially beneficial for population-based metaheuristics. The performance of our algorithm was compared with two construction heuristics that were previously used to build the initial solutions of the MVPPDP. The results proved the effectiveness of the proposed method on small-medium sized instances, where eight new solutions were obtained for the MVPPDP. Also, the GRASP method had a better performance than one of the other construction heuristics on 11 test instances. Nevertheless, the proposed method is not currently able to produce good results on large-sized instances (500 and 1000 customers). This is possibly due to decreasing the number of iterations because of time limitations. Future work will try to improve the performance of the algorithm by optimizing its runtime. In addition, a second phase may be added to try to insert un-visited customers to increase the profit of the company and improve the quality of the initial solutions. Finally, an improvement phase, using a selected population-based metaheuristic, will be added to improve the initial constructed solutions by the GRASP based method.

References

[1] Toth, Paolo and Vigo, The vehicle routing problem. SIAM, 2002.

[2] R. F. Parragh, Sophie N and Doerner, Karl F and Hartl, “A survey on pickup and delivery problems,” J. für Betriebswirtschaft, vol. 58, no. 1, pp. 21–51, 2008.

[3] J. Saha, “An algorithm for bus scheduling problems,” J. Oper. Res. Soc., vol. 21, no. 4, pp. 463–474, 1970.

[4] U. Y. ¨ uceer and A. ¨ O. ¨ ,a, “A truck loading problem,” Comput. Ind. Eng., vol. 58, no. 4, pp. 766–773, 2010.

[5] M.-W. Yan, Shangyao and Wang, Sin-Siang and Wu, “A model with a solution algorithm for the cash transportation vehicle routing and scheduling problem,” Comput. Ind. Eng., vol. 63, no. 2, pp. 464–473, 2012.

[6] P. H. Barnhart, Cynthia and Jin, Hong and Vance, “Railroad blocking: A network design application,” Oper. Res., vol. 48, no. 4, pp. 603–614, 2000.

[7] R. A. Díaz-Parra, O., Ruiz-Vanoye, J. A., Bernábe Loranca, B., Fuentes-Penna, A., & Barrera-Cámara, “A survey of transportation problems,” J. Appl. Math., vol. 2014, 2014.

[8] J. R. F. Cornillier, F. F. Boctor, G. Laporte, “An exact algorithm for the petrol station replenishment problem,” J. Oper. Res. Soc., vol. 59, no. 5, pp. 607–615, 2008.

[9] R. C. C. Zhu, J. Q. Hu, F. Wang, Y. Xu, “On the tour planning problem,” Ann. Oper. Res., vol. 192, pp. 67–86, 2012.

[10] D. Kodjak, “Policy discussion--heavy-duty truck fuel economy,” 10th Diesel Engine Emiss. Reduct., 2004.

[11] M. Gansterer, R. F. Hartl, and R. Vetschera, “The cost of incentive compatibility in auction-based mechanisms for carrier collaboration,” Networks, Jun. 2018.

[12] C. Lin, K. L. Choy, G. T. S. Ho, S. H. Chung, and H. Y. Lam, “Survey of Green Vehicle Routing Problem: Past and future trends,” Expert Syst. Appl., vol. 41, no. 4, pp. 1118–1138, Mar. 2014.

[13] J. Privé, J. Renaud, F. Boctor, and G. Laporte, “Solving a vehicle-routing problem arising in soft-drink distribution,” J. Oper. Res. Soc., vol. 57, no. 9, pp. 1045–1052, Sep. 2006.

[14] N. Aras, D. Aksen, and M. Tuğrul Tekin, “Selective multi-depot vehicle routing problem with pricing,” Transp. Res. Part C Emerg. Technol., vol. 19, no. 5, pp. 866–884, Aug. 2011.

[15] I. Gribkovskaia, G. Laporte, and A. Shyshou, “The single vehicle routing problem with deliveries and selective pickups,” Comput. Oper. Res., vol. 35, no. 9, pp. 2908–2924, Sep. 2008.

[16] C.-K. Ting and X.-L. Liao, “The selective pickup and delivery problem: Formulation and a memetic algorithm,” Int. J. Prod. Econ., vol. 141, no. 1, pp. 199–211, Jan. 2013.

[17] X.-L. Liao and C.-K. Ting, “An evolutionary approach for the selective pickup and delivery problem,” in Evolutionary Computation (CEC), 2010 IEEE Congress on, IEEE, 2010, pp. 1–8.

[18] W. Y. Ho, Sin C and Szeto, “GRASP with path relinking for the selective pickup and delivery problem,” Expert Syst. Appl., vol. 51, pp. 14–25, Jun. 2016.

[19] I. M. Coelho, P. L. A. Munhoz, L. S. Ochi, M. J. F. Souza, C. Bentes, and R. Farias, “An integrated CPU–GPU heuristic inspired on variable neighbourhood search for the single vehicle routing problem with deliveries and selective pickups,” Int. J. Prod. Res., vol. 54, no. 4, pp. 945–962, Feb. 2016.

[20] I. M. Coelho, P. L. A. Munhoz, M. N. Haddad, M. J. F. Souza, and L. S. Ochi, “A hybrid heuristic based on General Variable Neighborhood Search for the Single Vehicle Routing Problem with Deliveries and Selective Pickups,” Electron. Notes Discret. Math., vol. 39, pp. 99–106, Dec. 2012.

[21] B. P. Bruck, A. G. dos Santos, and J. E. C. Arroyo, “Hybrid metaheuristic for the single vehicle routing problem with deliveries and selective pickups,” in Evolutionary Computation (CEC), 2012 IEEE Congress on, IEEE, 2012, pp. 1–8.

[22] M. Gansterer, M. Küçüktepe, and R. F. Hartl, “The multi-vehicle profitable pickup and delivery problem,” OR Spectr., vol. 39, no. 1, pp. 303–319, Jan. 2017.

[23] M. G. Feo, Thomas A and Resende, “Greedy randomized adaptive search procedures,” J. Glob. Optim., vol. 6, no. 2, pp. 109–133, 1995.

[24] E.-G. Talbi, Metaheuristics: from design to implementation. John Wiley \& Sons, 2009.

[25] P. Priore-Moreno, R. Pino-Diez, C. Martínez-Carcedo, V. Villanueva-Madrileño, and I. Fernández-Quesada, “Application of GRASP methodology to vehicle routing problem,” in Proceedings on the International Conference on Artificial Intelligence (ICAI), 2012, p. 1.

[26] A. Layeb, M. Ammi, and S. Chikhi, "A GRASP algorithm based on new randomized heuristic for vehicle routing problem," J. Comput. Inf. Technol., vol. 21, no. 1, pp. 35–46, 2013.

[27] C. Duhamel, Christophe and Lacomme, Philippe and Prins, Christian and Prodhon, "A GRASP×ELS approach for the capacitated location-routing problem," Comput. Oper. Res., vol. 37, no. 11, pp. 1912–1923, Nov. 2010.

[28] Y. Marinakis, "Multiple Phase Neighborhood Search-GRASP for the Capacitated Vehicle Routing Problem," Expert Syst. Appl., vol. 39, no. 8, pp. 6807–6815, Jun. 2012.

[29] M. HADDAD, "AN EFFICIENT HEURISTIC FOR ONE-TO-ONE PICKUP AND DELIVERY PROBLEMS," Fluminense Federal Uneversity, 2017.

[30] G. Gutiérrez-Jarpa, G. Desaulniers, G. Laporte, and V. Marianov, "A branch-and-price algorithm for the Vehicle Routing Problem with Deliveries, Selective Pickups and Time Windows," Eur. J. Oper. Res., vol. 206, no. 2, pp. 341–349, Oct. 2010.

[31] S. Qiu, Xiaoqiu and Feuerriegel, "A MULTI-VEHICLE PROFIT-MAXIMIZING PICKUP AND DELIVERY SELECTION PROBLEM WITH TIME WINDOWS."

[32] A. Baniamerian, M. Bashiri, and R. Tavakkoli-Moghaddam, "Modified variable neighborhood search and genetic algorithm for profitable heterogeneous vehicle routing problem with cross-docking," Appl. Soft Comput., vol. 75, pp. 441–460, 2019.

[33] M. Wen, J. Larsen, J. Clausen, J. Cordeau, and G. Laporte, "Vehicle routing with cross-docking," J. Oper. Res. Soc., vol. 60, no. 12, pp. 1708–1718, 2009.

[34] C. Archetti, M. G. Speranza, and D. Vigo, "Chapter 10: Vehicle Routing Problems with Profits," in Vehicle Routing: Problems, Methods, and Applications, Second Edition, 2014, pp. 273–297.

[35] D. Toth, Paolo and Vigo, Vehicle routing : problems, methods, and applications. Society for Industrial and Applied Mathematics, 2014.

[36] P. H. Vidal, Thibaut and Maculan, Nelson and Ochi, Luiz Satoru and Vaz Penna, "Large neighborhoods with implicit customer selection for vehicle routing problems with profits," Transp. Sci., vol. 50, no. 2, pp. 720–734, 2015.

[37] C. Archetti, N. Bianchessi, and M. G. Speranza, "Optimal solutions for routing problems with profits," Discret. Appl. Math., vol. 161, no. 4–5, pp. 547–557, Mar. 2013.

[38] M. G. Archetti, Claudia and Feillet, Dominique and Hertz, Alain and Speranza, "The capacitated team orienteering and profitable tour problems," J. Oper. Res. Soc., vol. 60, no. 6, pp. 831–842, 2009.

[39] R. Lahyani, M. Khemakhem, and F. Semet, "Heuristics for rich profitable tour problems," in Modeling, Simulation and Applied Optimization, 2013 5th International Conference on, 2013, pp. 1–3.

[40] M. Küçüktepe, "A General Variable Neighbourhood Search Algorithm for the Multi-Vehicle Profitable Pickup and Delivery Problem," University of Vienna, 2014.