Multi-Depots Vehicle Routing Problem with Simultaneous Delivery and Pickup and Inventory Restrictions: Formulation and Resolution

BOUANANE Khaoula[1], BENADADA Youssef[2], BENCHEIKH Ghizlane[3]
Smart Systems Laboratory, Rabat IT Center, ENSIAS, Mohammed V University, Rabat, Morocco[1, 2]
Faculté des Sciences Juridiques Economiques et Sociales, Moulay Ismail University, Meknes, Morocco[3]

***Abstract*—Reverse logistics can be defined as a set of practices and processes for managing returns from the consumer to the manufacturer, simultaneously with direct flow management. In this context, we have chosen to study an important variant of the Vehicle Routing Problem (VRP) which is the Multi-Depot Vehicle Routing Problem with Simultaneous Delivery and Pickup and Inventory Restrictions (MD-VRPSDP-IR). This problem involves designing routes from multiple depots that simultaneously satisfy delivery and pickup requests from a set of customers, while taking into account depot stock levels. This study proposes a hybrid Genetic Algorithm which incorporates three different procedures, including a newly developed one called the K- Nearest Depot heuristic, to assign customers to depots and also the Sweep algorithm for routes construction, and the Farthest Insertion heuristic to improve solutions. Computational results show that our methods outperform the previous ones for MD-VRPSDP.**

Keywords*—Reverse logistic; inventory restrictions; VRPSDP; multi-depots version; Genetic Algorithm

I. Introduction

Our current production system is based on the use and processing of raw materials into finished products. The completion of this production cycle is through the final disposal or reuse of these products. This is how the last few years have seen the appearance of the emerging research problem: reverse logistics. Thus, issues related to efficiency and environmental effectiveness will have to be taken into consideration in the areas of business strategy, planning of the operation itself, as well as control of the distribution flows, in order to implement a reverse logistics of the product. Unlike the delivery of products to a customer, reverse logistics of returns is to manage flows from consumer to the manufacturer. The new challenges for researchers are to minimize transportation costs to make the reuse of products and materials more profitable than their elimination.

Reverse logistics can be defined as a set of practices designed to manage the return of products from customers to the manufacturer for repair, recycling or disposal at the lowest possible cost. To do this, a simple VRP is not adequate, it must be adapted to situations where vehicles can deliver end products and pick up returns simultaneously. The variant of the VRP most suited to this situation is the VRPSDP (Vehicle Routing Problem with Simultaneous Delivery and Pickup), where each customer is associated with delivery and pickup requests that must be made simultaneously.

In practice, applications of the VRPSDP are found especially within a reverse logistics context [1]. For instance, in the distribution system of food market chains [2], or in the urban public transport systems [3].

In this problem, each depot has a homogeneous vehicle fleet that must ensure the satisfaction of known delivery and pickup requests of a set of customers. Each customer must be visited once, this means that the vehicle upon arrived at the customer who must serve, the delivery and collection must be done at the same time. We assume that each depot is associated with a stock of products to be delivered and another for products collected from customers. The objective is to minimize the total distance traveled as well as the number of required vehicles while ensuring that the capacity constraints of vehicles and depots are not violated.

The MD-VRPSDP-IR is a very complex problem because it combines both the Multi-Depot version of the VRPSDP which is an NP hard problem and additional constraints such as inventory restrictions. To our knowledge, there is not yet a work in the literature that is interested in the interaction between these constraints: multiple depots, simultaneous delivery and pickup and inventory restrictions.

To avoid any confusion between certain variants of the VRP, we would like to clarify that the problem treated in this work is an extension of the VRPB (Vehicle Routing Problem with Backhauls), where the origin and the destination of all products delivered and picked up from customers are the depot. Unlike the VRPPD (Vehicle Routing Problem with Pickup and Delivery), where the interchanges of goods are made between customers.

In this paper, we propose a mathematical formulation as a Mixed Integer Linear Program (MILP), which aims to minimize both total travel cost and number of required vehicles. We implement the model in CPLEX to solve small problem instances optimally. Then, we propose a Hybrid Genetic Algorithm in which we use three different procedures to assign customers to depots, and then we embed the Sweep algorithm to construct routes for each depot and the Farther Insertion heuristic to improve the solution. The proposed heuristics are more complicated than those used for VRP involving only deliveries or pickups. The presence of combined delivery and pickup demands in our problem, and also restrictions on depot capacities mean that additional tests are required to preserve feasibility. The quality of our method is shown by tests on well-known benchmark instances of MD-VRPSDP, which is special case of our problem and by comparison with optimal results, obtained by CPLEX as well as reported result for existing heuristics.

In Section 2, a rich literature review is detailed. In Section 3, mathematical formulation and notations of MD-VRPSDP-IR are presented. Details of the proposed GAs are introduced in Section 4. In Section 5, the performance of the proposed GAs is examined by solving Gillett and Johnson's test problems and a computational example is represented with parameter settings. Section 6 concludes the paper with future works.

II. Related Literature Review

In this section, we propose to briefly discuss the literature of the VRPSDP and its Multi-Depot version, since we have not found a literature related to the MD-VRPSDP-IR.

VRPSDP is firstly introduced by [4]; he developed a model and a Cluster First – Route Second approach for the VRPSDP, and applied his model and the solution he proposed on a real case of a public library distribution system. Author in [1] discussed the importance of VRPSDP in the reciprocal logistic activities. He developed an Insertion-Based heuristic that use different criteria (travel distance, residual capacity and radial surcharge) to solve the problem. Afterward, many authors have become interested in the VRPSDP and its variants, and have developed several heuristics and metaheuristics to solve it. We mention here the most recent articles dealing with these problems. Author in [5] introduce the notion of Handling Cost in the VRPSDP; the items on the vehicle obeys the last-in-first-out policy, so handling operations are required if the delivery items are not the last loaded ones. They propose an Adaptive Large Neighborhood Search (ALNS) metaheuristic in which they embed the handling policies. Reference [6] deals with a special VRPSDP where three-dimensional loading constraints are assumed furthermore time windows constraints. To avoid any reloading effort, they consider two loading approaches of vehicles: loading from the backside with separation between delivery and pickup sections and loading at the long side. There method is a hybrid of an extended ALNS and conventional packing heuristics. Authors in [7] and [8] treat green VRPSDP; they propose models that minimize the cost of fuel consumption and pollutant emissions of vehicles. To solve his model, [7] uses Genetic Algorithms, which she hybrids with Sweep heuristic, and the Nearest Neighbor Heuristic to generate an initial population, and then Iterated Swap Procedure improves the chromosomes. Whereas, [8] applies the fuzzy approach when both pickup and delivery demands are uncertain, and they propose an ALNS heuristic. Reference [9] deals with a variant of the basic VRPSDP including the multiple trips and time windows characteristics. They propose a solution approach based on Tabu Search, with the sequential insertion algorithm to construct an initial solution. Other heuristics and metaheuristics have been proposed for different variants of VRPSDP; the most recent ones were published by [10]-[20].

Concerning the Multi-Depot version, we found in the literature that few studies. Starting with [21] who deal with the Multi-Depot case of simultaneous backhauling problems, their method consists of extending the classical Insertion-Based Heuristic to allow to the algorithm to insert more than one backhaul at a time. This method perform well for a small number of backhauls, but if this number increase, computational complexity increases rapidly. In [22], the author developed an integrated heuristic that treat linehaul and backhaul customers similarly.

Author in [23] proposed four Saving Based Algorithms for the Multi-Depot version of VRPSDP: Partition Based Algorithms, Nearest Customer Algorithm and two different Saving Based Algorithms. Author in [24] was the first to develop metaheuristics for the MDVRPSDP. The algorithm framework used in their procedure in based on the Iterated Local Search (ILS) with an Adaptive Neighborhood Selection mechanism (ANS). At first, they assign customers to their nearest depot for creating an initial solution, after, they apply Saving Algorithm to each depot. They used different structural neighborhood methods for improving and perturbation steps of ILS.

An Improved Genetic Algorithm (IGA) is developed in [25] to solve the MD-VRPSDP with Soft Time Windows. Firstly, customers are assigned to their nearest depot and initial solutions constructed by Scanning Algorithm. A greedy based strategy is used for cutting and merging routes. Finally, for optimizing and adjusting the feasible solutions, they used three neighborhood search methods and 3-opt local search.

To assign customers to depots, [26] employed the Minimum Cost Flow problem previously solved by a graph algorithm. In this way, the original problem becomes a set of several Single-Depot problems. After this, the Weber Basis Saving method is developed to construct the initial solution of each sub-problem. Finally, improvement phase is assured by the Modified Tabu Search.

At this point, we want to note that in the works cited above, concerning the Multi-Depot version of the VRPSDP, the authors assign customers to their nearest depots at first, then proceed to resolve each VRPSDP as a sub-problem. Our contribution in this paper is that we explore new ways to assign customers to depots while keeping a margin of randomness. More details are given in Section 4.

III. Problem Description and Formulation

The MD-VRPSDP-IR is the problem of construction routes for homogeneous vehicle fleets, which originate from several depots, visit a set of customers assigned to each depot, and return to the departure depot. The inventory restrictions constraint is reflected in the fact that each depot has two storage areas, one for the products that will be delivered to customers (SD: Stock for Deliveries), and the other for the products collected from customers (SP: Stock for Pickups). However, all goods transported must be taken from depots, and any collected returns must be sent to depots. The constraint assure that a customer can only be served if his delivery request is available in SD and his collection request has enough space to be stored in SP. Fig. 1 exemplifies the MD-VRPSDP-IR with 2 depots and 14 customers. The brackets above the customers contain delivery and pickup demands, and those above the depots represent depot capacities of delivery and pickup demands.

Let $G(V, E)$ be a graph, where V is the vertex set and $E = \{(i, j): i \neq j\}$ is the edge set. The vertex set V is partitioned into two subsets $V_d = \{1, \dots, m\}$ and $V_c = \{m+1, \dots, m+n\}$, which represent the set of depots and the set of customers, respectively. Each vertex $j \in V_c$ has a non-negative pickup demand $\boldsymbol{P_j}$, delivery demand $\boldsymbol{D_j}$ and a service time $\boldsymbol{t_j}$. Furthermore, in the depot vertex $j \in V_d$, there are no demands and service times $P_j = D_j = t_j = 0$. For all $i,\ j \in V$, a distance matrix $\boldsymbol{d_{ij}}$ and a travel time matrix $\boldsymbol{t_{ij}}$ are associated with E. A set K_d of identical vehicles of capacity $\boldsymbol{Q}$ is available at each depot $d \in V_d$. The optimal distribution of goods between depots and customers depends on inventory levels in depots, therefore each depot d has maximum capacities $\boldsymbol{SD_d}$ and $\boldsymbol{SP_d}$ for delivery and pickup requests, respectively.

A. Notions

1) Sets

V_d : Set of all depots.

V_c : Set of all customers.

V : Set of all nodes, $V = V_c \cup V_d$

K_d : Set of vehicles associated with depot d.

K : Set of all vehicles, $K = \cup_d K_d$

2) Indices

D : Depot

K : Vehicle

I : Start node

J : Destination node

3) Parameters

D_j: Delivery demand of customer ***j***

P_j : Pick-up demand of customer ***j***

t_j : Service time of customer ***j***

t_{ij}: Travel time of a vehicle from node ***i*** to node ***j***

d_{ij}: Distance between node $\boldsymbol{i \in V}$ and $j \in V$

c_{ij}: Travel cost of a vehicle from node $\boldsymbol{i \in V}$ to node $j \in V$

$C_{Mil\$}$: Mileage cost of a vehicle.

C_k: Cost of operating vehicle ***k***.

Q: The maximum capacity of a vehicle.

T : The maximum working time allowed for a vehicle during a working day.

SD_d : The maximum stock of delivery product in depot ***d***.

SP_d : The maximum stock of picked up product in depot ***d***.

M : Large number.

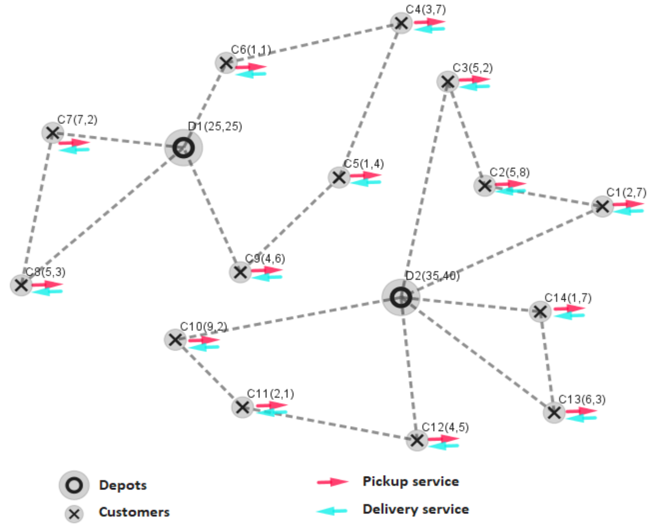


Fig. 1. Illustration of MD-VRPSDP-IR.

4) Decision variables

x_{ij}^k: $\boldsymbol{x_{ij}^k = 1}$ when vehicle $\boldsymbol{k}$ travels directly from node $\boldsymbol{i \in V}$ to node $j \in V$. $\boldsymbol{x_{ij}^k = 0}$ otherwise.

L_j : Load of vehicle after having serviced customer $j \in V_c$.

u_j : Variable used to prohibit sub tours; can be interpreted as position of node $j \in V_c$ in the route.

$IL_k = \sum_{i \in V} \sum_{j \in V_c} D_j\, x_{ij}^k \quad (k \in K)$: Load of vehicle $k \in K$ when leaving the depot (Initial Load).

$FL_k = \sum_{i \in V_c} \sum_{j \in V} P_i\, x_{ij}^k \quad (k \in K)$: Load of vehicle $k \in K$ after visiting last customer (Final Load).

B. Mixed Integer Linear Programming Model for MD-VRPSDP-IR

The objective of the proposed mathematical model is to minimize the total transportation cost z due to the weighted sum of the total distance traveled of all vehicles and the cost related to the number of required vehicles, where w_d and w_R are the weight factors of the total distance traveled and the number of used vehicles, respectively, and α and β are conversion factors from distance to cost (unit: Dh/km) and from number of vehicles to cost (unit: $Dh/vehicle$), respectively.

Minimize total cost z:

$$z = w_d \,.\, \alpha \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} d_{ij}\, x_{ij}^k \; + w_R \,.\, \beta \sum_{k \in K} \sum_{i \in V_d} \sum_{j \in V_c} x_{ij}^k \quad (1)$$

Constraints of the problem are given below:

$$\sum_{k \in K} \sum_{i \in V} x_{ij}^k = 1 \qquad (j \in V_c) \qquad (2)$$

$$\sum_{i \in V} x_{is}^k = \sum_{j \in V} x_{sj}^k \qquad (k \in K, s \in V_c) \qquad (3)$$

$$\sum_{j \in V_c} x_{ij}^k = \sum_{l \in V_c} x_{li}^k \qquad (k \in K, i \in V_d) \qquad (4)$$

$$\sum_{i \in V_d} \sum_{j \in V_d} x_{ij}^k = 0 \qquad (k \in K) \qquad (5)$$

$$\sum_{j \in V_c} t_j \sum_{i \in V} x_{ij}^k + \sum_{i \in V} \sum_{j \in V_c} t_{ij}\, x_{ij}^k \le T \qquad (k \in K) \qquad (6)$$

$$u_j \geq u_i + 1 - (n+m)\left(1 - \sum_{k \in K} x_{ij}^k\right) \quad (i, j \in V_c) \qquad (7)$$

Vehicle load constraints

$$L_j \geq IL_k - D_j + P_j - M\left(1 - x_{dj}^k\right) \quad (k \in K, d \in V_d, j \in V_c) \qquad (8)$$

$$L_j \geq L_i - D_j + P_j - M\left(1 - \sum_{k \in K} x_{ij}^k\right) \quad (i, j \in V_c) \qquad (9)$$

$$IL_k \leq Q \qquad (k \in K) \qquad (10)$$

$$L_j \leq Q \qquad (j \in V_c) \qquad (11)$$

Inventory restrictions constraints

$$\sum_{k \in K_d} IL_k \leq SD_d \qquad (d \in V_d) \qquad (12)$$

$$\sum_{k \in K_d} FL_k \leq SP_d \qquad (d \in V_d) \qquad (13)$$

Integrality constraints

$$x_{ij}^k \in \{0,1\} \qquad (i, j \in V, k \in K)$$

$$u_j \geq 0 \qquad (j \in V_c)$$

Constraints (2) ensure that each customer is visited exactly once by exactly one vehicle. Flow conservation is ensured by constraint (3). Constraints (4) required that each vehicle starts and ends its route at the same depot. Constraints (5) impose that a vehicle cannot travel between two depots. Constraints (6) ensure that the total duration of each route (including travel time and service time) does not exceed a pre-set limit. Constraints (7) eliminate the sub-tours to ensure that the solution is connected. After visiting the first customer, the vehicle load is calculated by constraint (8) and after leaving other customers, the vehicle load is calculated by constraint (9). Constraints (10) and (11) ensure that the vehicle capacity is respected at each section of the route. Constraints (12) and (13) require that stock levels in each depot are not surpassed.

A necessary but not sufficient condition to have feasible solutions is to ensure that all customers can be served; this is verified by the following constraints:

$$\sum_{j \epsilon V_c} D_j \leq \sum_{d \epsilon V_d} SD_d \quad , \qquad \sum_{j \epsilon V_c} P_j \leq \sum_{d \epsilon V_d} SP_d$$

However, it is not worth adding them to the mathematical model, because we can deduce them from the constraints (2), (12) and (13).

IV. Hybrid Genetic Approach

The MD-VRPSDP-IR is a NP-hard problem. As the problem instances increase in size, the exact solution methods become highly time-consuming. In recent years, GA has been applied successfully to a wide variety of hard optimization problems such as the classical VRP and its multi-depot version. The success is mainly due to its simplicity, easy operations, and great flexibility. These are the major reasons why we selected a GA as an optimization tool in this paper.

The problem studied in this work is an integration of two hard optimization problems: grouping and routing problems. A simple GA may not perform well in this situation. Therefore, the GA developed in this paper is hybridized with several heuristics to construct and improve the solutions. Fig. 2 shows the flowchart of three Hybrid Genetic Algorithms (GAs). The difference between them is in the assignment of customers to depots: GA1 attribute customers randomly to depots, GA2 use the K-Nearest Depot heuristic to assign customers to depots considering the depot-customer distances, but also a random selection step and GA3 assign customers to their nearest depots.

A. Chromosome Representation

The permutation representation is used for genetic representation of the MD-VRPSDP-IR as shown in Fig. 3. A chromosome is built as an array with three rows: 1) customers, which are listed in the order in which they are visited; 2) depots, where customers are assigned depending on depot capacities; 3) vehicles required in each depot to satisfy all demands of customers assigned to this depot. Routes are determined depending on vehicles capacity. The number of customer nodes determines the length of the chromosome.

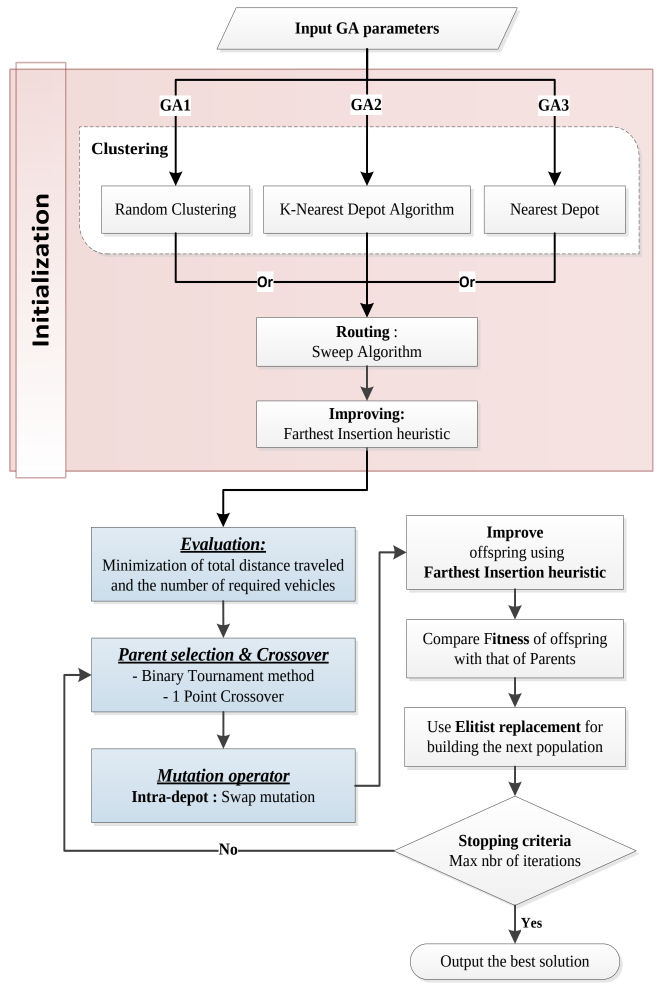


Fig. 2. The Flowchart of Gas.

B. Initial Population Construction

In this work, there are three phases to generate a feasible initial solution (Fig. 4). The first one is to assign customers to depots, that is, *the grouping problem*, for this, we use one of the three procedures mentioned above. The second phase is to perform, for each depot, a clustering of customers assigned to this depot and then determine a vehicle route for each cluster by using the Sweep algorithm, that is, *the routing problem*. The last phase consists of *improvement* of several routes already built, for this we use the Farthest Insertion Heuristic.

1) Grouping: It is worth to note that the grouping problem and the routing problem in the "cluster first, route second" approach are not independent. A bad assignment solution will result in routes of higher total cost (distance) than with a better assignment. The grouping procedures described in the following assign customers to depots so that the capacity of the depots is not exceeded.

Grouping can be done using one of the following three methods: 1) Attributing customers randomly to depots: we randomly choose a customer and then a depot, if the depot capacity is not yet reached, we assign this customer to this deposit, otherwise we choose another deposit and so on. 2) Using the K-Nearest Depot heuristic (See next paragraph). 3) Assigning customers to their nearest depots within the limit of stock availability in each depot.

2) The k-Nearest depot heuristic: We developed this algorithm to assign all customer to different depots based on the customer-depot distance, while keeping a random side in the procedure, as shown in Fig. 5. For each customer, we find the $\frac{m}{2}$ (where m is the number of depots) closest depots of this customer and who can serve it obviously. Then we randomly choose one of these depots to assign the customer. We first check the feasibility of this assignment, if the capacities of the depot allow this assignment, it is done, if not, we choose another deposit, and so on.

3) Routing: the sweep algorithm: The sweep algorithm belongs to the *Cluster First - Route Second* family. It begins by assigning to customers angular coordinates related to depot, and then scanning in the direction of increasing coordinates. In our paper, to order customers, we do not assign them polar coordinates, we use the order generated in the grouping phase.

Customers are added successively to a vehicle route following this order, and as soon as the capacity of the vehicle is reached, a new vehicle route is created and the process is repeated until all customers have been swept. Then, when all routes are formed, we execute the next phase.

4) Improving: the farthest insertion heuristic: After the construction is finished, routing costs can be reduced using a route improvement algorithm. In our improvement method, before validating a change, we must verify that the capacity of the vehicles performing the tours processed is respected in all points and that the change brings a gain in the cost of the solution.

In the FI heuristic, a route is constructed by progressively adding a customer one at a time until a complete route is formed. The part of the route that is already built remained unchanged during the tour construction process. The FI heuristic start with a route of two customers those are located farthest to one another. Then, an unvisited customer that is farthest to the route is selected. This customer is inserted between two consecutives customers that result in minimum increase of route cost.

Customers	5	2	10	14	9	11	12	3	6	7	4	8	13	1
Depots	1	2	2	2	1	2	2	2	1	1	1	1	2	2
Vehicles	2	3	5	4	2	5	5	3	2	1	2	1	4	3

Fig. 3. Chromosome Representation.

Customers	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Depots	1	2												

Grouping

Customers	7	4	12	8	5	11	10	3	6	2	14	9	13	1
Depots	1	2	2	2	1	2	2	2	1	1	1	1	2	2

Routing : Sweep algorithm

Customers	2	6	14	5	9	7	8	13	10	1	4	12	3	11
Depots	1	1	1	1	1	1	2	2	2	2	2	2	2	2
Vehicles	1	1	1	2	2	2	3	3	4	4	4	5	5	5

Improving : Farthest Insertion Heuristic

Customers	14	6	2	9	7	5	8	13	4	1	10	11	12	3
Depots	1	1	1	1	1	1	2	2	2	2	2	2	2	2
Vehicles	1	1	1	2	2	2	3	3	4	4	4	5	5	5

Fig. 4. Initialization of Gas.

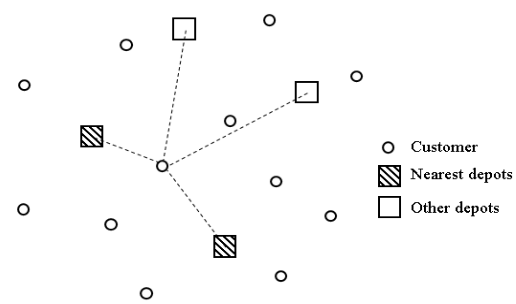


Fig. 5. The K-Nearest Depots Heuristic.

C. Fitness Function

Fitness function represents the method for the evaluation of individuals. Since each generated chromosome is a feasible solution, and our function combines route length with other parameter, that is the number of required vehicles, the fitness value of each chromosome is then calculated with weighted sum of all parameters [27]. This method requires adding the values of fitness functions together using weighted coefficients for each individual objective. That is, our multi-objective MD-VRPSDP-IR is transformed into a single-objective optimization problem, where the fitness function $F(x)$ of an individual x is returned as:

$$F(x) = \left(1 + \left(w_d \,.\, \alpha \,.\, \sum_{k \in R} D_k + w_R \,.\, \beta \,.\, |R|\right)\right)^{-1}$$

$$where \quad D_k = \sum_{i \in V} \sum_{j \in V} d_{ij} x_{ij}^k$$

w_d and w_R are weight parameters associated with the total traveled cost of all vehicles and the number of required vehicles, respectively. The weight values of the parameters used in this function were established empirically.

Fig. 6. Example of the 1X Crossover.

D. Parent Selection and Crossover

Parent selection is performed through a *binary tournament*, which twice randomly chooses two individuals from the population, and keeps the one with the highest value of fitness. This process is repeated until the required number of individuals is obtained.

In this paper, we use the One Point Crossover (1X). The crossing operator is applied just on the first two range of the chromosome; those of customers and depots, as shown in Fig. 6. Afterwards, routing and improving procedures are applied to the offspring to build the routes for each depot. To build the offspring, we first start with the row of customers; the first part of the first parent is copied, and then the elements of the second part of this parent are reordered in the order of appearance they have in the second parent. Afterwards, the allocation of customers to depots is done by respecting the depots capacities; for each client, we first check if it can be assigned to its initial depot (in the first parent), otherwise we choose another nearest depot, and so on.

E. Mutation

The mutation operator plays the role of a disruptive element; it explores a wider search space and allows maintaining the diversity of the next population, avoiding the algorithm to converge too quickly towards a local optimum. We employed the Swap mutation, which we applied as an intra-depot mutation that involves a single depot. Swap mutation is simple; it consists of randomly taking two genes (2 customers) from the chromosome and swapping them. If the offspring is not feasible, it is deleted.

V. Computational Results and Discussions

This section describes computational experiments carried out to study the performance of the proposed GAs. The algorithm is coded in C and run on a laptop computer with an Intel Core i7 2.9 GHz processor with 8 GB RAM, under the operating system Windows® 7. First, we compare the performance of GAs, which have the best results will be used in the tests that follow. Then, to validate the MILP model for the MD-VRPSDP-IR proposed in this paper, we compare our GA results with those obtained by CPLEX, for a small instance, through an illustrative example. To assess the effectiveness of the best GA, it is tested on its special case MD-VRPSDP, since we did not find reported results for MD-VRPSDP-IR. For this, we assume that depot capacities are infinite. And then we compare results obtained by the best GA with [21] and [23] for which there are reported results for the MD-VRPSDP, and are using the same data.

A. Benchmarks

For the numerical experiments, we adopt the data set provided by [21] as the tested instances. It includes 22 problem instances (2 to 5 depots, 50 to 249 customers) generated from 11 benchmark problems of [28] (the first 8 ones are provided from [29] and the last 4 ones from [30]). The 22 problem instances are partitioned as sets X and Y based on the difference of deliveries and pickups.

We use the method proposed by [21] and used by [23] for splitting the original demand into pickup and delivery demands. Let x_i and y_i denote the coordinates of customer i, and let D_i^{org} denote the demand for customer i in the original problem. The distance matrix is generated using the original coordinates and is calculated with Euclidean distance. However, D_i^{org} is split into delivery demand D_i and pickup demand P_i as follows:

$$D_i = r_i \times D_i^{org} \quad and \quad P_i = (1 - r_i) \times D_i^{org}$$

$$where \quad r_i = min\left(\frac{x_i}{y_i}, \frac{y_i}{x_i}\right)$$

In this way, set X of 11 instances is generated. The other set Y, likewise with 11 instances, is generated by exchanging the pickup and delivery demands in problem instances of set X. The basic characteristics of instances are shown in Table 1.

In addition to these characteristics, we will need the storage capacity SD of products to be delivered and the storage capacity SP of the collected products, for each depot and each instance. The SD and SP values used are created by ourselves and are compatible with the instance characteristics and the conditions of the problem. We assume that the values of SD and SP are equal for all the depots of the same instance. Depots' information is as Table 2 shows.

TABLE I. Basic Characteristics of Data Sets for the MD-VRPSDP

N° Inst.	n	d	Q	Depot coordinates
GJ1	50	4	80	(20,20) , (30,40) , (50,30) , (60,50)
GJ2	50	4	160	(20,20) , (30,40) , (50,30) , (60,50)
GJ3	75	5	140	(40,40) , (50,22) , (55,55) , (25,45) , (20,20)
GJ4	100	2	100	(35,20) , (35,50)
GJ5	100	2	200	(15,35) , (55,35)
GJ6	100	3	100	(15,20) , (50,20) , (35,55)
GJ7	100	4	100	(15,35) , (55,35) , (35,20) , (35,50)
GJ8	249	2	500	(-33,33) , (33,-33)
GJ9	249	3	500	(70,0) , (-50,60) , (-50,-60)
GJ10	249	4	500	(75,0) , (0,75) , (-75,0) , (0,-75)
GJ11	249	5	500	(70,0) , (40,-80) , (40,80) , (-60,20) , (-60,-20)

n: number of customers, d: number of depots, Q: vehicle capacity

TABLE II. DEPOT'S INFORMATION

N° Inst.	SD	SP	N° Inst.	SD	SP
GJ1X	120	85	GJ1Y	85	120
GJ2X	120	85	GJ2Y	85	120
GJ3X	170	120	GJ3Y	120	170
GJ4X	440	320	GJ4Y	320	440
GJ5X	440	320	GJ5Y	320	440
GJ6X	290	215	GJ6Y	215	290
GJ7X	215	160	GJ7Y	160	215
GJ8X	3050	3100	GJ8Y	3100	3050
GJ9X	2040	2070	GJ9Y	2070	2040
GJ10X	1530	1550	GJ10Y	1550	1530
GJ11X	1220	1240	GJ11Y	1240	1220

B. Parameter Settings

First, we employ GJ1X instance to determine the appropriate number of iterations (Nbr_Iter) and population size (Pop_Size) for GAs, we test combinations:

$$Pop_{Size} = \{50, 100, 150, 200\}$$

$$Nbr\_Iter = \{300, 500, 1000, 5000\}$$

Results of several iterations are summarized in Table 3. For each combination, we run the program 30 times, the best objective function value and the average of all objective function values are summarized in column I and II, respectively. The computation time is given as average CPU times (s).

From these results, considering objective function values, the best solutions are given by the combination 500-5000 (Pop_Size-Nbr_Iter) as well as by the combination 200-300. However, combination 200-300 is preferable when considering also CPU time; it has a much less important CPU time than the combination 500-5000. Therefore, we use the combination 200-300 for Instances GJ1 to GJ7 and the combination 500-5000 for instances GJ8 to GJ11.

The other parameters used in GAs are crossover rate $p_c = 0.7$ and mutation rate $p_m = 0.01$. To obtain these values, we proceeded in the same way as for the population size and the number of iterations; we test combinations of $p_c = \{0.5, 0.6, 0.7, 0.8\}$ and $p_m = \{0.01, 0.05, 0.1\}$, the same instance GJ1X is employed to test them by changing the value of one parameter while keeping the other fixed. These values are then used in all other tests.

C. Experiments and Results

1) Comparison of GAs performances: A computational study is carried out to compare GA1 with random assignment of customers to depots, GA2 using the K-ND heuristic and GA3 which assign customers to the nearest depot. Table 4 reports the best solutions for the MD-VRPSPD-IR. To obtain the routing cost (Routing $) without taking into account the cost of using vehicles, we set the conversion factors at α=1 and β=0. After, we calculate the total transportation cost (Trans $) considering the number of used vehicles using the conversion factors α=1 and β=100, as follows:

$$Trans\ \$ = (\alpha * Routing\ \$) + (\beta * Nbr\ of\ Vehicles)$$

By comparing the routing costs, we find that the results given by GA1 are very high, and therefore are not competitive with those of GA2 and GA3. As for CPU time, it undergoes an insignificant change. GA2 gives better results than GA3 (in most cases). GA2 is also preferable when considering the number of required vehicles; it is usually smaller for GA2 than for GA3. We opted for a weighted sum of the routing cost and the number of used vehicles to compare the performance of GAs. It is found that the performance of GA2 is superior to that of GA3 in terms of total cost of transportation within nearly equal average computational time. The best solutions generated by GA2 are much better than those generated by GA3, this is due to the fact that GA2 incorporates the K-ND heuristic, which affects customers to depots taking into account the depot-customer distances, but also leaves a side of random. If we assign customers to the nearest depot, the assignments will always be the same for a given instance, and this will decrease the performance of the algorithm because it prevents it from exploring more, and thus excludes much solutions.

TABLE III. COMPUTATIONAL RESULTS FOR COMBINATIONS OF POPULATION SIZE AND NUMBER OF ITERATIONS.

Pop_Size	Nbr_Iter	I	II	CPU
50	300	382	419	0,12
100	300	386	407	0,13
150	300	377	399	0,13
200	300	**355**	**388**	**0,16**
50	500	396	414	0,17
100	500	373	410	0,18
150	500	383	400	0,16
200	500	382	398	0,20
50	1000	393	410	0,23
100	1000	389	408	0,24
150	1000	370	395	0,26
200	1000	364	397	0,31
50	5000	391	418	0,75
100	5000	376	401	0,86
150	5000	371	393	0,98
200	5000	**352**	**383**	**1,52**

TABLE IV. COMPARISON OF GAS PERFORMANCES FOR MD-VRPSDP-IR

Instance	GA1				GA2				GA3			
	Total $	Routing $	Nbr_Veh	CPU	Total $	Routing $	Nbr_Veh	CPU	Total $	Routing $	Nbr_Veh	CPU
GJ1X	1309	509	8	0,35	1079	279	8	0,35	1189	389	8	0,45
GJ2X	721	321	4	0,37	565	165	4	0,28	605	205	4	0,41
GJ3X	1336	536	8	0,38	1020	220	8	0,32	1337	437	9	0,35
GJ4X	2839	1639	12	0,44	2494	1294	12	0,36	2520	1320	12	0,41
GJ5X	1539	939	6	0,53	1308	708	6	0,42	1334	734	6	0,42
GJ6X	2623	1423	12	0,44	1974	874	11	0,41	2223	1023	12	0,49
GJ7X	2559	1359	12	0,45	1981	781	12	0,38	2156	956	12	0,41
GJ8X	6487	4587	19	3,04	5417	3517	19	3,15	5374	3474	19	3,19
GJ9X	6327	4427	19	3,06	4895	3195	17	2,85	5295	3395	19	3,06
GJ10X	6258	4358	19	3,01	4905	3005	19	3,28	5282	3382	19	3,17
GJ11X	6059	4159	19	2,93	4707	2907	18	3,11	5199	3299	19	3,38
Average	3460	2205	12,5	1,36	2759	1540	12,2	1,36	2956	1692	12,6	1,43
GJ1Y	1346	546	8	0,42	1083	283	8	0,37	1105	405	7	0,39
GJ2Y	734	334	4	0,35	548	148	4	0,29	613	213	4	0,36
GJ3Y	1354	554	8	0,35	1039	239	8	0,45	1367	467	9	0,37
GJ4Y	2858	1658	12	0,37	2486	1286	12	0,38	2528	1328	12	0,39
GJ5Y	1587	987	6	0,43	1302	702	6	0,33	1341	741	6	0,42
GJ6Y	2695	1495	12	0,39	1935	835	11	0,41	2217	1017	12	0,46
GJ7Y	2472	1272	12	0,41	1993	793	12	0,42	2176	976	12	0,63
GJ8Y	6451	4551	19	2,91	5389	3489	19	2,89	5388	3488	19	3,37
GJ9Y	6389	4489	19	2,92	4878	3178	17	2,97	5355	3355	20	3,43
GJ10Y	6312	4412	19	3,04	4860	2960	19	3,24	5287	3387	19	2,96
GJ11Y	6204	4304	19	2,97	4711	2911	18	3,19	5186	3286	19	3,42
Average	3491	2237	12,5	1,32	2748	1529	12,2	1,36	2960	1697	12,6	1,47

It is very important to note that the value assigned to the conversion factor β is set arbitrarily to 100 (a small value) just to show that the number of required vehicles in each solution is as important as the routing cost, and may even be larger when the value of β increases, which is the case in reality. That said, when the value of β increases, it directly and significantly affects the total cost of transportation. You can easily notice that if we increase the value of the conversion factor β, the results will switch quickly to a much higher performance for GA2 than for GA3, because in most instances, GA2 uses fewer vehicles than GA3, which proves the efficiency and strength of the developed K-ND heuristic.

2) Comparison with CPLEX: We use an illustrative example, with 2 depots and 12 customers, to compare the results obtained by CPLEX with those of GA2. Location of depots and customers and delivery and pick-up demands of customers are shown in Figs. 7 and 8, respectively.

Vehicle capacity is set at 80 and depot capacities are set at SD_1= SD_2=100 and SP_1=SP_2=50. To obtain the routing cost, conversion factors are set at α=1 and β=0. Results are summarized in Table V and illustrated in Fig. 9. Four vehicles served 12 customers, 2 for each depot.

We can easily notice that the results obtained by the algorithm developed in this paper are very close to the optimal value obtained by CPLEX solver, which uses branch and bound algorithm for solving MILP models. In addition, the proposed algorithm gives better solutions within significantly shorter time frame.

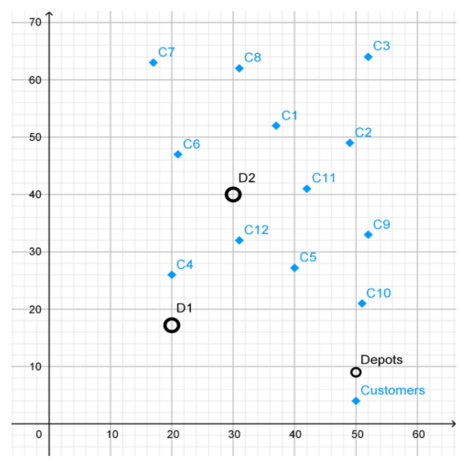


Fig. 7. Locations of Depots and Customers.

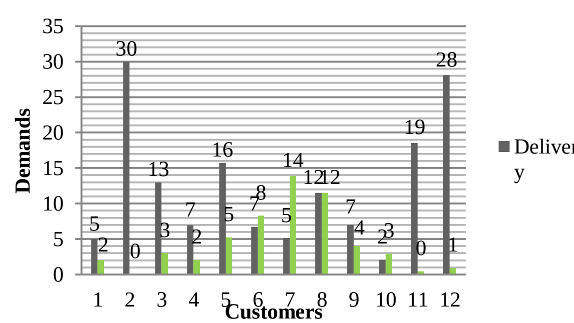
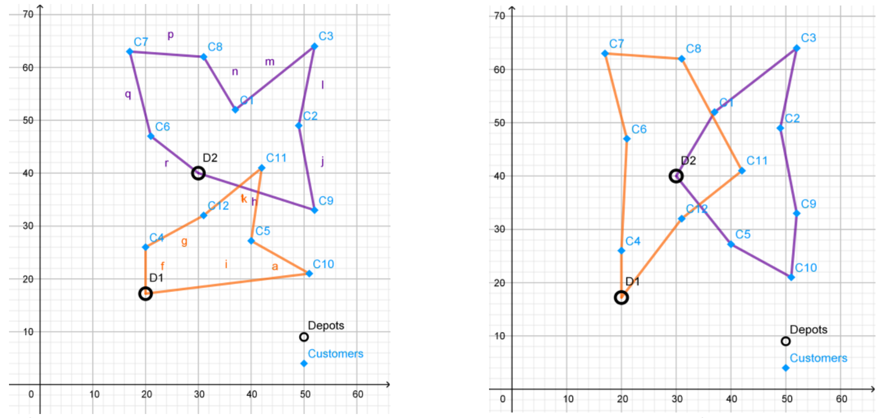


Fig. 8. Delivery and Pickup Demands of Customers.



Routes for solution obtained by CPLEX | Routes for solution obtained by GA

Fig. 9. Illustration of Results for Instance with 2 Depots and 12 Customers.

TABLE V. COMPARISON OF RESULTS OF CPLEX AND GA

	Routes	**Routing $**	**CPU**
CPLEX	{D1 - C4 - C12 - C11 - C5 - C10 - D1}	221	24min 15s
	{D2 - C9 - C2 - C3 - C1 - C8 - C7 - C6 - D2}		
GA2	{D1 - C4 - C6 - C7 - C8 - C11 - C12 - D1}	222,3	0,09 s
	{D2 - C1 - C3 - C2 - C9 - C10 - C5 - D2}		

3) Computational results and performance analysis: The objective is to minimize the weighted sum of the travel distances and the number of required vehicles. We assume that depot capacities are infinite. To calculate the total transportation cost, we set conversion factors α equal to 1 and β equal to 100. Results are reported in Table 6.

Unlike the results obtained for the MD-VRPSDP-IR, and by comparing the routing costs, we find that GA3 gives better results than GA2 (in most cases). However, GA2 is preferable when considering the number of required vehicles. Consequently, the performance of GA2 remains higher to that of GA3 in terms of total cost of transportation, even though its routing cost is slightly worse than that of GA3.

TABLE VI. GA2 AND GA3 PERFORMANCES FOR MD-VRPSDP

Instance	GA2					GA3			
	Total $	Routing $	Nbr_Veh	CPU		Total $	Routing $	Nbr_Veh	CPU
GJ1X	1206	406	8	0,36		1273	373	9	0,39
GJ2X	531	131	4	0,36		625	125	5	0,31
GJ3X	1254	454	8	0,38		1446	446	10	0,41
GJ4X	2054	854	12	0,41		2102	902	12	0,47
GJ5X	1315	715	6	0,42		1427	727	7	0,36
GJ6X	2398	1098	13	0,39		2475	1175	13	0,42
GJ7X	2092	892	12	0,43		2331	1031	13	0,43
GJ8X	5406	3506	19	2,76		5451	3451	20	2,82
GJ9X	5332	3332	20	2,79		5216	3216	20	2,87
GJ10X	4778	2878	19	2,77		4861	2861	20	3,2
GJ11X	4221	2321	19	2,85		4272	2272	20	3,82
Average	2781	1508	12,7	1,27		2862	1507	13,5	1,41
GJ1Y	1052	352	7	0,37		1309	409	9	0,36
GJ2Y	539	139	4	0,34		629	129	5	0,37
GJ3Y	1219	419	8	0,45		1457	457	10	0,37
GJ4Y	2032	832	12	0,38		2325	1125	12	0,42
GJ5Y	1338	738	6	0,42		1415	715	7	0,38
GJ6Y	2197	997	12	0,38		2251	951	13	0,38
GJ7Y	2025	825	12	0,45		2319	1019	13	0,39
GJ8Y	5417	3517	19	2,82		5471	3471	20	2,87
GJ9Y	5395	3395	20	2,95		5316	3316	20	2,73
GJ10Y	4883	2983	19	2,92		4881	2881	20	3,95
GJ11Y	4377	2477	19	3,59		4412	2412	20	3,05
Average	2770	1516	12,5	1,37		2890	1535	13,5	1,39

TABLE VII. COMPARISON OF THE AVERAGE RESULTS FOR THE MD-VRPSDP

Instances	Salhi and Nagy (1999)					Gajpal and Abad (2009)					GA2			
	Total $	Routing $	Nbr_Veh	CPU		Total $	Routing $	Nbr_Veh	CPU		Total $	Routing $	Nbr_Veh	CPU
GJ1X	2074	674	14	0,2		-	541	-	0,08		1206	406	8	0,36
GJ2X	1196	596	6	2,3		-	492	-	0,08		531	131	4	0,36
GJ3X	2034	734	13	1,5		-	638	-	0,26		1254	454	8	0,38
GJ4X	2993	1193	18	1,6		-	932	-	0,61		2054	854	12	0,41
GJ5X	1909	909	10	26,5		-	751	-	0,62		1315	715	6	0,42
GJ6X	2854	954	19	0,7		-	886	-	0,6		2398	1098	13	0,39
GJ7X	2573	973	16	1,5		-	878	-	0,6		2092	892	12	0,43
GJ8X	8326	5326	30	52,2		-	3751	-	9,56		5406	3506	19	2,76
GJ9X	7026	4426	26	150		-	3398	-	9,47		5332	3332	20	2,79
GJ10X	7546	4446	31	157		-	3311	-	6,5		4778	2878	19	2,77
GJ11X	7423	4323	31	40,5		-	3263	-	9,42		4221	2321	19	2,85
Average	4178	2232	19,5	39,5			1713		3,4		2781	1508	12,7	1,27
											(33.4%)[a]	(12.0%)[b]	(34.9%)[c]	
GJ1Y	1814	614	12	0,2		-	541	-	0,08		1052	352	7	0,37
GJ2Y	1019	519	5	0,3		-	492	-	0,08		539	139	4	0,34
GJ3Y	2137	737	14	1,4		-	638	-	0,26		1219	419	8	0,45
GJ4Y	2962	1162	18	1,7		-	932	-	0,63		2032	832	12	0,38
GJ5Y	1712	912	8	26,5		-	751	-	0,36		1338	738	6	0,42
GJ6Y	2603	1003	16	3,1		-	886	-	0,61		2197	997	12	0,38
GJ7Y	2573	973	16	1,5		-	878	-	0,61		2025	825	12	0,45
GJ8Y	5504	4804	7	24,7		-	3751	-	9,6		5417	3517	19	2,82
GJ9Y	7601	4501	31	27,8		-	3398	-	6,54		5395	3395	20	2,95
GJ10Y	7083	4183	29	35,9		-	3311	-	9,6		4883	3983	19	2,92
GJ11Y	7457	4357	31	40,5		-	3263	-	6,57		4377	2477	19	3,59
Average	3860	2160	17,0	14,9			1713		3,2		2770	1516	12,5	1,37
											(28.2%)[a]	(11,5%)[b]	(26.5%)[c]	

[a] *The total transportation cost obtained from Salhi and Nagy (1999) improved by GA2*

[b] *The routing cost obtained from Gajpal and Abad (2009) improved by GA2.*

[c] *The number of required vehicles from Salhi and Nagy (1999) improved by GA2.*

Table 7 reports the results obtained by existing heuristics and GA2 for MD-VRPSDP. In the previous results, those of Gajpal and Abad (2009) are better.

The results show that the performance of the algorithm developed in this paper is better than the performance of previous algorithms. For the instances X, Table 7 shows that our proposed algorithm improves the average value of Gajpal and Abad (2009) by 12% and for the instances Y, the improvement is 11.5%. And the results, of the number of required vehicles, obtained by GA2 further improve the average values of Salhi and Nagy (1999) by 34.9% and 26.5% for instances X and Y, respectively. It should be noted in particular that the CPU time is considerably much less compared to existing heuristics; an improvement of more than 85% is observed. Considering these results and CPU times, it can be stated that, the proposed hybrid GA perform well and find good solutions very efficiently. Finding adequate (good enough) solutions in a short time frame is the ultimate goal of GAs, even when the problem size is growing.

VI. CONCLUSION

MD-VRPSDP-IR is important and practical given the need for integrating forward and reverse flows of material. It is an extension of the VRPSDP which is not yet addressed in the literature. It is a more complicated problem, considering that it needs to tackle multiple depots, inventory restrictions and the VRPSDP problem simultaneously. The considered objective is to minimize the total transportation cost due to the weighted sum of the total distance traveled and the cost related to the number of required vehicles, as mentioned in Section 3 after introducing MD-VRPSPD-IR and its mathematical formulation.

This study contributes to the VRPSDP field by providing an efficient hybrid GA that provides good solutions in a short time frame for MD-VRPSDP-IR. Our contribution in this paper is that we developed a new method, the K-ND heuristic, to assign customers to depots, and we compare its performances with those obtained by the random assignment as well as by the assigning customers to the nearest depot. The proposed algorithm embeds, for each depot as a sub-problem, the Sweep algorithm to construct routes and the Farther Insertion heuristic to improve the solution. Details of the integrity of the proposed method were given in Section 4.

The efficiency of our newly developed heuristic is attested by performance evaluation of the proposed algorithm with computational experiments for MD-VRPSDP-IR and MD-VRPSDP. Moreover, according to the results obtained by

CPLEX, for a small instance, it can be concluded that the proposed Hybrid GA both performs well and is efficient, and gives good and feasible solutions.

Further studies may explore more procedures for assigning customers to depots such as assignment through urgencies which assigns the customers with highest urgency first, that is a way to define a precedence relationship between customers. This work has also to continue testing and comparing other construction and improvement heuristics such as Petal method. Other topics for future work are to include a new crossover and mutation operators, with flexible rates, that will fit more with the nature of the studied problem. Additionally, the proposed method may be applied to a real world routing problems with simultaneous pick-up and deliveries with inventory restrictions.

References

[1] J. Dethloff, “Vehicle routing and reverse logistics: The vehicle routing problem with simultaneous delivery and pick-up”, Operations Research Specktrum, vol. 23, pp 79–96, 2001.

[2] J. F. Chen, and T. H. Wu, “Vehicle routing problem with simultaneous deliveries and pickups”, Journal of the Operational Research Society, vol. 57, pp 579-587, December 2017.

[3] E. Berhan, “Stochastic vehicle routing problems with simultaneous pickup and delivery services”, Journal of Emerging Trends in Computing and Information Sciences, ISSN 2079-8407, vol. 6(7), July 2015.

[4] H. Min, “The multiple vehicle routing problem with simultaneous delivery and pick-up points”, Transportation Research, vol. 23(5), pp. 377–386, September 1989.

[5] R.P. Hornstra, K.J. Roodbergen, and L.C. Coelho, “The Vehicle routing problem with simultaneous pickup and delivery and handling cost”, Interuniversity Research Center on Entreprise Networks, Logistics and and Transportation, Working paper N° 27, June 2018.

[6] H. Koch, A. Bortfeldt, and G. Wascher, “A hybrid solution approach for the 3L-VRP with simultaneous delivery and pickups”, Working paper N° 5, 2017.

[7] Y. Nilufa, “Development of a fuel consumption optimization model for the vehicle routing problem with simultaneous delivery and pickup”, Thesis, November 2017.

[8] S. Majidi, S.M. Hosseini-Motlagh, S. Yaghoubi, and A. Jokar, “Fuzzy green vehicle routing problem with simultaneous pickup-delivery and time windows”, RAIRO Operations Research, January 2017.

[9] S. Suprayogi, and Y. Priyandari, “Tabu search for the vehicle routing problem with multiple trips, time windows and simultaneous delivery-pickup”, Jurnal Teknik Industri, vol. 19(2), ISSN 1411-2485, December 2017.

[10] O. Polat, C. B. Kalayci, O. Kulak, and H. O. Günther, “A perturbation based variable neighborhood search heuristic for solving the Vehicle Routing Problem with Simultaneous Pickup and Delivery with Time Limit”, European Journal of Operational Research, vol. 242(2), pp. 369-382, April 2015.

[11] L. Zhu, and J.B. Sheu, “A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup–delivery and time windows”, Computers & Industrial Engineering, vol. 83, pp.111-122, May 2015.

[12] H. Karimi, “The capacitated hub covering location-routing problem for simultaneous pickup and delivery systems”, Computers & Industrial Engineering, vol. 116, pp. 47-58, February 2018.

[13] C. Lagos, . GGuerrero, E. Cabrera, M.P. Andres, F. Johnson, and F. Paredes, “An improved particle swarm optimization algorithm for the VRP with simultaneous pickup and delivery and time windows”, IEEE Latin America Transactions, vol. 16(6), pp.1732-1740, June 2018

[14] H ang, G. Zhijing, Y. Peng, and S. Junqing, “Vehicle routing problem with simultaneous pickups and deliveries and time windows considering fuel consumption and carbon emissions”, Chinese Control and Decision Conference, May 2016.

[15] W. Jiahai, Z. Ying, W. Yong, Z. Jun, C.L.P. Chen, and Z. Zibin, “Multiobjective vehicle routing problems with simultaneous delivery and pickup and time windows: formulation, instances and algorithms”, IEEE Transactions on Cybernetics, vol. 46(3), pp. 582-594, March 2016.

[16] S. Mostafayi, S. Moazeni, M. Dahmardeh, and K. Mokhtari, “Vehicle routing problem with regard to simultaneous pickup and delivery, time windows and workers assignment on the basis of their abilities and availability”, MAGNT Research Report, vol. 3(1), pp. 423-434, January 2015.

[17] AM. Shahdaei, and AM. Rahimi, “Solving vehicule routing problem with simultaneous pick-up and delivery with the application of genetic algirithm”, Indian Journal of Fundamental and Applied Life Sciences, vol. 6(S1), pp. 247-259, 2016.

[18] C.X.C.A. Bárbara, P.H. Siqueira, F.A. Giancarlo, and S. Luzia Vidal, “Particle swarm optimization for vehicle routing problem with fleet heterogeneous and simultaneous collection and delivery”, Applied Mathematical Sciences, vol. 8(77), pp. 3833 - 3849, 2014.

[19] R. Liu, X. Xie, V. Augusto, and C. Rodriguez, “Heuristic algorithms for a vehicle routing problem with simultaneous delivery and pickup and time windows in home health care”, European Journal of Operational Research, Elsevier, vol. 230(3), pp. 475-486, January 2013.

[20] S. Cetin, and C. Gencer, “A Heuristic algorithm for vehicle routing problems with simultaneous pick-up and delivery and hard time windows”, Open Journal of Social Sciences, vol.3, pp. 35-41, 2015.

[21] S. Salhi, G. and Nagy, “A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling”, Journal of the Operational Research Society, vol. 50(10), pp. 1034–1042, September 1999.

[22] G. Nagy, and S. Salhi, “Heuristic algorithms for single and multiple depot vehicle routing problem with pickups and deliveries”, European Journal of Operational Research, vol. 162(1), pp.126–141, April 2005.

[23] Y. Gajpal, and P.L. Abad, “Saving based for algorithm for multi-depot version of vehicle routing problem with simultaneous pickup and delivery”, International Journal of Enterprize Network Management, vol. 3(3), pp. 201-222, 2009.

[24] J. Li, P.M. Pardalos, H. Sun, J. Peiand, and Y. Zhang, “Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups”, Expert Systems with Applications, vol. 42, pp 3551–3561, May 2015.

[25] Y.G. Cai, Y.L. Tang, and Q.J. Yang, “An improved genetic algorithm for multi-depot heterogeneous vehicle routing problem with simultaneous pickup and delivery time windows”, Applied Mechanics and Materials, vols. 738-739, pp. 361-365, 2015.

[26] Y. Shimizu, T. Sakaguchi, and J.K. Yoo, “A hybrid method for solving multi-depot VRP with simultaneous pickup and delivery incorporated with weber basis saving heuristic”, Journal of Advanced Mechanical Design, Systems, and Manufacturing, vol.10(1), 2016.

[27] S. Karakatic, and V. Podgorelec, “A survey of genetic algorithms for solving multi depot vehicle routing problem”, Applied Soft Computing, vol. 27(0), pp. 519–532, February 2015.

[28] B.E. Gillett, and J.G. Johnson, “Multi-terminal vehicle-dispatch algorithm”, OMEGA, the International Journal of Management Science, vol. 4(6), pp. 711-718, 1976.

[29] N. Christofides, and S. Eilon, “An algorithm for the vehicle dispatching problems”, Operations Research Quarterly, vol. 20, pp. 309-318, September 1969.

[30] B.E. Gillett, and L.R. Miller, “A heuristic algorithm for the vehicle-dispatch problem”, Operations Research, vol. 22(2), pp. 340-349, April 1974.