A Novel Meta-Heuristic Combinatory Method for Solving Capacitated Vehicle Location-Routing Problem with Hard Time Windows

Chapter · October 2016

DOI: 10.1007/978-3-319-38789-5_77

CITATIONS
8

READS
6,329

4 authors, including:

Ali Asghar Rahmani Hosseinabadi
University of Regina
52 PUBLICATIONS **1,005** CITATIONS
SEE PROFILE

Shahab S. Band
493 PUBLICATIONS **9,639** CITATIONS
SEE PROFILE

Valentina Emilia Balas
Aurel Vlaicu University of Arad
472 PUBLICATIONS **3,321** CITATIONS
SEE PROFILE

A Novel Meta-Heuristic Combinatory Method for Solving Capacitated Vehicle Location-Routing Problem with Hard Time Windows

Ali Asghar Rahmani Hosseinabadi, Fataneh Alavipour, Shahaboddin Shamshirbnd and Valentina E. Balas

Abstract Capacitated location-routing problem (CLRP), is one of the new research areas in distribution management. This topic combines two problems: locating the facilities and vehicle routing. The goal of CLRP is to open a set of depots, allocating the costumers to depots and then to design the vehicle tours in order to minimize the overall cost. The limitations of time windows has many applications in the real world, however it has not been noticed enough in the CLRP problem. This article considers the capacitated location-routing problem with hard time windows (CLRPHTW). In this paper, first a mixed-integer linear programming model for CLRPHTW problem is presented and then in order to solve this problem a meta-heuristic method based on variable neighborhood search algorithm is proposed. To assess the performance of the proposed method, this framework is examined with a set of examples. The computational tests demonstrate the efficiency of the proposed method.

Keywords Location · Vehicle routing · Time window · Meta-heuristic method · Variable neighborhood search · Optimization

A.A.R. Hosseinabadi
Department of Computer, Technical and Vocational University,
Behshahr Branch, Behshahr, Iran
e-mail: A.R.Hosseinabadi@iaubeh.ac.ir

F. Alavipour
Department of Electrical, Computer and Biomedical Engineering,
Qazvin Islamic Azad University, Qazvin Branch, Qazvin, Iran
e-mail: F.Alavipour@qiau.ac.ir

S. Shamshirbnd (✉)
Faculty of Computer Science and Information Technology,
Department of Computer System and Technology,
University of Malaya, 50603 Kuala Lumpur, Malaysia
e-mail: Shahab1396@gmail.com; Shamshirband@um.edu.my

V.E. Balas
Department of Automation and Applied Informatics,
Aurel Vlaicu University of Arad, Arad, Romania
e-mail: Valentina.balas@uav.ro

V.E. Balas et al. (eds.), *Information Technology and Intelligent Transportation Systems*, Advances in Intelligent Systems and Computing 454,
DOI 10.1007/978-3-319-38789-5_77

1 Introduction

In the last two decades, the competition between firms for supply of goods and services, has become a serious fact in their progress path. Today firms and factories require unified and flexible production activities from preparation of raw material to delivery of products to the consumers. Distribution and support systems are part of the supply chain, and the producers need to plan, run and control the efficient flow of goods' storage, services and the related information from the start point to consumption [1].

Location-routing problem (LRP), is one of the important and practical problems in distribution and logistics management. LRP is a combination of two problems: locating the facilities and vehicle routing that considers these two problems at the same time. If the problems of locating and routing are not considered at the same time, costs of supply chain will increase. Both of these problems involve the complexities of NP-hard problems, thus LRP is a problem which involves time complexities in the form of NP-hard problems. Therefore solving the LRP problem in large scale using exact methods is difficult and almost impossible [1], and in most of the conducted research, heuristic and meta-heuristic methods are developed to solve these this kind of problems.

In LRP problem, the capacity constraint includes depot or the vehicle (not both of them). Recently some of the researchers have studied the LRP problem with capacity constraint both in depot and in the vehicle [2]. This subject is called capacitated location-routing problem (CLRP). Prins et al. [2] in their paper, have proposed a mathematical model for CLRP. They have proposed a solution for CLRP by combining Greedy Randomized Adaptive Search Procedure (GRASP), a learning procedure and unifying the routs mechanisms. In other paper, they also have used a Memetic algorithm with population management. In this algorithm, the solution is improved by using local search methods and replacing the conventional mutation operator with a dynamic population management technique based on distance to population [3].

Barreto et al. [4] used a heuristic method based on customer classification to solve CLRP. They used several hierarchical and non-hierarchical methods for classification. Prins et al. [5] used Lagrangean Relaxation and Tabu Search methods to develop a two phase approach for solving CLRP. This algorithm alternatively exchange the information between a locating phase and a routing phase. In the first phase, the routs and their customers are integrated in the form of a super customer and the problem is transformed into a facility locating problem. Lagrangean relaxation is used to deal with the allocation constraint in the facility locating problem. In the second phase, Tabu Search method is used to develop the routing of the several obtained depots from the first phase.

In [6] a combined method is used based on particle swarm optimization algorithm to solve the problem. In [7] (GRASP+ELS) algorithm which is a combination of greedy randomized adaptive search procedure and evolutionary local search is employed for CLRP problem. Yu et al. [8] used heuristic simulated annealing algorithm to solve CLRP problem. In order to improve the performance of the simulated

annealing algorithm they used three neighborhood structures. They claimed that using this structure will improve the simulated annealing algorithm to solve CLRP problem. They used three neighborhood structures with probable selection to improve the performance of the simulated annealing algorithm. Recently Nguyen et al. [9] in their paper used greedy randomized adaptive search procedure with learning process to solve CLRO in two levels. In other research, they also solved the same problem using a method based on iterative neighborhood search algorithm and improved the results of their previous research [10].

Although time window constraint has numerous applications in real world, little attention has been paid to it in CLRP problem. Jabal-Ameli and Ghaffari-Nasab [11] has modeled the location-routing problem in both hard and soft time windows and by using a simple example has shown the credibility of their model. Nikhbakhsh and Zegardi has studied the two level location-routing problem with soft time window constraint. The authors of this article have employed a four-index mathematical planning model and a heuristic method in order to solve this problem. By using the relaxation Lagrangean method, they also obtained a lower bound for the problem and compared it with the results of their heuristic algorithm.

The main purpose of this article is to add the time window constraint to the CLRP problem in order to obtain a more practical solution. The time window constraint is taken from the vehicle routing problem with time window constraint (VRPTW). This subject is called capacitated location-routing problem with hard time window (CLRPHTW). In this article first the mathematical modeling of the problem is discussed. Next, considering the complexity level of the problem solution and inability of the exact methods in solving problems with average and large dimensions in reasonable time, a meta-heuristic algorithm based on variable neighborhood search is proposed to solve the problem. The novel features of this article, is addition of waiting time to the mathematical model in [11] for the capacitated location-routing problem with hard time windows and designing a meta-heuristic algorithm to solve this problem.

The rest of the article is organized as it follows: in the second section, the problem is defined and its mathematical model is presented. In the third section, the proposed algorithm and its details are explained. In the fourth section, the results of the computational experiments are illustrated and finally the proposed algorithm and the conclusion are described.

2 Problem Definition and Its Mathematical Model

In the capacitated location-routing problem with hard time windows, a set of customers and a set of candidate points for construction of depots with predetermined geographical coordination exist. The customer demand and capacity of depots is known. All the transportation vehicles have the same capacity. Every customer is assigned only one depot and one vehicle in order to receive his order, therefore constraints for depots capacity and vehicle capacity should be considered.

Every vehicle starts its tour from the depot and after delivering the service to the assigned costumers, returns to the same depot. Every costumer has a predetermined time interval $[e_i, l_i]$, where e_i is the soonest time and l_i is the latest time to receive the service from the vehicle. In the case of hard time windows, the customer receive the service only in the determined interval and if the vehicle arrives sooner that the start time, it should wait until the time window opens. The purpose is to minimize the costs and to determine the suitable location for construction of depots and planning the tours of the vehicles for each depot; such that the all customers' demands is provided [5].

In order to explain the mixed integer linear programming (MILP) of the problem, first the symbols and parameters of the mathematical model are explained, and then by adding the considerations about the interval of the time windows to the capacitated location-routing problem of Prins et al. [5], the problem under study is modeled.

Sets:

I The set of depot candidate points
J The set of customers
V The set of all points $I \cup J = V$
K The set of vehicles

Model Parameters:

d_j Demand of Jth customer
O_i Cost of opening the candidate depot i
F_i Cost of using a vehicle in candidate depot i
C_{ij} Cost of traveling from point I to point j
t_{ij} Time of travelling from point i to j
Cap_i Capacity of candidate depot i
Q Capacity of each transportation vehile
T Maximum time during a tour
e_i Soonest time to deliver service to point i
l_i latest time to deliver service to point i
St_i Time duration of delivering service the customer j
M A large number

Decision variables:

At_i^k Time required for vehicle k to arrive to point i
W_i Waiting time in point i
y_i If candidate depot I is opened, one and otherwise zero
f_{ij} If the demand of customer j is provided by depot I, one and otherwise zero
X_{ij}^k If vehicle K goes directly from point I to point j, one and otherwise zero

Considering the above mentioned points, the mixed integer linear programming of CLRPHTW is presented as below:

$$\text{Minimize} \quad \sum_{i\in I} Oi + yi \sum_{i\in V} + \sum_{j\in V} + \sum_{k\in K} + Cij X_{ij}^{k} + \sum_{i\in I} \sum_{j\in I} \sum_{k\in K} FiX_{ij}^{k} \quad (1)$$

Subject to:

$$\sum_{i\in V} \sum_{k\in K} X_{ij}^{k} = 1, \qquad \forall j \in J \quad (2)$$

$$\sum_{i\in V} \sum_{j\in J} d_j X_{ij}^{k} \le Q, \qquad \forall k \in K \quad (3)$$

$$\sum_{j\in J} d_j Y_{ij} \le cap_i y_i, \qquad \forall i \in I \quad (4)$$

$$\sum_{i\in V} \sum_{j\in v} t_{ij} X_{ij}^{k} \le T, \qquad \forall k \in K \quad (5)$$

$$\sum_{i\in s} \sum_{j\in s} X_{ij}^{k} \le |S| - 1, \qquad \forall S \subseteq J, k \in K \quad (6)$$

$$\sum_{j\in V} X_{ij}^{k} - \sum_{j\in v} X_{ji}^{k} = 0, \qquad \forall i \in V, k \in K \quad (7)$$

$$\sum_{i\in I} \sum_{j\in J} X_{ij}^{k} \le 1, \qquad \forall k \in K \quad (8)$$

$$\sum_{m\in V} X_{im}^{k} + \sum_{h\in V} X_{ih}^{k} \le 1 f_{ij}, \qquad \forall i \in I, j \in J, k \in K \quad (9)$$

$$At_i^{k} + St_i + W_i + t_{ij} - At_j^{k} \le (1 - X_{ij}^{k})M \qquad \forall i \in I, j \in J, k \in K \quad (10)$$

$$e_j \le At_j^{k} + W_j \le l_i, \qquad \forall j \in J, k \in K \quad (11)$$

$$At_i^{k} + W_j + St_i + t_{ij} - l_i \le (2 - y_1 - X_{ij}^{k}) \times M, \qquad \forall i \in I, j \in J, k \in K \quad (12)$$

$$At_i^{k} = 0, \qquad \forall i \in I, k \in K \quad (13)$$

$$W_i = 0, \qquad \forall i \in I \quad (14)$$

$$At_i \ge 0, \qquad \forall i \in V \quad (15)$$

$$W_i \ge 0, \qquad \forall i \in V \quad (16)$$

$$f_{ij} \in \{0, 1\}, \qquad \forall i \in I, j \in J \quad (17)$$

$$y_i \in \{0, 1\}, \qquad \forall i \in I \quad (18)$$

$$X_{ij}^{k} \in \{0, 1\}, \qquad \forall i \in I, j \in V, k \in K \quad (19)$$

In this model, the objective function (1), minimizes the sum of depot costs and routing costs which routing costs include vehicle and travelling costs. Constraint (2) guarantees that every customer belongs to only one rout.

Constraints (3) and (4) are vehicle and depot capacity constraints. The maximum time for travelling a tour is shown by expression (5). Constraint (6) is for eliminating the sub tours. Constraint (7) guarantee the tours are continuous and constraint (8) makes sure that every vehicle returns to the same depot from which it started the tour. Expression (9) states that every customer is assigned to one depot, if there is a tour that connects these two together.

Expression (10) shows the arrival time to one customer and to the next customer. Constraints (11) and (12) guarantee that time windows of customers and depots are

not violated. Constraints (13) and (14) determine the initial conditions of the waiting time and arrival time variables for the depots. Constraints (15) and (16) make sure that decision variables of arrival time and waiting time are positive and finally constraints (17)–(19) state that other decision variables are binary. After defining the problem and presenting its mathematical model, we propose a meta-heuristic method for the solution.

3 The Proposed Solution Method

The proposed approach for solving CLRPHTW is a method based on variable neighborhood search meta-heuristic algorithm. In the proposed algorithm, first generalized Push-Forward Insertion Heuristic algorithm is used to generate the initial solution for start of the proposed algorithm. The initial solution (S_0) is used as the current solution for the algorithm. Now the main algorithm starts based on the variable search neighborhood algorithm. In order to have a better search of the solution space, the simulated annealing algorithm is used in the framework of the variable neighborhood algorithm. In the following the components of the new method are explained in detail and in the end, the structure of the proposed algorithm is presented.

3.1 Variable Neighborhood Search Algorithm

Variable Neighborhood Search (VNS) algorithm first was introduced by Mladenovi'c Hansen [12] in year 1997. Its pseudo code is provided in Fig. 1. The main idea of this algorithm is to change the neighborhood structure during the search in order to prevent trapping into local minimum. Because of simple implementation and acceptable quality of the obtained results, this algorithm was soon recognized as a good method for solving optimization problems. VSN algorithm starts the optimization procedure by generating initial solution, defining neighborhood structures and using a method for searching the neighborhood. The neighborhood structures of the algorithm are shown by N_l, $l = \{1, 2, \ldots, l_{max}\}$, where N_l is the lth neighborhood. After defining the possible neighborhood structures, the order is determined. Two important points at this stage are "selection of appropriate neighborhood structures and determination of suitable structure order" (for example order based on size of the neighborhood structures). Generating initial high quality solution, defining neighborhood structure, determining neighborhood order and using the suitable method for local search, are important factors which affects the quality of the solution results. The algorithm starts searching by using the initial solution (S_0) and until the termination criteria is met, the main loop of the algorithm is iterated. The main loop of the VSN algorithm includes two phases: moving phase and local search phase.

In the shaking phase, using the lth neighborhood structure, the algorithm moves from the current solution to the neighborhood solution (S'). In the local search phase

Fig. 1 Pseudo code for variable neighborhood search algorithm

Input: a set of neighborhood structures $N_l, l = 1,2,...,l_{\max}$
S=generate initial solution ( );
Repeat
$l = 1$;
While $(l \le l_{\max})$
{
S' =Shaking (S, N_l)
S'^* =Local search (S')
if $f(S'^*) < (f(S)$
$S \leftarrow S'^*$
$l = 1$;
else
$l = l + 1$;
}
Until stopping condition are met ;
Output : The best solution ;

by applying local search methods, searching is performed on the S' solution until local optimum is obtained. Now in the moving phase, if the optimum solution obtained is better than the current solution S, will replace it and the search returns to N_l, otherwise the next neighborhood structure $(N_l + 1)$ is used to continue the search. The search is continued until $<l_{max}$. Figure 1 shows the pseudo code for variable neighborhood search algorithm.

3.2 Simulated Annealing Algorithm

Simulated annealing algorithm (SA) is a heuristic method based on local search. This algorithm is able to prevent trapping in a local minimum by accepting worse solution with a lower probability. The acceptable quality of simulated annealing solutions has motivated the researcher community to apply it for solving complex combinatorial optimization problems in real world applications. The usage of simulated annealing algorithm was made popular by Kirkpatrick et al. [13]. The basic idea of simulated annealing is derived from annealing process of metals in metallurgy industry.

The process of optimization in simulated annealing, is the search for a solution (near) of global minimum. The algorithm starts from a random solution as the initial solution and the system temperature will be equal to this initial temperature ($T = T_0$). In every iteration, a solution in neighborhood of the current solution is obtained. The objective function of the new solution is compared to the current solution. If the solution is better it will replace the current solution and if it is worse, it will replace the current solution by a probability which is obtained from the Boltzmann

Fig. 2 The pseudo code of simulated annealing algorithm

```
Initialize parameters;
S=generate initial solution ( );
T = T_0;
While (T < T_final)
{
   Until (N ≤ I − Iter)
    {
        Generate solution S' in the neighborhood of S
                if f(S'*) < (f(S)
                        S ← S'
                else
                        Δ = f(S') − f(S)
                        r = random( );
                        if (r < exp(−Δ / k * T))
                        S ← S'
    }
        T = a × T;
}
Return the best solution found;
```

function $\exp(-\frac{\Delta}{KT})$. This mechanism prevent the algorithm from being trapped in a local minimum. In this relation Δ is the difference in objective function between the current solution and the new solution, K is the Boltzmann constant which is determined beforehand and T is the current temperature. The process of neighborhood search is continued until the number of iterations reaches to a predetermined value. After this step, the system temperature is reduced. This process is continued until the termination criteria is met [13]. The pseudo code of this algorithm is shown in Fig. 2.

3.3 Generating the Initial Solution

The algorithm requires an initial high quality solution to start the search, therefore the generalized PFIH algorithm for VRPTW in Solomon [14] paper is used to generate the initial solution. The PFIH algorithm is one of the usual heuristic algorithms for generating initial feasible solutions for solving the routing problem with time window constraint. This algorithm, by considering time window and vehicle capacity constraints, generates feasible solutions. The required steps for generating initial solution are explained below:

1. First all the customers are included in the list of uncounted customers.
2. Every customer in the list of uncounted customers is assigned to the nearest depot. Now every depot which has the most customers in the nearby, will be opened.
3. From the uncounted list, considering the distance to the opened depot, customers are assigned to that depot until the depot capacity is full.
4. After assigning the customers, the assigned customers will be omitted from the uncounted list and the depot will be omitted from the candidate depot list.
5. If there is still any customer in the list of uncounted customers, return to step 2, otherwise go to final step.
6. In this step, the proposed method by Yu et al. [8] for producing the initial solution for vehicle location-routing problem is employed. The initial solution of the proposed algorithm will be used in a reasonable time for achieving high quality solutions.

3.4 Representing the Solution and Calculating the Objective Function

In order to implement the computer code of the algorithm, first a structure is required to represent the solution of the problem. The structure for representing the solution, makes the problem understandable for the computer. This structure should be able to represent all the details of the solution clearly.

Designing the suitable structure has considerable role in performance of the algorithm. The structure for representing the solution in CLRPHTW is similar to LPR. If we have n customers, m construction sites for depots and k vehicles, then the solution will be represented by an array with the following order: n customers which are represented by 1, 2, …, n numbers, m depots which are represented by $n + 1$, $n + 2$, …, $n + m$ numbers and $k - 2$ zero elements. Figure 3 illustrates a small CLRP with 10 customers and 3 candidate construction sites for depots.

In this figure depots number 11 and 13 are opened. In this method for solution representation, a depot is open when between this depot and the other depot there is at least one customer in the solution representation. As the figure shows, customers 3, 5, 7, 2 and 1 are assigned to depot 1 and other customers receive service from depot number 13. Also in this method of solution representation, every two depots have two vehicle tours. For instance in the first tour of depot 11, the tour starts from depot 11 and after delivery to customers 3, 5 and 7 respectively, the vehicle returns to depot 11.

11	3	5	7	0	2	1	12	13	4	8	6	0	10	0	0	0

Fig. 3 Illustration of the employed solution for the proposed algorithm

The important point is the violation of depot capacity constraint, vehicle capacity and time windows that may occur in this kind of solution representation. In this paper in order to resolve this problem, a penalty term is added to the objective function to eliminate the infeasible solutions. In this strategy, the penalty is proportional to the extent that constraints are violated. In this paper, the parameters of P-vehicle, P-depot and P-tw are defined for violating the vehicle capacity constraint, depot constraint and time window constraint respectively.

3.5 Neighborhood Structures

In the proposed algorithm, four neighborhood structures are employed. The structures used in this paper are the ordinary neighborhood structures for solving vehicle routing problem with time windows. This four structures include Relocation operator, Swap operator, Or-opt operator and 2-opt operator. The order of the applied neighborhood structures is described below.

In the relocation operator, first two i and j elements in the solution array are selected randomly, then the i element is moved and is placed in the location next to element j. In the swap operator, after selecting two elements, they are replaced in the solution array. Or-opt operator selects a tour and delivers service to three customers and then places this chain in other location of the tour for improvement. 2-opt operator selects two tours from the solution array randomly, then divides each tour in a randomly selected location, next replaces the customers from the first and second parts of the first tour respectively with second and first parts of the second tour in the current solution array.

3.6 The Proposed Algorithm for Solving CLRPHTW

After explaining the main components, in this section we completely introduce the proposed algorithm for solving the location-routing problem with hard time windows. First the algorithm starts the work with the initial solution S_0.

The initial solution is considered as the current solution of the system ($S = S_0$). For solving the problem four neighborhood structures are employed. The algorithm starts with the first neighborhood structure, $l = 1$. First in the shaking phase based on the lth neighborhood structure, the algorithm moves from the current solution S to the neighborhood solution S'. Now in the local search phase, a local search is performed on the neighborhood solution S' so that the local optimum S'^* is obtained. Neighborhood search in the proposed algorithm is performed employing the simulated annealing algorithm by using the lth neighborhood structure as the movement operator in order to move to the neighborhood points. Here the length of the Markov chain for neighborhood search with a certain neighborhood structure is equal to N successive iterations without improvement in the objective function. In the

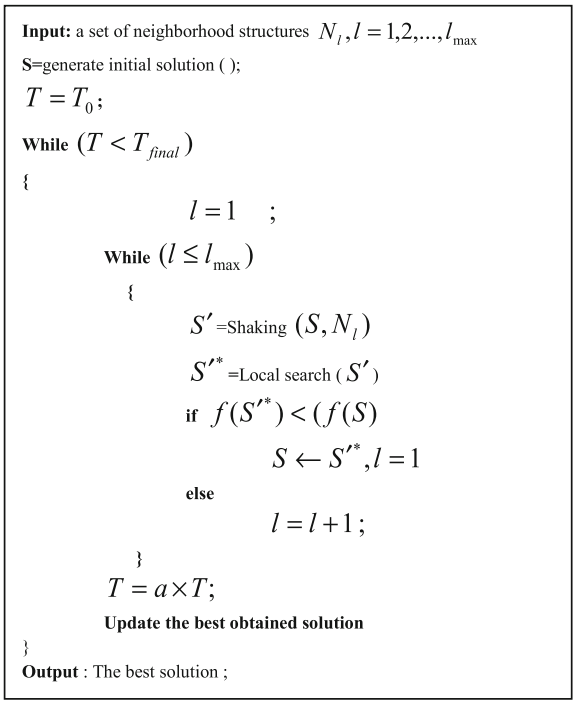

Fig. 4 The pseudo code of the proposed algorithm for solving the CLRPHTW problem

movement or non-movement phase if the value of the objective function for local optimum S'^* is better than S, *we set* $S = S'^*$ and $l = 1$, otherwise the nest neighborhood structure $l = l + 1$ is used. As long as $l \le l_{max}$ the loop for variable neighborhood search algorithm in a certain temperature is continued. Here $l_{max} = 4$, and after that the temperature is updated according to the geometric reduction law ($T = \alpha \times T$). The proposed algorithm is continued until the final temperature is achieved. The flow chart of the proposed algorithm is illustrated in Fig. 4.

4 Computational Experiments

The proposed algorithm was programmed in C++ language on a Laptop with hardware features of Core i5 2.53 GHz CPU and 4 GB RAM memory. Suitable initialization of algorithm parameters have an important role on the quality of the obtained results. Therefore some values are recommended for each parameter, then according to Table 1 the value of each parameter is varied while other parameters are kept constant at their lowest possible value, and the quantity of the varying parameter is adjusted based on the objective function and the required time to achieve the solution. After performing the above experiment on one of the generated examples, the

Table 1 The proposed values for adjusting the algorithm parameters

Number	Parameters	Recommended values
1	Penalties	50, 100, 200, 500
2	N	0.5L (L − 1), L(L − 1), 2L(L − 1), 3L(L − 1)
3	T_0	30, 50, 70, 100
4	α	0.9, 0.93, 0.95, 0.97, 0.99
5	T_{final}	0.001, 0.01, 0.1,1
6	K	0.1, 0.2, 0.5, 1

algorithm parameters were adjusted as it follows: all the penalty factors (every unit of violating the depot capacity, vehicle capacity and time window) are considered equal to 200 unit, the initial temperature (T_0) was considered 50 °C and the final temperature T_{final} was 0.001 °C. The reduction temperature factor (α) is 0.95 and the Boltzmann constant is 0.2. The number of successive iterations without improvement in the local search (N) is considered L(L − 1) where L is the length of the solution array.

Since there is no benchmark example for the this algorithm, we need to generate a number of examples. In this paper, assessment of the proposed algorithm is done in two parts: in the first part some small size examples are employed to assess the proper performance of the model and the algorithm and in the second part average and large size examples are applied for evaluation.

4.1 Evaluation of Algorithm Performance in Small Size Examples

In order to evaluate the validity of the mathematical model and the performance of the proposed algorithm in small size examples, first some examples should be generated. In this paper 10 small size examples including maximum 15 customers and 5 candidate depot sites are generated. The examples are produced according to the following procedure: 20 points are generated randomly in an area having 50 × 50 dimensions. The first 15 points are the customers and the other 5 points are candidate sites to construct the depots. To generate the demands of each customer, a number is generated randomly and with a uniform distribution in the interval [10, 20]. For the 15 customers, first a number in the interval of [0, 20] is generated randomly as the soonest delivery time and then a number in the interval of [30, 35] is generated randomly as the latest delivery time. The time window of each depot is considered as [0, 100]. There are 10 transportation vehicles, all of them are the same with the capacity of 80 products. Each depot have a capacity of 200 units and the cost of using a depot is 200 unit. The maximum traveling time to destination for

every vehicle is considered 200 time units. The traveling cost for every time unit is considered 1 monetary unit. Table 2 shows the location coordinates, time windows and the customer demands in the above generated examples.

The generated examples in this part are different in the number of customers and number of depot construction sites. For example in a problem with n customers and m candidate depot sites (n and m are maximum 15 and 5 respectively) the n first customers from the customers generated list and m first depots from the depot candidate list are selected. All the other parameters for the examples are considered equal.

The mathematical model for each of the examples are solved by the optimization software GAMS 23.6 with CPLEX solver in a laptop with hardware specifications of Core i5 2.53 GHz processor and 4 GM RAM memory in the time period of maximum 18,000 s (5 h).

In Table 3, a comparison between the solutions of the proposed algorithm in a single run and the solutions resulted from solving the mathematical model using the software is presented. In Table 3, the specifications of the 10 small size generated examples are presented in the first three columns; the examples differ in number of customers and number of candidate depot sites. In the three next columns, solution, solution time and the status of the obtained solution by GAMS software are shown respectively. In the status column, ***opt*** indicates the optimality of the solution and ***time*** indicates the time constraint to achieve the solution. The next two columns, the solution and time to achieve the solution by the proposed algorithm in a single run are shown. The last column shows the deviation of the algorithm solution with respect to the optimal solution in percent.

As it can be seen, the algorithm has reached to the optimal solution in the first 7 examples. In examples 8 and 9, the proposed algorithm has obtained solutions similar to GAMS, but solutions in GAMS are obtained with the 5 h' time constraint; however there is no guarantee that the obtained solutions are optimal. In the last example during the defined time constraint, GAMS software even did not reach to a solution equal to the proposed algorithm. considering Table 3, it can be concluded that: "as the problem size increases, the required time to achieve the optimal solution increases intensely, while the proposed algorithm has the ability to achieve a satisfactory solution in a very short time". The obtained solutions show the validity of the presented mathematical model. In the following part, the performance of the algorithm for average and large size examples will be evaluated.

4.2 Evaluation of Algorithm Performance for Average and Large Size Examples

Exact solution and obtaining the optimal results usually is not feasible expect for some small size examples. In addition, because there is no method for solving the capacitated location-routing problem with hard time window constraints, it is not

Table 2 The specifications of the generated small size example

Points	Location coordinates		Demand	Time window		Points	Location coordinates		Demand	Time window	
	X	Y		E	L		X	Y		E	L
1	27	4	18	1	37	11	22	33	11	18	49
2	16	30	16	8	48	12	26	20	20	12	50
3	5	24	1	5	37	13	23	41	20	11	42
4	31	35	20	16	32	14	44	36	15	3	31
5	39	35	16	9	46	15	26	49	18	17	34
6	21	32	11	19	38	16	36	9	0	0	100
7	4	1	13	3	35	17	24	37	0	0	100
8	13	3	16	5	38	18	30	13	0	0	100
9	7	16	20	3	32	19	18	21	0	0	100
10	14	27	20	2	32	20	31	46	0	0	100

Table 3 Comparison between GAMS software results and the proposed algorithm results

No Problem	Number of customers	No depot	Games			Proposed Algorithm		GAP (%)
			Answer	Time (s)	Status	Answer	Time (s)	
1	5	2	342.867	1.016	opt	342.867	0	0
2	7	2	457.845	16.782	opt	457.845	0	0
3	10	2	520.602	1801.34	opt	520.602	1	0
4	10	3	425.199	824.43	opt	425.199	1	0
5	12	2	524.147	12400	opt	524.147	1	0
6	12	3	428.981	14812.4	opt	428.981	2	0
7	12	5	387.127	17563	opt	387.127	2	0
8	15	2	631.029	18000	time	631.029	2	–
9	15	3	612.264	18000	time	612.264	2	–
10	15	5	626.015	18000	time	605.089	2	–

possible to compare the obtained results with previous method. Because of this, in the present paper, the two proposed algorithms are compared to each other. The first method which is called VSN algorithm, is the variable neighborhood search method without applying the simulation annealing algorithm in its neighborhood search structure. The second algorithm is called VNS+SA and employs the proposed variable neighborhood structure and applies simulated annealing algorithm in its neighborhood search structure.

Up to our knowledge, there is no benchmark example for capacitated location-routing problem with hard time constraints. Therefore in this paper, some of the benchmark examples in Solomon's work [14] after applying some variations are used as benchmark examples for CLRPHTW problem.

Solomon's benchmark examples for vehicle routing problem with time window constraint are available in [15]. Solomon [14] have proposed some benchmark examples for VRPTW problem as well. Their database is the most well-known database for assessment of proposed algorithms' performance for VRPTW problem and similar problems. In their proposed examples, number of customers is equal to 25, 50 and 100. They considered 100 customers and 1 depot to generate the examples and generated smaller size examples by considering the first 25 and 50 customers. They classified their examples in three main categories: C, R and RC.

The geographical locations which are considered for examples of C category are classified, for R category are random and for RC category are a combination of both classified and random and are generated in 2 dimensions. In each category, the customer and depot locations and customers' demands are considered the same and they only differ in time windows. Each category has consisted of 1 and 2 subcategories and examples in subcategory 2 has wider time windows. The transportation vehicles are the same with equal capacity and their quantity is known and limited.

To edit Solomon's benchmark examples, 4 depots are added to examples with 25 customers and also 9 depots are added to other examples. All the depots have the same capacity and costs. Generated examples for CLRPHTW are named by adding letter M to the beginning of Solomon's codes.

Every algorithm for every example is run 10 times and the best obtained answer together with the run time is shown. In these tables, the example code, the best obtained answer and the run time of both algorithms is presented. The last column is the deviation percentage between the best obtained answer by VNS+SA algorithm compare to VSN algorithm. If the obtained solution of VNS algorithm is S and that of VNS+SA algorithm is S', then the deviation percentage is calculated from the following relation: $GAP = \frac{s'-s}{s} \times 100$.

As it is indicated in Table 4, in 7 examples out of 18 examples including 25 customers and 5 candidate depot sites, the solutions of the proposed algorithm are equal to the solutions of the variable neighborhood search algorithm and in other cases VNS+SA algorithm has resulted in better solutions compared to VNS algorithm. In all the examples with 50 customers and 10 candidate depot sites except MC203 example, VNS+SA algorithm has resulted in better solutions compared to VNS algorithm (Table 5).

In order to improve the validity of the proposed algorithm, in examples with 100 customers and 10 candidate depot sites, in addition to comparing the solutions of the proposed algorithms from the view point of the best obtained solution, Paired-sampled T test is also used for statistical analysis. Because for every example and every algorithm 10 runs are available, therefore the sample size for every example and every algorithm is equal to 10 as well. Here we test the hypothesis of performance improvement of HVNS algorithm compared to the proposed VNS algorithm. For this purpose, we consider the hypothesis test below:

Null hypothesis: The average of results for the proposed VNS+SA algorithm is equal to the average of results of the proposed VNS algorithm.

Alternative hypothesis:The average of results for the proposed VNS+SA algorithm is lower than the average of results of the proposed VNS algorithm.

Since the statistical analysis in a meaningful level is 0.05 of the test, if the P-value is lower than $\alpha = 0.05$, there is no reason for accepting the null hypothesis. Table 6 represents the results of performance evaluation of the two proposed algorithms together with the results of statistical analysis of the paired-sample T test. In this table, Sig.(2-tailed) column reports the P-value and the last column reports the paired-sample T test value.

It can be observed from Table 6 that the solutions of the VNS+SA algorithm are better than VNS algorithm in all the 18 problems with 0.95 certainty level except the 6th problem; by comparing the difference percentage between the best obtained solution by VNS+SA algorithm and VNS algorithm, the same result is concluded.

Table 4 Details of the experiments for algorithm performance in examples with 25 customers and 5 candidate depots

Code example	VNS		VNS+SA		GAP (%)	Code example	VNS		VNS+SA		GAP (%)
	Objective function value	Time	Objective function value	Time			Objective function value	Time	Objective function value	Time	
MC101	1117.83	6	1077.92	7	−3.69	MR201	1160.73	6	1112.35	6	−4.35
MC102	1100.34	6	1076.85	6	−2.18	MR202	1038.93	6	1019.93	6	−1.86
MC103	1092.24	5	1069.68	7	−2.11	MR203	980.86	5	973.05	6	−0.8
MC201	1159.75	6	1136.11	7	−2.08	MRC101	1351.62	6	1306.92	6	−3.42
MC202	1152.89	5	1149.33	7	−0.31	MRC102	1211.55	6	1116.41	7	−8.52
MC203	1148.35	5	1140.49	6	−0.69	MRC103	1100.46	5	1100.46	7	0
MR101	1727.12	6	1727.12	6	0	MRC201	1124.05	6	1124.05	7	0
MR102	1573.8	4	1573.79	6	0	MRC202	1101.85	6	1101.85	6	0
MR103	1317.24	6	1317.24	6	0	MRC203	1091.17	6	1091.17	6	0

Table 5 Details of the experiments for algorithm performance in examples with 50 customers and 10 candidate depots

Code example	VNS		VNS+SA		GAP (%)	Code example	VNS		VNS+SA		GAP (%)
	Objective function value	Time	Objective function value	Time			Objective function value	Time	Objective function value	Time	
MC101	1364.28	32	1361.5	39	−0.2	MR201	1713.34	36	1612.33	40	−6.26
MC102	1365.73	33	1360.2	40	−0.41	MR202	1507.21	36	1460.5	43	−3.2
MC103	1357.67	38	1356.75	50	−0.07	MR203	1381.57	45	1320.8	45	−4.6
MC201	1288.09	53	1282.81	52	−0.41	MRC101	2221.74	28	2132.6	58	−4.18
MC202	1293.48	51	1259.63	50	−2.69	MRC102	1951.77	42	1795.27	63	−8.72
MC203	1263.36	32	1263.92	48	0.04	MRC103	1718.82	32	1651.16	60	−4.1
MR101	2584.13	35	2519.69	65	−2.56	MRC201	1581.32	35	1552.68	34	−1.84
MR102	2304.86	32	2216.65	50	−3.98	MRC202	1517.77	34	1517.03	33	−0.05
MR103	2043.92	34	1973.19	35	−3.58	MRC203	1441.66	35	1417.38	36	−1.71

Table 6 Details of experiments for algorithm performance in examples with 100 customers and 10 candidate depots.

Code example	VNS		VNS+SA		GAP (%)	P-value	Test results
	Objective function value	Time	Objective function value	Time			
MC101	2889.65	162	2709.25	148	−6.66	0.006	To accept in return for the imposition of
MC102	2727.91	131	2664.98	150	−2.36	0.004	To accept in return for the imposition of
MC103	2732.63	104	2635.32	155	−3.69	0.012	To accept in return for the imposition of
MC201	2045.2	139	1999.64	155	−2.28	0.001	To accept in return for the imposition of
MC202	2041.44	130	1927.03	144	−5.94	0.014	To accept in return for the imposition of
MC203	2053.4	137	1958.14	139	−4.86	0.000	To accept in return for the imposition of
MR101	4257.25	112	4083.84	176	−4.25	0.005	To accept in return for the imposition of
MR102	3905.81	153	3829.73	197	−1.99	0.072	Re impose zero
MR103	3391.64	142	3297.53	160	−2.85	0.042	To accept in return for the imposition of
MR201	2616.21	146	2458.96	150	−6.39	0.000	To accept in return for the imposition of
MR202	2430.31	131	2384.63	155	−1.92	0.132	Re impose zero
MR203	2210.83	127	2051.59	154	−7.76	0.000	To accept in return for the imposition of
MRC101	4033.47	124	3907.27	157	−3.23	0.031	To accept in return for the imposition of
MRC102	3800.9	126	3519.61	149	−7.99	0.013	To accept in return for the imposition of
MRC103	3313.4	124	3290.72	168	−0.69	0.124	Re impose zero
MRC201	2946.97	95	2918.49	133	−0.98	0.072	Re impose zero
MRC202	2764.36	133	2784.52	147	0.72	0.365	Re impose zero
MRC203	2435.31	102	2454.5	138	0.78	0.616	Re impose zero

4.3 Performance Evaluation of the Proposed Algorithm in Solving the CLRP Problem

In order to test the performance of the proposed algorithm, we used the Prodhon's problems that are available in [16]. In these examples both the routs and depots are capacitated. The algorithm is run 10 times and the best obtained solution is compared to the solutions of previous algorithms which are available in the literature. The best available algorithms in the literature for solving CLRP problem include: GRASP algorithm [2], MAPM algorithm [3], LRGTS [5] algorithm, GRASP + ELS algorithm [7] and SALRP algorithm [8].

Table 7 Results of proposed algorithm in comparison with other algorithms

Code example	BKS	GRASP	MAPM	LRGTS	GRASP+ ELS	SALRP	Proposed method	CPU (sec)	GAP (%)
20-5-1a	54,793	55,021	54,793	55,131	54,793	54,793	54,793	25	0.00
20-5-1b	39,104	39,104	39,104	39,104	39,104	39,104	39,104	22	0.00
20-5-2a	48,908	48,908	48,908	48,908	48,908	48,908	48,908	33	0.00
20-5-2b	37,542	37,542	37,542	37,542	37,542	37,542	37,542	38	0.00
50-5-1a	90,111	90,632	90,160	90,160	90,111	90,111	90,111	39	0.00
50-5-1b	63,242	64,741	63,242	63,256	63,242	63,242	63,242	40	0.00
50-5-2a	88,298	88,786	88,298	88,715	88,643	88,298	88,298	46	0.00
50-5-2b	67,308	68,042	67,893	67,698	67,308	67,308	67,308	37	0.00
50-5-2bis	84,055	84,055	84,055	84,181	84,055	84,055	84,055	33	0.00
50-5-2Bbis	51,822	52,059	51,822	51,992	51,822	51,822	51,822	48	0.00
50-5-3a	86,203	87,380	86,203	86,203	86,203	86,456	86,203	54	0.00
50-5-3b	61,830	61,890	61,830	61,830	61,830	62,700	61,830	31	0.00
100-5-1a	276,960	279,437	281,944	277,935	276,960	277,035	275,919	245	−0.38
100-5-1b	214,885	216,159	216,656	214,885	215,854	216,002	214,646	289	−0.11
100-5-2a	194,124	199,520	195,568	196,548	194,267	194,124	194,677	381	0.28
100-5-2b	157,150	159,550	157,325	157,792	157,375	157,150	157,265	245	0.07
100-5-3a	200,242	203,999	201,749	201,952	200,345	200,242	200,247	218	0.00
100-5-3b	152,467	154,596	153,322	154,709	152,528	152,467	152,503	291	0.02
100-10-1a	290,429	323,171	316,575	291,887	301,418	291,043	290,919	348	0.17
100-10-1b	234,210	271,477	270,251	235,532	269,594	234,210	233,503	281	−0.30
100-10-2a	244,265	254,087	245,123	246,708	243,778	245,813	244,253	312	0.00
100-10-2b	203,988	206,555	205,052	204,435	203,988	205,312	203,988	237	0.00
100-10-3a	250,882	270,826	253,669	258,656	253,511	250,882	251,120	388	0.09
100-10-3b	204,597	216,597	204,815	205,883	205,087	205,009	205,578	295	0.48

In Table 7, the first column shows the code for each example. The first and second characteristic numbers of the example indicate number of customers and number of candidate depots respectively. The second column shows the best obtained solution so far and the remaining columns show the obtained solution of the selected algorithms in the history of the subject and the best obtained solution of the proposed algorithm. In the end, the run time of CPU in seconds and the deviation from the best obtained solution so far (GAP) are given. Table 7 shows that the proposed algorithm in this paper, was able to achieve the best obtained solution so far for 15 out of 24 cases, it means that the deviation percentage from the best obtained solution is 0 %.

The proposed algorithm in 3 cases has obtained results better than the best obtained solution so far. The proposed algorithm has deviation between 0.02 and 0.48 % in the 6 remaining examples. Totally, the average deviation of the best obtained solution for the 24 chosen benchmark examples is equal to 0.01 %. The quality of the obtained solutions and the reasonable time to achieve the solutions by the proposed algorithm indicate that it can compete with other well-known algorithms for solving the capacitated location-routing problem without time window constraint.

5 Conclusion

The capacitated location-routing problem is an important and widely applied problem in management of supply chain and logistics. In this paper, in order to make the problem more practical and realistic, the time window constraint is added to the problem. In this paper, after setting the required assumptions, the problem was modeled with all of its constraints and assumptions. Next, by considering the exponential complexity of time for large problems and inability of exact optimization methods in solving large size problems, a meta-heuristic method was applied to solve the problem. The applied method to solve the problem is an approach based on variable neighborhood decline algorithm. Since there was no benchmark example to assess the performance of the algorithm, a number of examples were generated. Next, the performance of the proposed combinatory algorithm for capacitated-location routing problem with hard time windows and the validity of the presented model for this problem were assessed. at the end, the algorithm performance for solving capacitated location-routing problem without considering time windows was tested. The obtained results demonstrated that the algorithm was capable of achieving appropriate solutions in a reasonable time.

References

1. Nagy G, Salhi S (2007) Location-routing: issues, models and methods. Eur J Oper Res 177(2):649–672
2. Prins C, Prodhon C, Calvo RW (2006) Solving the capacitated location- routing problem by a GRASP complemented by a learning process and a path relinking. A Q J Oper Res 4(3):221–238
3. Prins C, Prodhon C, Calvo RW (2006) A memetic algorithm with population management (MA|PM) for the capacitated location-routing problem. Lecture notes in computer science, vol 3906. Springer, Berlin, pp 183–194
4. Barreto S, Ferreira C, Paixa J, Santos BS (2007) Using clustering analysis in capacitated location-routing problem. Eur J Oper Res 179:968–977
5. Prins C, Prodhon C, Ruiz A, Soriano P, Calvo RW (2007) Solving the capacitated location-routing problem by a cooperative Lagrangean relaxation granular tabu search heuristic. Transp Sci 41:470–483
6. Marinakis Y, Marinaki M (2008) A particle swarm optimization algorithm with path relinking for the location routing problem. J Math Model Algorithms 7:59–78
7. Duhamel C, Lacomme P, Prins C, Prodhon C (2010) A GRASP×ELS approach for the capacitated location-routing problem. Comput Oper Res 37:1912–1923
8. Yu VF, Lin S-W, Lee W, Ting C-J (2010) A simulated annealing heuristic for the capacitated location-routing problem. Comput Ind Eng 58:288–299
9. Nguyen V-P, Prins C, Prodhon C (2012) Solving the two-echelon location routing problem by a GRASP reinforced by a learning process and path relinking. Eur J Oper Res 216:113–126
10. Nguyen V-P, Prins C, Prodhon C (2012) A multi-start iterated local search with tabu list and path relinking for the two-echelon location-routing problem. Eng Appl Artif Intell 25:56–71
11. Jabal-Ameli MS, Ghaffari-Nasab N (2010) Location-routing problem with time windows: novel mathematical programming formulations. In: 7th International industrial engineering conference. Isfahan, Iran
12. Mladenovi'c N, Hansen P (1997) Variable neighborhood search. Comput Oper Res 24:1097–1100
13. Kirkpatrick S, Gelatti CD, Vecchi MP (1983) Optimization by simulated annealing. In: Science is currently published by American association for the advancement of science, vol 220, no. 4598, pp 671–680
14. Solomon MM (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. Oper Res 35:254–265
15. http://web.cba.neu.edu/~msolomon
16. http://prodhonc.free.fr/homepage