An Augmented Variable Neighborhood Search to Solve the Capacitated Location-Routing Problem

Alireza Mohamadi-Shad, Hamed Niakan, and Hasan Manzour

Abstract — In this paper we proposed a new variable neighborhood search (VNS) for solving the location-routing problem with considering capacitated depots and vehicles. A set of capacitated vehicles, a set of depots with restricted capacities, and associated opening costs, and a set of customers with deterministic demands are given. The problem aims to determine the depots to be opened, fleet assignment to each depot, and the routes to be performed to satisfy the demand of the customers. The objective is to minimize the total costs of the open depots, the setup cost associated with the used vehicles, and transportation cost. We proposed a new VNS which is augmented with a probabilistic acceptance criterion as well as a set of efficient local searches. The computational results implemented on four well-known data sets demonstrate that the proposed algorithm is competitive with other well-known algorithms while reaching many best-known solutions and updating six best new results with reasonable computational time. Conclusions and future research avenues close the paper.

Index Terms — Capacitated location routing, variable neighborhood search, heuristic, local search, simulated annealing, dynamic penalty function.

I. INTRODUCTION

In this paper, we focus on Capacitated Location-Routing Problem (CLRP) where restriction on both depots and vehicles in terms of capacity is taken into account. The CLRP stated in this paper can be represented on a complete, weighted, and undirected network \( G=(V, E) \). \( V=I \cup J \) includes a set of \( m \) potential nodes for depots indexed by \( I=\{1, 2, ..., m\} \) and a set \( n \) customers indexed by \( J=\{1, 2, ..., n\} \). Each depot, \( i \in I \), is characterized by a limited capacity and associated fixed cost. Each customer, \( j \in J \), has a non-negative predefined demand which should be satisfied. We assume that using a vehicle (i.e., starting a route) incurs a fixed cost, and there is a travel cost to move from node \( i \) to node \( j \). The goal is to determine which depots need to be opened, how to allocate vehicles to the depots, and how to dispatch vehicles to fulfill customers so that the total system cost is minimized without violating the constraints.

The total system cost encompasses three components including the opening cost of established depots, fixed cost of incurring vehicles (cost of starting a tour), and the total transportation cost. The following constraint must be met (i) each customer must be assigned to one and only one vehicle (ii) each vehicle must begin and end at the same depot and its total load must not exceed vehicle capacity (iii) the total demand of customers allocated to a depot must not exceed capacity of the depot. Note that CLRP is a NP-hard problem, as it encompasses two NP-hard problems (facility location and vehicle routing).

LRP has received a great deal of attention in literature. Interested reader can refer to Prodhon and Prins [20], Lopes et al. [10], and Nagy and Salhi [14] as valuable surveys for LRP. We briefly mention those studies on LRP which are closely related to ours i.e., CLRP. Perl and Daskin [15], [16] develop a sequential algorithm to solve CLRP in which the problem is decomposed into three sub-problems. Later, the proposed method is modified by Hansen et al. [8], and Wu et al. [23]. Tuzun and Burke [22] develop a two-phase tabu search heuristic to cope with this problem. Prins et al. [18] propose a metaheuristic wherein Greedy Randomized Adaptive Search Procedure (GRASP) is executed in the first phase, based on an extended and randomized version of Clarke and Wright’s algorithm. In the second phase, path relinking as a post optimization is used to enhance the results even further. Prins et al. [17] hybrid a memetic algorithm with population management (MA|PM) to tackle the problem. Prins et al. [19] combine the Lagrangian relaxation technique with Granular Tabu Search (GTS) in an iterative manner to solve the CLRP. The algorithm alternated between a depot location phase and a routing phase, exchanging information on the most promising edges. Barreto et al. [3] develop a sequential method to solve the problem. They firstly clustered customers according to the capacity of vehicles, and then solve a TSP for each cluster. Then, they assign tours to depots. Marinakis and Marinaki [11] combine a particle swarm optimization (PSO) algorithm, greedy randomized adaptive search procedure (MPNS-GRASP) algorithm, the expand- ing neighborhood search (ENS) strategy and a path relinking (PR). Marinakis et al. [12] modified their approach by using honeybees mating optimization (HBMO) algorithm instead of PSO. Yu et al. [24] introduce a hierarchical heuristic algorithm based on Simulated Annealing (SA) algorithm. In order to better explore the solution space, they use a special encoding scheme and three neighborhood structures. Duhamel et al. [4] integrate a GRASP and Evolutionary Local Search (ELS), called GRASP+ELS, for solving CLRP. They tested their proposed algorithms on three different well-known benchmark instances with or without limited capacities on depot. Escobar et al. [5] develop a two-phase hybrid heuristic algorithm (2-Phase HGTGTS) to tackle the CLRP. In the first phase, known as Construction phase, a hybrid

Submitted on February 23, 2021. Published on June 10, 2021.
Alireza Mohamadi-Shad, Department of Industrial Engineering, Bu-Ali Sina State University, Iran.
Hamed Niakan, Department of Industrial & Systems Engineering, Wayne State University, USA.
(e-mail: hniakan@wayne.edu)
Hasan Manzour, Department of Industrial and Systems Engineering, University of Washington, USA.

DOI: http://dx.doi.org/10.24018/ejers.2021.6.4.2380

Vol 6 | Issue 4 | June 2021
method capable of generating good initial solution is developed. Then, during the second phase, i.e., intensification phase, a modified granular tabu search along with a random perturbation procedure is applied. Ting and Chen [21] apply a hierarchical Multiple Ant Colony Optimization algorithm (MACO) to solve CLRP. Fazelizadeh et al. [7] introduced a min-max autonomous vehicle routing problem which seems promising for solving such transportation network problems and determining the most efficient routing strategies. In his other paper, Fazelizadeh et al. [6] proposed a choice modeling approach embedded in a two-stage stochastic programming mode which is very practical to efficiently solve big network/optimization problem. Although VNS has been claimed to be a really effective metaheuristic for combinatorial optimizations Mladenovic’ and Hansen [13], it has not been applied for CLRP yet. However, Jarboui et al. [9] embed a variable neighborhood descent as the local search into the general variable neighborhood heuris- tic framework to solve the LRP with uncapacitated vehicles and obtained promising results. Hence, the main aim of the paper is to develop an augmented variant of VNS suitable for CLRP.

The rest of this paper is outlined as follows. The proposed solution approach is presented in Section 2. Computational results are presented and analyzed in Section 3, and finally we close the paper by providing some future research avenues.

II. SOLUTION FRAMEWORK

The proposed algorithm to solve the CLRP is built based upon Variable Neighborhood Search (VNS) introduced by Mladenovic’ and Hansen [13]. In order to diversify the generated solutions, a simulated annealing acceptance criterion is embedded into the framework. Moreover, a dynamic penalizing strategy allowing the search to be oscillated between feasible and infeasible part of solution space is introduced to avoid trapping into the local optima. A new local search in addition to some other local searches are also embedded into the proposed VNS.

The VNS is a simple but effective approach. The main idea of VNS comes from this fact that a global optimum is actually the best local optima. Thus, VNS aims to alter the neighborhood whenever a local optimum is obtained. Hence, VNS is principally a local search tool that has the capability of escaping from trapping into local optima by searching a set of different neighborhoods to find a better local optima. At the next subsection, we briefly outline the VNS structure. Another innovative but very practical approach for solving capacitated location-routing problem is presented by Ansaripour and Trafalis [1]. They used a robust optimization model with box constraints for city logistic terminal locations. This approach can be useful to manage the uncertain demand in the market!

A. The Basic of VNS

The VNS composves three basic steps: i) shaking step, that is a stochastic phase which finds a random neighbor of the incumbent solution, ii) local search phase, which aims to find a local optimum usually in a deterministic fashion phase, and iii) the acceptance step, which admits a solution that improves the objective. Practically, implementing the VNS algorithm requires to (1) find an initial solution, (2) define a set of neighborhoods (shaking phase) (3) define a criterion to determine when to perform change of neighborhood and (4) define the method to explore each of them (local search phase).

Suppose that there are \( k_{\text{max}} \) neighborhood structures \( k = \{1, 2, ..., k_{\text{max}}\} \) such that \( N_k \) is the \( k \)th neighborhood type. VNS begins with an initial solution, \( S \) (current local optima). VNS encompasses two main loops. The former is the shaking loop with which solution \( S \) moves from one point to a neighborhood point, say \( S^\prime \). In the latter one, the local search phase (internal loop), the solution \( S \) is improved based on some local searches or a defined mechanism to reach to a local optima, say \( S^\prime \). After terminating the local search phase, the obtained local optima are compared to the \( S \) i.e., the best local optima so far. If \( S^\prime \) is better than \( S \), \( S^\prime \) is substituted with \( S \) (Move-or-Not phase) and the search reverts back to shaking phase, and the first neighborhood search applies. Otherwise, the next neighborhood search, i.e., \( N_{k+1} \), in the shaking phase is applied on \( S \). The algorithm is terminated when \( k \) exceeds \( k_{\text{max}} \).

B. An Overview of the Proposed Algorithm

The proposed algorithm begins with an initial solution, say \( S = S_0 \). A random solution is chosen initially. Then, the main loop of the algorithm (i.e., shaking loop) starts with the first neighborhood structure i.e., \( k = 1 \), and the initial solution is converted to \( S^\prime \). The set of neighborhood searches is defined later. Next, the internal loop (i.e., local search loop) is applied to find a local optimum, say \( S^\prime \). In contrast to the classical VNS where \( S \) is substituted with \( S^\prime \) (\( S = S^\prime \)) only when \( S^\prime \) produces better result, in this paper, we exploit a simulated annealing criterion to determine when to accept a solution. In fact, an inferior solution in move-or-not phase can be chosen according to the new defined acceptance criteria. This enables the algorithms to explore the solution space more. The main steps of our algorithm are given in Fig. 2, but the explanations of these steps are elaborated in the following subsections.

C. Explanation of the Main Steps

1. Solution representation and cost function

In this paper, we used a simple solution representation. An array with the length of \( L \) is used to encode a solution. The set of indices \( \{1, 2, ..., n\} \) denotes the customers, the set of indices \( \{n+1, n+2, ..., n+m\} \) denotes the depots and a set of dummy zeros is used to separate routes belonging to the same depots. As illustrated in Fig. 3, we can assume an instance with 7 customers and 3 candidate depots. In this example, the first route starts from depot 11 and serves three customers (i.e., 3, 5, 7) and goes back to its depot. The
second tour associated with depot 11 fulfills the second and first customers, respectively. The depot 12 is closed as there is no customer between depot 12 and 13. Similarly, customers 8,6 and 10,9 are fulfilled by the depot 13 using two separate routes. An illustration is given in Fig. 4. Similar encoding scheme is used. The main benefit of this encoding is to be capable of producing very diverse solutions which helps to explore solution space whereas it may increase the computational time.

Mathematically speaking, the cost function associated with a solution, say $S$ can be obtained by the following formula:

$$
\eta(S) = C(S) + \beta_1 \xi_1(S) + \beta_2 \xi_2(S)
$$

where $C(S)$ is the value of the objective function. $\xi_1$, $\xi_2$ are the value of the violations associated with the vehicle and depot capacity, respectively. $\beta_1$, $\beta_2$ are the penalty coefficients corresponding to one unit violation of vehicle and depot capacity, respectively. At step (6), these coefficients are updated after changing the temperature with the factor of $(1+\delta_1)$ and $(1+\delta_2)$, respectively. In addition, one can use the new techniques presented by Ansaripour et al. [2], to deal with the stochastic aspects of the problem when it arises. This approach would be very helpful to find a close form expression for the cumulative distribution function of the maximum value of the objective function which is not very easy to solve by other existing solutions in the literature!

2. Shaking phase (Step 3-1 of Fig. 3)

The proposed algorithm utilizes four neighborhood structures (i.e., $k_{max} = 4$) for both shaking and local search phases. The first three local searches are used within the domain of VRP while the last one is introduced in this paper. The order of the neighborhoods which is chosen after some experimentation is as follows: the relocation is used as $N_1$, the SWAP is used as $N_2$, the inversion is chosen as $N_3$, and Combined Relocation and SWAP move (CRS) is the last neighborhood search.

In the first neighborhood search, two random numbers are generated e.g., $i, j$ and the $i$th element is removed from its position and inserted immediately after the $j$th element of the solution representation. As illustrated in Fig. 5a, two elements 10 and 12 (i.e., 4, 6 customers) are chosen, and the fourth customer is removed and inserted after the sixth customer. In the next neighborhood search, known as SWAP, two points are randomly chosen, and the positions of those are inter-changed. As illustrated in Fig. 5b, the positions of fifth and seventh elements are changed. The inversion neighborhood search selects two elements randomly and then change the positions of those, and also inverse the sequence of all elements among those. As illustrated in Fig. 5c the fourth and ninth elements are changed and the order of elements among those are reversed. In the last neighborhood search, CRS, at first two elements, say $i, j$, are randomly chosen. In addition, two random numbers $\mu_1$, $\mu_2$ in the interval $[0, k]$ are randomly selected. The selected random number must satisfy these constraints $i + \mu_1 \leq j$, $j + \mu_2 \leq L$. Then, the position of $i$th element and next $\mu_1$ elements after $i$ are altered with the position of $j$th element as well as $\mu_2$ elements after $j$. For instance, in Fig. 5d-1, $i = 3$, $j = 9$, $\mu_1 = 0$, $\mu_2 = 3$. The Fig. 5d-2 shows another example where $i = 3$, $j = 9$, $\mu_1 = 2$, $\mu_2 = 2$. In this local search, if $\mu_1=0$, $\mu_2 = 0$, the local search reduces to SWAP move.
3. Local search phase (Step 3-2 of Fig. 3)

After the shaking phase, we need to take into account the local search phase. In the local search phase, the proposed algorithm uses a simple and straightforward procedure to avoid extensive computational time. That is, the algorithm is devised such that the local search phase takes advantage of the same neighborhood structures used in the shaking phase. The local search phase is repetitively performed until there is no improvement in $f$ consecutive iterations. The procedure is presented in Fig. 6.

The main philosophy of this approach is that decreasing the objective function monotonically increases the chance of trapping into a local optimum. Thus, by relaxing this constraint and providing this flexibility for the algorithm to accept a worst solution according to a probability distribution (i.e., simulated annealing acceptance criterion), we can increase the chance of enhancing the results even further.

In addition, a monotonically increasing coefficient of penalty function enables algorithms to explore the solution space wisely without extensive computational time.

III. COMPUTATIONAL RESULTS

In this section, the computational results of the proposed VNS algorithm are presented. The proposed VNS algorithm is coded in Visual C++ and run on a Core 2 duo processor at 2.00 GHz with 1 GB of RAM under Windows 7 operating system. A set of preliminary experiments are performed to determine a proper set of parameters producing satisfactory results across most instances even though they are not optimal for all instances. According to these experiments, the parameters are set as follows: $T_0 = 50$, $T_f = 0.001$, $\alpha = 0.92$, $I = L$ ($L - 1$), $\beta_1 = 5$, $\beta_2 = 10$, $\delta_1 = \delta_2 = 0.02$. Please note that $L$ refers to the length of array associated with the initial solution.

We implemented the proposed Hybrid VNS (HVNS) on three different set of benchmark instances from the literature, (i) Prodon’s instances, (ii) Barreto’s instances, (iii) Tuzan and Burke’s data sets, containing a total of 79 instances, with the same parameter settings.

In order to assess the effectiveness of the proposed algorithm, it is compared with seven of the most recently published algorithms: The greedy adaptive randomized search procedure (GRASP) developed by Prins et al. [18], the memetic algorithm with population management (MA|PM) by Prins et al. [17], the lagrangean relaxation granular tabu search (LRGTS) proposed by Prins et al. [19], the heuristic simulated annealing algorithm (SALRP) developed by [24], the GRASP hybridized with an

| a | 1 | 1 | 3 | 5 | 7 | 0 | 2 | 1 | 12 | 13 | 8 | 6 | 0 | 10 | 9 | 0 | 0 |
| b | 1 | 1 | 3 | 5 | 7 | 0 | 2 | 1 | 12 | 13 | 8 | 6 | 0 | 10 | 9 | 0 | 0 |
| c | 1 | 1 | 3 | 5 | 7 | 0 | 2 | 1 | 12 | 13 | 8 | 6 | 0 | 10 | 9 | 0 | 0 |
| d-1 | 1 | 1 | 3 | 5 | 7 | 0 | 2 | 1 | 12 | 13 | 8 | 6 | 0 | 10 | 9 | 0 | 0 |
| d-2 | 1 | 1 | 3 | 5 | 7 | 0 | 2 | 1 | 12 | 13 | 8 | 6 | 0 | 10 | 9 | 0 | 0 |
| d-3 | 1 | 1 | 3 | 5 | 7 | 0 | 2 | 1 | 12 | 13 | 8 | 6 | 0 | 10 | 9 | 0 | 0 |

Fig. 5. (a) relocation move. (b) SWAP move. (c) Inversion move. (d-1,2) CRS move.

Fig. 6. Local search phase of the proposed algorithm.

4. Simulation Annealing Criterion (Step 3-3 of Fig. 3)

The cost function of the obtained local optima in classical VNS is non-increasing. In this paper, however, we embed a probabilistic criterion enabling the algorithm to accept a local minimum even though it is inferior compared to the best obtained local minimum. In fact, if obtained local minimum is better than the best local minimum, it is substituted. Then, the algorithm reverts back to the main loop and resumes from the first neighborhood structure. Otherwise, making decision about substitution is according to a probability function. Thus, $S = S'$ if $r < \text{exp}(-\Delta/T)$ where $r$ is a random number between $(0,1)$, and $\Delta$ is absolute cost between $S$ and $S'$. $T$ is the current system temperature, and the search starts from the first neighborhood. Otherwise, the next neighborhood structure is applied on $S$ and the local search phase is performed again. This process is repetitively done till $k \leq k_{\text{max}}$. The VNS algorithm is done for a specific temperature, then the temperature is updated and the whole VNS algorithm restarts.

Step 1) Define the set of neighborhood structures, and the maximum number of consecutive iterations without improvement, say $f$

Step 2) $S$ is the neighborhood solution of an iteration

Step 3) repeat the following steps while $2at \leq 1$

$S_{t} = \text{NewMove}$

if $f(S_{t}) < f(S)$

$S_{t} = S_{t}$

else $f(S_{t}) = f(S)$

Step 4) return $S_{t}$ as a local minimum.

DOI: http://dx.doi.org/10.24018/ejert.2021.6.4.2380
The five first proposed algorithms are robust, and the authors just ran one run. The robustness means that the solution obtained by these algorithms has relatively similar performance over multiple runs. Nevertheless, Duhamel et al. [4] performed 5 runs for GRASP×ELS and Ting and Chen [21], performed 10 runs for MACO. In fact, the multiple runs could improve the quality of the obtained results. In this paper, we provided both single run and multiple run (10 runs) to provide comprehensive and fair results in comparison to all other proposed algorithms.

The results are elaborated in Tables I-VII. Two tables are given for each data set, and for the Prodhon’s instances, we added another table to show the contribution of each component of the proposed solution. The first table of each data set is associated with the result of the proposed method and the best given solutions in the literature, and the second reports comparison results. In the first table of each data set, the columns show the name of instances (Problem ID), number of customers (n), number of depots (m), capacity of vehicles (Q), the best given solution in the literature (BKS), objective function for 1 and 10 runs (cost), corresponding computational time (CPU) and gap (GAP), respectively. The GAP is calculated by:

\[
GAP = \frac{C - BKS}{BKS} \times 100
\]

where \( C \) is the associated cost value. The last column reports the Coefficient of Variation (CV) that is a good indicator of robustness of the proposed algorithm. The second table of each data set composes the problem ID and GAP associated with the seven aforementioned proposed algorithms in the literature in addition to the proposed algorithm in this paper. Moreover, the last two rows denote the average gap and the number of best solutions obtained by each method for each method. If the obtained solution is the same as BKS or better than the best-known solution, we bold it in the Tables I, III, and V. In addition, the best obtained solutions among the proposed algorithm and the considered seven other algorithms are bold in Tables II, IV, and VI. Note that the BKS does not necessarily come from these seven last published papers or the proposed algorithm in this paper. The detail information about each data set is elaborated as follows.

A. Prodhon’s Instances

The Prodhon’s set contains 30 LRP instances with capacitated routes and depots. The complete data sets are available at http://prodhonc.free.fr/. One attribute of this data set is that it contains the largest instances with capacitated depots (10 depots and 200 customers). The number of depots is either 5 or 10, the number of customers is chosen from the set \{20, 50, 100, 200\} and vehicle capacities are either 70 or 150. The vehicle fixed cost is considered in this group of instances. Moreover, another attribute of this data set is that it contains both uniform and clustered customers distribution (either two clusters or three). The demand of customers is generated in a uniform distribution in interval [10], [20].

Table I summarizes the computational results of the proposed algorithm associated with Prodhon’s instances. Table II is dedicated to the comparison results. Table III reports the contribution of each component of the proposed VNS in refining the objective function. According to Table I, the average gaps associated with single and multiple runs are 0.54 and 0.04. The single and multiple runs HVNS reaches 11 and 24 over 30 instances of best-known solutions in the literature respectively. In addition, the HVNS with single run updated the solution of one instance (i.e., 200-10-1b) and multiple runs HVNS updates the BKS for 5 instances. The obtained results clearly show the high quality performance of the proposed method in terms of generating promising results.

According to Table II, the proposed multiple runs HVNS reaches 27 over 30 best solutions among these 7 tested algorithms. Quite evidently, the proposed algorithm dominates all other previous algorithms in terms of both average solution quality and the number of best solutions. The proposed algorithm did not find the best results only in a few cases with negligible gap. In respect to single run, the proposed algorithm is competitive with those papers with multiple runs (i.e., 2-Phase HGTS and MACO) while dominates all other single run algorithms in terms of average gap.

As can be seen in Table III, the value of adding different components of the proposed algorithm is reported. The GAPs between the HVNS and the HVNS without SA, new local search, and penalizing strategy are reported in Table III. It indicates that all components have had significant impact on improving the quality of the solution. The SA seems to be most important component while the local search and penalizing strategy are the next important components, respectively.

Since the computational time depends on many factors (e.g., machines, compiler, programing language, etc.) it is difficult to compare the run time directly. However, we would like to emphasize that the machine we used is at least 3 times slower than the one used by Ting and Chen [21], Escobar et al. [5]. Considering this fact, the proposed algorithm is competitive (e.g., Ting and Chen [21], Escobar et al. [5]) in terms of computational time.
| Problem ID | GRASP MA | M | LRTGS | SALRP | GRASP\*ELS | 2-Phase HRTGS | MACO | HV | S |
|------------|----------|---|--------|--------|------------|----------------|------|----|---|
| 20-5-1a    | 0.42     | 0 | 0      | 0.62   | 0.00       | 0.00           | 3    | 0  | 0 |
| 20-5-1b    | 0.00     | 0 | 0      | 0.00   | 1.00       | 0.00           | 4    | 0  | 0 |
| 20-5-2a    | 0.00     | 0 | 0      | 0.00   | 1.00       | 0.00           | 5    | 0  | 0 |
| 20-5-2b    | 0.00     | 0 | 0      | 0.00   | 0.00       | 4.00           | 3    | 0  | 0 |
| Ave.       | 0.10     | 0 | 0.15   | 0.17   | 0.00       | 0.00           | 4    | 0  | 0 |
| 50-5-1a    | 0.58     | 2 | 0.05   | 0.05   | 0.00       | 0.00           | 3    | 0  | 0 |
| 50-5-1b    | 2.37     | 0 | 0.00   | 0.00   | 0.00       | 0.00           | 27   | 0  | 0 |
| 50-5-2a    | 0.55     | 0 | 3.47   | 2.00   | 0.00       | 0.00           | 21   | 0  | 0 |
| 50-5-2b    | 1.09     | 3 | 0.87   | 0.58   | 0.00       | 0.00           | 37   | 0  | 0 |
| 50-5-2bis  | 0.00     | 0 | 0.00   | 0.00   | 0.00       | 0.00           | 25   | 0  | 0 |
| 50-5-2Bbis | 0.46     | 0 | 0.00   | 0.00   | 0.00       | 0.00           | 17   | 0  | 0 |
| Ave.       | 0.41     | 0 | 0.12   | 0.20   | 0.22       | 0.00           | 17   | 0  | 0 |
| 100-5-1a   | 1.68     | 28 | 2.59  | 0.14   | 0.81       | 0.00           | 157  | 0.51 | 0.00 |
| 100-5-1b   | 1.19     | 23 | 1.42  | 0.35   | 1.12       | 0.00           | 60   | 0.33 | 0.00 |
| 100-5-2a   | 3.02     | 17 | 0.98  | 0.36   | 0.23       | 0.00           | 148  | 0.50 | 0.00 |
| 100-5-2b   | 1.56     | 22 | 0.15  | 0.44   | 0.04       | 0.00           | 148  | 0.50 | 0.00 |
| 100-5-3a   | 1.96     | 23 | 0.83  | 0.29   | 0.08       | 0.00           | 148  | 0.50 | 0.00 |
| 100-5-3b   | 1.84     | 22 | 0.19  | 0.30   | 0.38       | 0.00           | 148  | 0.50 | 0.00 |
| Ave.       | 1.81     | 28 | 0.12  | 0.30   | 0.22       | 0.00           | 148  | 0.50 | 0.00 |

**TABLE I: PRODIHN'S Data Set, Comparison with BKS**

| problem ID | n | m | Q | BKS | Cost | CPU | GAP | C. V |
|------------|---|---|---|-----|------|-----|-----|-----|
| 20-5-1a    | 5 | 70 | 150 | 54,793 | 54,793 | 19  | 25  | 0.00 |
| 20-5-1b    | 5 | 70 | 150 | 39,104 | 39,104 | 19  | 22  | 0.00 |
| 20-5-2a    | 5 | 70 | 150 | 48,908 | 48,908 | 33  | 33  | 0.00 |
| 20-5-2b    | 5 | 150 | 37,542 | 37,542 | 37  | 38  | 0.00 |
| 50-5-1a    | 5 | 70 | 90,111 | 90,111 | 43  | 39  | 0.00 |
| 50-5-1b    | 5 | 70 | 150 | 63,242 | 63,242 | 48  | 40  | 0.00 |
| 50-5-2a    | 5 | 70 | 88,298 | 88,298 | 51  | 46  | 0.00 |
| 50-5-2b    | 5 | 70 | 67,308 | 67,308 | 37  | 36  | 0.00 |
| 50-5-2b1   | 5 | 70 | 84,055 | 84,055 | 43  | 33  | 0.00 |
| 50-5-2b2   | 5 | 70 | 51,822 | 51,822 | 41  | 48  | 0.00 |
| 50-5-3a    | 5 | 70 | 86,203 | 86,203 | 65  | 54  | 0.00 |
| 50-5-3b    | 5 | 70 | 62,061 | 61,830 | 38  | 37  | 0.00 |
| 50-5-3b1   | 5 | 70 | 200,079 | 200,079 | 254 | 182 | 0.00 |
| Ave.       | 3.60 | 96 | 1.38 | 0.74 | 0.41 | 0.22 | 0.53 | 0.17 |

**TABLE II: PRODIHN's Instances, Comparison with Other Algorithms**

| Problem ID | CPU | GAP |
|------------|-----|-----|
| 20-5-1a    | CPU | 0.00 |
| 20-5-1b    | CPU | 0.00 |
| 20-5-2a    | CPU | 0.00 |
| 20-5-2b    | CPU | 0.00 |
| Ave.       | CPU | 0.00 |
| 50-5-1a    | CPU | 0.00 |
| 50-5-1b    | CPU | 0.00 |
| 50-5-2a    | CPU | 0.00 |
| 50-5-2b    | CPU | 0.00 |
| Ave.       | CPU | 0.00 |
| 100-5-1a   | CPU | 0.00 |
| 100-5-1b   | CPU | 0.00 |
| 100-5-2a   | CPU | 0.00 |
| 100-5-2b   | CPU | 0.00 |
| Ave.       | CPU | 0.00 |

**European Journal of Engineering and Technology Research**

ISSN: 2736-576X
TABLE III: CONTRIBUTION OF EACH COMPONENT OF VNS

| problem ID | HVNS | HVNS with SA | HVNS with new local search | HVNS without dynamic penalizing |
|------------|------|--------------|----------------------------|---------------------------------|
|            | Cost | Cost         | Cost                       | Cost                            |
| 20-5-1a    | 54,793 | 54,793       | 8.00                       | 54,793                          |
| 20-5-1b    | 39,104 | 39,104       | 0.00                       | 39,104                          |
| 20-5-2a    | 48,908 | 48,908       | 15.00                      | 48,908                          |
| 20-5-2b    | 37,542 | 37,542       | 19.00                      | 37,542                          |
| 50-5-1a    | 90,111 | 91,894       | 25.19                      | 91,166                          |
| 50-5-1b    | 63,242 | 63,242       | 31.00                      | 63,242                          |
| 50-5-2a    | 88,208 | 88,838       | 41.61                      | 88,786                          |
| 50-5-2b    | 67,308 | 67,449       | 23.01                      | 67,484                          |
| 50-5-3a    | 84,055 | 84,172       | 27.14                      | 84,110                          |
| 50-5-2Bbs | 51,822 | 52,059       | 33.46                      | 52,059                          |
| 50-5-3a    | 86,203 | 89,914       | 49.30                      | 86,757                          |
| 50-5-3b    | 61,830 | 61,885       | 28.09                      | 61,830                          |
| 100-10-1a  | 275,066 | 283,802     | 212.38                     | 282,357                          |
| 100-10-1b  | 213,618 | 217,205     | 155.12                     | 214,484                          |
| 100-10-2a  | 193,671 | 195,156     | 142.77                     | 195,120                          |
| 100-10-2b  | 157,129 | 157,636     | 134.32                     | 157,178                          |
| 100-10-3a  | 200,079 | 201,474     | 170.70                     | 200,851                          |
| 100-10-3b  | 152,441 | 155,022     | 315.19                     | 154,429                          |
| 100-10-1a  | 288,843 | 295,855     | 101.23                     | 294,845                          |
| 100-10-1b  | 231,028 | 236,674     | 149.24                     | 234,526                          |
| 100-10-2a  | 243,590 | 246,586     | 130.12                     | 245,099                          |
| 100-10-2b  | 203,988 | 206,379     | 112.17                     | 206,158                          |
| 100-10-3a  | 330,882 | 325,480     | 149.13                     | 324,064                          |
| 100-10-3b  | 264,158 | 260,809     | 134.18                     | 260,799                          |
| 200-10-1a  | 476,516 | 489,339     | 954.27                     | 484,841                          |
| 200-10-1b  | 377,809 | 389,516     | 960.31                     | 389,176                          |
| 200-10-2a  | 449,470 | 460,655     | 1050.24                    | 458,991                          |
| 200-10-2b  | 375,159 | 380,581     | 1132.14                    | 380,997                          |
| 200-10-3a  | 472,962 | 488,191     | 940.32                     | 486,431                          |
| 200-10-3b  | 364,737 | 376,791     | 1011.30                    | 374,593                          |

B. Barreto’s Instances

The first set is given by Barreto et al. [3], containing 13 instances available at http://sweet.ua.pt/~sbarreto/. Their traveling cost is assumed to be the Euclidean distances (not rounded), and all routes are capacitated. An interesting feature of this data set is the existence of both lower and upper bounds provided in Barreto et al. [3]. These instances are either from the literature or are modified by adding both capacitated and incapacitated depots to the classical VRP instances. The number of candidate sites varies from 5 to 10 while the numbers of customers vary from 21 to 150. The vehicle variable cost is not considered in this set of instances.

TABLE IV: BARRETO’S DATA SET, COMPARISON WITH BKS

| problem ID | n   | m   | Q  | BKS | Cost | CPU | GAP |
|------------|-----|-----|----|-----|------|-----|-----|
|            | 5   | 5   | 160| 565.6| 565.6| 16  | 0.00|
|            | 75  | 10  | 140| 844.4| 854.03| 90  | 0.00|
|            | 75  | 10  | 200| 833.4| 847.1| 178 | 0.64|
|            | 88  | 8   | 90000000| 355.8| 356.03| 109 | 0.00|
|            | 150 | 10  | 80000000| 43919.9| 43660.3| 355 | 0.00|
|            | 21  | 5   | 6000| 424.9| 424.9| 8   | 0.00|
|            | 22  | 5   | 4500| 585.1| 585.1| 9   | 0.00|
|            | 29  | 5   | 4500| 512.1| 512.1| 12  | 0.00|
|            | 32  | 5   | 8000| 562.2| 562.2| 9   | 0.00|
|            | 32  | 5   | 11000| 504.3| 504.3| 21  | 0.00|
|            | 36  | 5   | 2500| 460.4| 460.4| 16  | 0.00|
|            | 27  | 5   | 2500| 3062| 3062| 9   | 0.00|
|            | 134  | 8   | 850| 5709| 5716.42| 211 | 2.53|

Table IV summarizes the computational results of the proposed algorithm associated with Barreto’s data set. Taking Table IV into account, the average gaps for single and multiple runs are 0.54, and 0.13, respectively. This proposed method reaches the best existence results in 9 out of 13 instances and for the rest of the instances has relatively good performance with a maximum gap of 0.54%. Multiple runs lead to a decrease in the average gap from 0.54 to 0.13 and also increases the number of best obtained solutions from 8 to 9. Taking into consideration Table IV it can be concluded that the proposed algorithm with multiple runs outperforms all other methods in terms of solution quality with the average gap of 0.13 except GRASP×ELS, and MACO with average gap quality of 0.08 and 0.07, respectively.
C. Tuzun and Burke’s Instances

The third set of instances was first described in Tuzun and Burke [22], that includes 36 instances. The instances are available at http://prod.honc.free.fr/. Different from other two sets of instances, there is no capacity restrictions for facilities. Number of customers are chosen from the set \( n \in \{100, 150, 200\} \), and number of depots are either 10 or 20, vehicle have homogenous capacity (\( Q = 150 \)) and demands are generated uniformly in interval [11, 20]. Traveling costs corresponds to the Euclidean distances and are not rounded.

Table V denotes that the average gap for single and multiple runs are 1.12, and 0.68, respectively. The number of best-known solutions is 2 and 6 for single and multiple runs respectively. In addition, the solution of instance 13-31-22 is improved (\( GAP = -0.04 \)).

According to Table VI, which is presented for comparison with other algorithms, it is quite evident that HVNS with multiple runs outperforms strictly all other selected heuristic approaches both in terms of average gap as well as the number of best solutions. The performance of single run HVNS is not as good as multiple runs such that the average gaps increase from 0.13 to 1.12. The multiple runs of HVNS reaches 10 best solutions among the 7 tested methods.

| Problem ID     | GRASP | MAIPM | LRGTS | SALRP | GRASP+ELS | 2-Phase HTGTS | MACO | HVNS |
|----------------|-------|-------|-------|-------|-----------|---------------|------|------|
| Christofides69-50x5 | 5.92  | 0.00  | 3.68  | 0.00  | 0.00      | 0.00          | 1.11 | 0.00 |
| Christofides69-75x10 | 2.04  | 2.57  | 2.26  | 0.43  | 0.00      | 1.00          | 0.00 | 0.00 |
| Christofides69-100x10 | 3.38  | 2.00  | 1.14  | 0.59  | 0.00      | 0.00          | 0.00 | 0.00 |
| Daskin95-88x88 | 0.31  | 1.77  | 3.63  | 0.00  | 0.00      | 0.00          | 0.00 | 0.00 |
| Daskin95-150x10 | 1.61  | 0.21  | 1.06  | 2.71  | 0.00      | 0.00          | 0.00 | 0.00 |
| Gaskell67-21x5 | 1.11  | 0.00  | 0.00  | 0.00  | 0.00      | 0.00          | 0.00 | 0.00 |
| Gaskell67-22x5 | 0.00  | 0.56  | 0.39  | 0.00  | 0.00      | 0.00          | 0.00 | 0.00 |
| Gaskell67-29x5 | 0.59  | 0.00  | 0.00  | 0.00  | 0.00      | 0.00          | 0.00 | 0.00 |
| Gaskell67-32x5 | 1.73  | 0.00  | 0.00  | 0.00  | 0.00      | 0.00          | 0.00 | 0.00 |
| Gaskell67-36x5 | 0.00  | 0.00  | 0.00  | 0.00  | 0.00      | 0.00          | 0.00 | 0.00 |
| Min92-27x5 | 4.49  | 0.67  | 0.18  | 0.00  | 0.00      | 0.00          | 0.00 | 0.00 |
| Min92-134x8 | 1.63  | 0.20  | 0.26  | 1.76  | 0.00      | 0.00          | 0.00 | 0.00 |

Table VI: Tuzun and Burke’s Data Set, Comparison with BKS

| Problem ID | n   | m   | Q   | BKS  | Cost | CPU  | GAP   |
|------------|-----|-----|-----|------|------|------|-------|
| 11-11-12   | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-22   | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-32   | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-42   | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-52   | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-62   | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-72   | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-82   | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-92   | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-102  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-112  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-122  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-132  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-142  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-152  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-162  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-172  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-182  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-192  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-202  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-212  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-222  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-232  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-242  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-252  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-262  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-272  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-282  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |
| 11-11-292  | 100 | 150 | 140 | 67   | 1.12 | 0.00 | 0.00  |
| 11-11-302  | 100 | 200 | 150 | 0.00 | 1.12 | 0.00 | 0.00  |

Table VII: Summarizes the Computational Results of the Proposed Algorithm Associated with This Data Set
IV. CONCLUSION

In this paper we proposed a hybrid variable neighborhood search (HVNS) to solve the capacitated location-routing problem (CLRP). The proposed algorithm takes advantage of simulated annealing criterion and a dynamic penalty function to explore the solution space more effectively. In addition, a new effective neighborhood search is also introduced. The computational experiments are carried out on three well-known sets of instances from the literature. The proposed HVNS is able to obtain optimal or near-optimal solutions in a reasonable computation time for most benchmark instances. It reaches 21 (single run) and 39 (multiple runs) best known solutions and updates 1 (single run) and 6 (multiple runs) best known solutions in 79 instances considered in this study. The obtained results demonstrate the effectiveness of the proposed method in providing promising results without extensive computational time.

REFERENCES

[1] Afrooz Ansari, Afroz, and Theodoros B. Trafalis. A robust multicriteria optimization model for city logistic terminal locations. In IE Annual Conference. Proceedings, page 2801. Institute of Industrial and Systems Engineers (IIE), 2013.
[2] Afrooz Ansari, Afroz, Adriana Mata, Sara Nourzadeh, and Hillel Kumin, et al. Some explicit results for the distribution problem of stochastic linear programming. Open Journal of Optimization, 5(04):140, 2016.
[3] Sergio Barreto, Carlos Ferreira, and Jose Paixao, and Beatriz Sousa Santos. Using clustering analysis in a capacitated location-routing problem. European Journal of Operational Research, 179(3): 968–977, 2007.
[4] Christophe Duhamel, Philippe Lacomme, and Prins, and Caroline Prodhon. A grasp 1 els approach for the capacitated location-routing problem. Computers & Operations Research, 37(11): 1912–1923, 2010.
[5] John Willmer Escobar, and Rodrigo Linfati, and Paolo Toth. A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. Computers & Operations Research, 40(1):70–79, 2013.
[6] Seyed Sajjad Fazeli, and Ratna Babu Chinnam, and Alper Murat. Two-stage stochastic choice modeling approach for electric vehicle charging station network design in urban communities. IEEE Transactions on Intelligent Transportation Systems, 2020.
[7] Seyed Sajjad Fazeli, Saravanan Venkatachalam, and Jonathon M Smereka. Efficient algorithms for autonomous electric vehicles’ min-max routing problem. arXiv preprint arXiv:2008.03333, 2020.
Pierre H Hansen, B Hegedahl, S Hjortkjaer, and Børge Obel. A heuristic solution to the warehouse location-routing problem. *European Journal of Operational Research*, 76(1):111–127, 1994.

Bassem Jarboui, Houda Derbel, Said Hanafi, and Nenad Mladenovic. Variable neighborhood search for location routing. *Computers & Operations Research*, 40(1):47–57, 2013.

Rui Borges Lopes, Carlos Ferreira, Beatriz Sousa Santos, and Sérgio Barreto. A taxonomical analysis, current methods and objectives on location-routing problems. *International Transactions in Operational Research*, 20(6):795–822, 2013.

Yannis Marinakis and Magdalene Marinaki. A particle swarm optimization algorithm with path relinking for the location routing problem. *Journal of Mathematical Modelling and Algorithms*, 7 (1):59–78, 2008.

Yannis Marinakis, Magdalene Marinaki, and Nikolaos Matsatsinis. Honey bees mating optimization for the location routing problem. In *Engineering Management Conference, 2008. IEEE Europe 2008. IEEE International*, pages 1–5. IEEE, 2008.

Nenad Mladenovic and Pierre Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997.

Gábor Nagy and Said Salhi. Location-routing: Issues, models and methods. *European Journal of Operational Research*, 177(2):649–672, 2007.

Jossef Perl and Mark S Daskin. A warehouse location-routing problem. *Transportation Research Part B: Methodological*, 19(5):381–396, 1985.

Jossef Perl and Mark S Daskin. A warehouse location-routing problem. *Transportation Research Part B: Methodological*, 19(5):381–396, 1985.

Christian Prins, Caroline Prodhon, and Roberto Wolfler Calvo. A memetic algorithm with population management (ma|pm) for the capacitated location-routing problem. In *Evolutionary computation in combinatorial optimization*, pages 183–194. Springer, 2006.

Christian Prins, Caroline Prodhon, and Roberto Wolfler Calvo. Solving the capacitated location-routing problem by a grasp complemented by a learning process and a path relinking. *4OR*, 4(3):221–238, 2006.

Christian Prins, Caroline Prodhon, Ángel Ruiz, Patrick Soriano, and Roberto Wolfler Calvo. Solving the capacitated location-routing problem by a cooperative Lagrangean relaxation-granular tabu search heuristic. *Transportation Science*, 41(4):470–483, 2007.

Caroline Prodhon and Christian Prins. A survey of recent research on location-routing problems. *European Journal of Operational Research*, 2014.

Ching-Jung Ting and Chia-Ho Chen. A multiple ant colony optimization algorithm for the capacitated location routing problem. *International Journal of Production Economics*, 141(1):34–44, 2013.

Dilek Tuzun and Laura I Burke. A two-phase tabu search approach to the location routing problem. *European Journal of Operational Research*, 116(1):87–99, 1999.

Tai-Hsi Wu, Chinyao Low, and Jiunn-Wei Bai. Heuristic solutions to multi-depot location-routing problems. *Computers & Operations Research*, 29(10):1393–1415, 2002.

Vincent F Yu, Shih-Wei Lin, Wenyih Lee, and Ching-Jung Ting. A simulated annealing heuristic for the capacitated location routing problem. *Computers & Industrial Engineering*, 58(2):288–299, 2010.