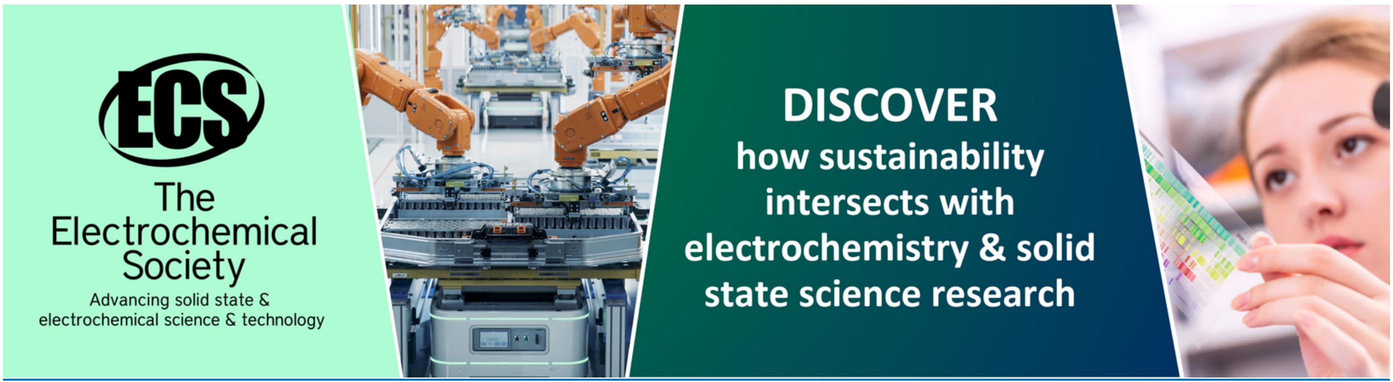
Journal of Physics: Conference Series

PAPER • **OPEN ACCESS**

VND in CVRP, MDVRP, and VRPTW cases

To cite this article: D Satyananda and S Wahyuningsih 2019 *J. Phys.: Conf. Ser.* **1320** 012025


ECS
The Electrochemical Society
Advancing solid state & electrochemical science & technology
DISCOVER
how sustainability intersects with electrochemistry & solid state science research

VND in CVRP, MDVRP, and VRPTW cases

D Satyananda[1] and S Wahyuningsih[2]

[1,2] Mathematics Department, Universitas Negeri Malang, Jalan Semarang No. 5 Malang, Indonesia

[1]darmawan.satyananda.fmipa@um.ac.id

Abstract. Vehicle Routing Problem (VRP) has an important role in logistics distribution from the depot to the customer, to get the minimum cost delivery route. To get optimal results, it is necessary to improve route from the initial solution. Variable Neighbourhood Descent (VND) is one of the metaheuristics that examine of a number of neighbourhood operators to get the optimal route. A VRP route is called optimal if there are no other routes that can be generated from all the neighbourhood operators used in VND. This article describes the application of VND to get the optimal route on CVRP, MDVRP, and VRPTW. The results of the experiment on some test data used indicate that VND can be used to get more optimal length and travel time route.

1. Introduction

Vehicle Routing Problem (VRP) is one graph theory implementation which has important role in logistic and distribution management worldwide. Objective of VRP is to design least cost delivery routes through a set of geographically scattered customers, subject to a number of side constraints [1].

The most basic thing in VRP is the existence of a depot (or more), a number of customers, distance between depots/customers, a number of vehicles with a certain capacity, and requests from each customer. The existence of different problem characteristics raises various VRP variants. Some of them are Capacitated VRP (CVRP) that uses a number of vehicles with the equal capacity, VRP with Time Window (VRPTW) that has time window for each customer, Mixed Fleet VRP (MFVRP) that use vehicles which have different capacities, Multi Depot VRP (MDVRP) that has more than 1 depot and customers are served by certain depot, and other more variants [2].

Two main types of algorithms are well-known for complex problem such as VRP: exact and heuristic. The exact method may solve a “big” problem; however, it could take enormous run time to complete. Hence, a heuristic algorithm is introduced that can immediately provide an approximate solution which is sometimes considered as the optimal solution. Researchers focused on approximate algorithm that provides near-optimal solution in quick time [3].

Heuristics are typically used when there is no known method to find an optimal solution, under the given constraints (of time, space etc.) or at all. In computer science, there are two fundamental goals: finding algorithms with proven good run times, as well as proven good or optimal solution quality. However, heuristic usually finds pretty good solutions, but there is no proof the solutions could not get arbitrarily bad; or it usually runs reasonably quickly, but there is no argument that this will always be the case [4].

Local optimum or local minima is the problem usually found in heuristics. In local optimum, a solution never reaches a feasible state or global optimum value. In this condition, the algorithm is "trapped" in certain circumstances where the operation only generates values not far from the local optimum area. Optimum at one locale is not necessarily optimum in another locale. The global optimum is a proven optimum solution for all locals. To overcome this problem, metaheuristic is introduced; metaheuristic is a framework to build heuristics for combinatorial issues and global optimization [5].

One framework to generate solutions using metaheuristic is Variable Neighbourhood Search (VNS) and Variable Neighbourhood Descent (VND) as a sub of VNS. VNS and VND up to now succeed to be applied in many cases such as Traveling Salesman Problem [6], Vehicle Routing Problem [3] [4], Redundancy Allocation Problem [7], Flowshop Scheduling Problem [8], Order Batching Problem [9], and in minimum labelling Steiner tree problem [10]. VNS/VND belongs to a local search metaheuristic group, applying iteratively to various neighbourhood structures, by modifying an initial solution [11]. The use of various neighbourhood structures in the local search will result in a better solution than the original solution.

This article discusses the implementation VND and its operators in some favors of VRP: CVRP, MDVRP, and VRPTW. The paper is organized as follows. In Introduction Section, background of the article is presented. Literature section comprehend the review related to VRP, VND and neighbourhood operators. Next section is about methodology, and finally test results for some instances and some computational results are discussed in Results and Discussion Section

2. Literature Review

2.1. *Variable Neighbourhood Descent*

VNS was developed by Mladenović since 1995. VNS systematically changes the neighbourhood, either in the gradual form leaving local optimum or "jumping" from the condition of local optimum (in the form of a valley) by using perturbation. Neighbourhood is a set of solution of a particular operation. Figure 1 shows the local optima condition and perturbation (or "shaking", in other literature) for jumping to other expected circumstances. Usually the perturbation uses a randomly-selected operator of a predefined operator.

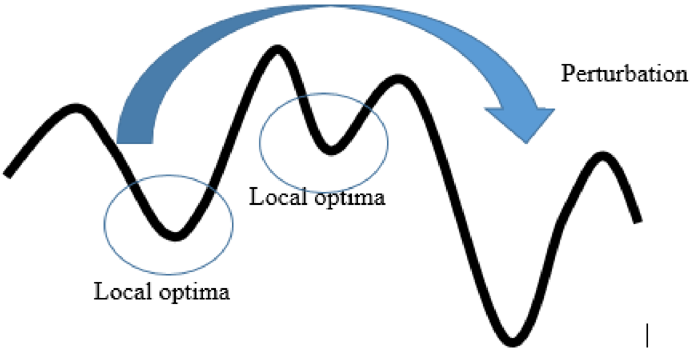


Figure 1. Local optima condition and perturbation to jump to other local optima

VND was proposed by Hansen and Mladenović in 2003 [3]. In VND, the neighbourhood change is in deterministic way, by eliminating the random shaking or perturbation stage. VND explores the solution space using various neighbourhood structures to ensure that there are optimal solutions with respect to all neighbourhoods that can be considered as global solutions. VND (even VNS) is based on principles: a local optimum in a certain neighbourhood structure does not correspond to local optimum in other neighbourhood structure (however they are close each other in many problems,), and global optimum corresponds to all local optimum in all neighbourhood structures [12].

The algorithm of VND can be seen in figure 2 [5].

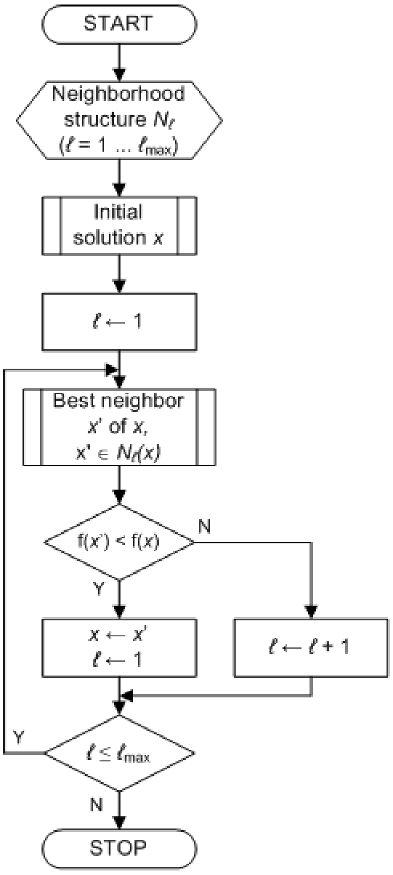

START
Neighborhood structure N_ℓ (ℓ = 1 ... ℓ_{max})
Initial solution x
$\ell \leftarrow 1$
Best neighbor x' of x, $x' \in N_\ell(x)$
$f(x') < f(x)$
N
Y
$x \leftarrow x'$
$\ell \leftarrow 1$
$\ell \leftarrow \ell + 1$
Y
$\ell \leq \ell_{max}$
N
STOP

Figure 2. Stages of VND

At the exploration of neighbourhood stage, a local search for a neighbourhood that gives the best x' is performed. At the next stage (“move or not” stage) a decision is taken. Neighbourhood index ℓ is set to the first neighbourhood and x’ become a new x when solution of x' is better than x, or to the next neighbourhood when x' equal or even worse than x. This is to ensure that x' will be optimum for all neighbourhoods, starting from the first neighbourhood. The iteration stops after $\ell > \ell_{max}$ which states that no more neighbourhoods are better that current best neighbourhood.

The choice of structure (operator to find a “best” solution in certain neighbourhood) can be done sequentially, nested, or a combination of the two [6] [12]. Selecting neighbourhood in random order gives better results than deterministic determinations [13], and does not require a neighbourhood setting [14].

Some things noticed in choosing neighbourhood are: (1) Number of neighbourhood operators used, (2) type of neighbourhood operators used (intra or inter neighbourhood), (3) order of the neighbourhood operator, and (4) whether the use of the neighbourhood ensure the most optimal results. Mistakes in selection can result in enormous computing time; remain trapped in a certain neighbourhood, and less optimal results.

VND in many cases is combined with other method. [3] used Iterated Local Search (ILS) VND in VRPTW, [8] used ILS VND in flowshop scheduling problem, [15] used ILS VND in VRP with Multiple Trip, [12] used Intensified ILS Method with Random VND in VRPTW, and [16] used Iterated VND for CVRP problems. Those articles proved that the combination could give better result.

2.2. Neighbourhood operators

Classification of neighbourhood operators are in two classes: intra-neighbourhood and inter-neighbourhood. Intra-neighbourhood operations are performed on a single route with the goal of minimizing the overall distance, and inter-neighbourhood operations are performed on two or more different routes with the goal of minimizing the overall distance and minimizing the number of vehicles as well [4].

Examples of inter-neighbourhood are [14]:

- Shift(1,0): One customer *i* is moved from route r_1 to r_2. This operation is also known as 1-0 Exchange Neighbourhood.
- Swap(1,1): Interchange between customer *i* from route r_1 and customer *j* from route r_2. This operation is known as 1-1 Exchange Neighbourhood
- Shift (2,0): Two adjacent customers *i* and *j* (or an arc (*i*, *j*)) are moved from route r_1 to route r_2. The transfer of the opposite arc(*j*, *i*) is also considered. Another name of this operation is 2-0 Relocate Neighbourhood
- Swap(2,1): Interchange between adjacent customers *i* and *j* from route r_1 and customer *k* from route r_2. The opposite arc (*j*, *i*) is also considered as well. This operation is also known as 1-2 Exchange Neighbourhood
- Swap(2,2): Interchange between adjacent customers *i* and *j* from route r_1 and two other adjacent customers *k* and *l*, from route r_1. The opposite arcs (*j*, *i*) and (*l*, *k*) are also considered, yielding 4 possible combinations. 2-2 Exchange Neighbourhood is another name of this operation.
- Cross Exchange: exchange of two arcs (two strings of consecutive nodes) from two routes, preserving the order of customers in each string [6], in swapping two paths belonging to two different routes [4].

Examples of intra-neighbourhood are [13]:

- Or-opt: one to three consecutive nodes are moved into other position in the route
- Two-opt: two nonadjacent edges are removed, and two new edges are formed then to make a new route
- Exchange: exchanging two nodes position. This is another version of swap(1,1), but it applied to the same route
- Reinsertion: a node is removed, and inserted into other position in the route

2.3. CVRP

CVRP is standard version of VRP, which only consider vehicle capacity as the only constraint. CVRP is modeled with connected graph $G = (V, E)$ whereas $V = \{ v_0, v_1, \ldots, v_n \}$ is set of vertices, and $\mathrm{E} = \{ (v_i, v_j) \mid v_i, v_j \in \mathrm{V}, v_i \neq v_j \}$ is a set of edges. Vertex 0 usually considered as a depot (distribution center or warehouse) and vertices 1..*n* as customers. The depot has a number of vehicles with homogenous capacity *Q*. Each edge (v_i, v_j) that connects vertex *i* to *j* has a weight that states the distance between them. Every customer has q_i demand. CVRP determines the route to deliver goods from the depot to all customers and back to the depot, each customer is only visited once by a vehicle, and the number of customer requests in a route must not exceed the vehicle capacity [16]. If total demand of the customers in certain route exceed the capacity of the vehicle, a new route must be established for the remaining customer. The objective of CVRP is to minimize total route length (total travel time as well) and number of vehicle (number of route as well since each route relates to certain vehicle).

2.4. MDVRP

MDVRP has many common characteristics with CVRP as a basic form of VRP. Vertice set *V* is divided into two subsets: $V_d = \{ V_0, V_1, ..., V_m \}$ as the set of depots and $V_c = \{ V_{m+1}, ... V_n \}$ as the customer set. Each depot serves certain subsets of customers [17]. Grouping customers into a specific depot (clustering) can use a number of ways, including based on the nearest Euclidean distance of the customer to the depot [18], k-means clustering [19], and probability [20]. Routing stages in each cluster can be completed in the same way as CVRP. The objective of MDVRP is basically the same as objective of CVRP.

2.5. VRPTW

In VRPTW, each customer *i* has time window [e_i, l_i] to serve the demand. A vehicle must arrive at the customer at e_i or afterthat, and must leave customer before l_i. If the vehicle come before e_i, then it has additional waiting time w_i, to wait for opening of the window [12]. The depot has time window [e_0, l_0]: Vehicles may not leave the depot before e_0 and must be back before or at time l_0. [21]. Due to time constraints, then each route has travel time (related to vehicle speed and distance between customers), and each customer has service time s_i (general service time for each customer, or associated with demand d_i). The objective of VRPTW is to minimize total route length, and number of vehicle (number of routes as well since each route relates to certain vehicle). Other objective is the minimization of total service time, total waiting time, and total travel time, which are less studied [11].

3. Research Method

This article discussed implementation of VND metaheuristic for CVRP, MDVRP, and VRPTW. The initial route was generated using Sequential Insertion algorithm. Clustering of vertices into a depot was using nearest Euclidean distance method. There were 10 neighbourhood operators used: (a) inter route operator: 1-0 Exchange neighbourhood (N1), 1-1 Exchange neighbourhood (N2), 1-2 Exchange neighbourhood (N3), 2-2 Exchange neighbourhood (N4), 2-0 Relocate neighbourhood (N5), Cross-Exchange neighbourhood (N6); and (b) intra route operator: Or-opt neighbourhood (N7), Two-opt neighbourhood (N8), Exchange (N9), Reinsertion (N10). Selection of operator was performed in random order.

The initial route establishment and repair process was implemented in a computer program created with Borland Delphi, executed in computer with Intel i-7 microprocessors, with 32 GB memory and Microsoft Windows 7 operating system. The program was tested to generate route of 100 customers. The data tested were dataset from some libraries: Breedam for CVRP: dataset 60G1 and 60G3, Cordeau for MDVRP: dataset P06 and P18, and Solomon for VRPTW: dataset C208 (clustered located customers) and dataset R211 (random distributed location). Every dataset was tested 10 times to get the average, best score, and worst score. There is no requirement on restrictions of route length for all datasets. Table 1 summarizes poperty of each dataset.

Table 1. Dataset specifications

Variant	Dataset	Customer	Depot	Vehicle	Capacity
CVRP	60G1	100		unlimited	100
	60G3	100		unlimited	200
MDVRP	P06	100	3	6	100
	P18	240	6	5	60
VRPTW	R211	100		25	1000
	C208	100		25	700

4. Result and Analysis

The computer program made was then executed using test data prepared. Table 2 shows the results of program execution. For each dataset, table 2 shows length of the initial route, number of route (and route for each depot as well in MDVRP cases), average, worst, and best length of the route of 10 observations, and reduction in route length. For VRPTW, overall travel time and reduction are also displayed.

Overall, there is a reduction in route length and travel time in all datasets tested. This shows that VND can provide better results than the initial route. The worst case produced is also still below the length of the initial route.

Table 2. Result of experiments

Variant	Dataset	Initial	Number of route	Average	Worst	Best	Reducti on (%)	Note
CVRP	60G1	1701,4	10	1566,458	1584,71	1545,85	7,93	distance
	60G3	1006,48	5	997,64	997,64	997,64	0,88	distance
MDVRP	P06	1409,94	Dep. 1: 6, Dep. 2: 4, Dep. 3: 6	941,215	967,57	922,81	33,24	distance
	P18	6616,41	Dep. 1: 4, Dep. 2: 4, Dep. 3: 4, Dep. 4: 4, Dep. 5: 4, Dep. 6: 4	4312,515	4424,88	4266,28	34,82	distance
VRPTW	R211	1809,02	4	861,945	936,71	808,25	52,35	distance
		2830		2754,484	2912,73	2600,46	2,67	time
	C208	986,35	4	619,707	655,89	601,77	37,17	distance
		10495,91		10397,31	12075,85	9667,68	0,94	time

Customers in both CVRP datasets have same structure (customer position, distance between customers, and their demand), and differ only in the capacity of the vehicle serving it (100 in 60G1 and 200 in 60G3 dataset). All customers have a demand of 10. Number of vehicles available in both datasets are not restricted, but the results showed that not all vehicles used. In the 60G1 dataset only 10 vehicles are needed (in other words, only 10 routes generated), and only 60 vehicles are needed in the 60G3 dataset. At 60G2 there was only a 0.88% reduction in length of route compared to the initial route, and there is no variation in the route produced between tests. This is seen in the similarity of results between the best case and the worst case. The 60G1 dataset can produce greater reductions because it is still possible to exchange between customers on one route with other routes.

Testing for MDVRP cases can produce a significant reduction of about 33% of the length of the initial route. This is opposite of the case for the previous CVRP. Large reductions indicate there are more variations in route settings for each refinement. Number of vehicles needed is not as much as the vehicles provided.

For VRPTW, there is a relatively large reduction in the route length generated in both test data (52% and 37%, respectively). This big difference is because the capacity of the available vehicles is very large, thus allowing more opportunities to improve (refine) the route. Small reductions also occur for travel times on each route. Number of vehicles used is also very small compared to available vehicles (4 of 25 vehicles).

Sequential Insertion algorithm for initial route generation tends to minimize the number of vehicles needed, but maximizes the number of customers on each route. The route improvement process after that tends not to change the number of routes generated. This is thought to be due to limited vehicle capacity, and there is not much remaining capacity on each route that allows merging routes.

At the time of the repair process (exploration of neighbourhood stage), basically carried out the exchange of one or more customers in one route with customers in other route (inter neighbourhood), or exchange between customers in the same route (intra neighbourhood). The exchange may result in reduced number of routes because the customer is moved to another route (for example in Swap(1.0) operator), but this is not happened in the case currently being tested. The exchange process must still pay attention to the constraints that each variant has. In the case of CVRP, the constraint is only capacity of the vehicle, hence every exchange will be checked whether it exceeds the existing capacity. The chosen route is the one which does not exceed the capacity of the vehicle and has a shorter route length. The same thing happened to MDVRP because basically MDVRP is a multiple CVRP. Customer exchange is only done at the same depot because customers are clustered prior to initial route generation.

VRPTW has more complex considerations. In addition to vehicle capacity as the basic consideration, the exchange must also consider the time window of each customer and depot. The arrival time of the vehicle must not exceed the customer closing time (l_i) and better to arrive at the customer opening time (e_i) or afterthat. For each time of insertion, arrival and departure time of a vehicle must be recalculated, starting from the insertion position (in CVRP and MDVRP just add a demand for the new customer added to the route, or reduce the demand from the customer omitted on one route). The route being considered is a route whose overall travel time does not exceed the closing time of the depot (l_0). This consideration led to very large iterations compared to CVRP and MDVRP. From observations, the average iteration of the two VRPTW test data is 9 million iterations (equivalent to 9 million times of the route improvement effort), 1.2 million iterations in both CVRP cases, and 95,000 iterations in both MDVRP cases. The slight number of iterations in MDVRP is also relevant to reduction of significant route lengths, as mentioned earlier. This shows high success of repairing routes, and also the faster the optimal route for all neighbourhood operators.

The high iteration is also directly proportional to time needed to complete the entire process (VRPTW takes about 10 minutes to complete all iterations with the computer specifications mentioned earlier). To be used as a commercial product, there needs restrictions as the product is acceptable by user, for example by reducing the number of operators used, limiting the number of customers, modifying the initial settings (e.g. increasing the number of vehicles or increasing vehicle capacity), or by applying parallel programming.

5. Conclusion

From the results of the experiment it was proved that VND can be used as an alternative to get the optimal route on VRP with a significant reduction in some test cases. The improvements must still pay attention to the constraints applied to each VRP variant. The weakness of the large test data is the number of iterations performed for route repair efforts, in the case of routes with many points. This certainly needs to be considered further when applied as a web-based application because there is a time limit for executing per request on the server.

Acknowledgements

This article is part of research entitled “Utilization of Google Map Services for the Application of VRP Problem Solving” (year 2018, second year of two years planned). Special thanks addressed to Directorate of Research and Community Services (DRPM), Indonesian Ministry of Research, Technology, and Higher Education (Kemenristekdikti) as funding this research.

References

[1] Laporte G, Toth P, and Vigo D 2013 Vehicle routing: historical perspective and recent contributions. *EURO J Transp Logist* **2** 1-4.

[2] Wahyuningsih S, Satyananda D, and Hasanah D 2016 Implementations of TSP-VRP variants for distribution problem. *Glob J Pure Appl Math* **12** 723–732.

[3] Harzi M, and Krichen S 2017 Variable neighborhood descent for solving the vehicle routing problem with time windows. *Electron Notes Discret Math* **58** 175–182.

[4] Pop PC, Fuksz L, and Marc AH 2014 A Variable Neighborhood Search Approach for Solving the Generalized Vehicle Routing Problem *Hybrid Artif Intell Syst* 13–24.

[5] Hansen P, and Mladenović N 2014 Variable neighborhood search. In: Burke EK, Kendall G (eds) *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. New York: Springer Science+Business Media pp. 313–337.

[6] Mjirda A, Todosijević R, Hanafi S, et al 2017 Sequential variable neighborhood descent variants: an empirical study on the traveling salesman problem. *Int Trans Oper Res* **24** 615–633.

[7] Liang Y C, Lo M H, and Chen Y C 2007 Variable neighbourhood search for redundancy allocation problems. In: *IMA Journal of Management Mathematics* pp. 135–155.

[8] Yahyaoui H, Krichen S, Derbel B, et al 2015 A hybrid ILS-VND based hyper-heuristic for permutation flowshop scheduling problem. In: *Procedia Computer Science* 632–641.

[9] Menéndez B, Pardo EG, Alonso-Ayuso A, et al 2017 Variable Neighborhood Search strategies for the Order Batching Problem. *Comput Oper Res* **78** 500–512.

[10] Consoli S, Darby-Dowman K, Mladenović N, et al 2009 Variable neighbourhood search for the minimum labelling Steiner tree problem. *Ann Oper Res* **172** 71–96.

[11] Chen B, Qu R, Bai R, et al 2017 An investigation on compound neighborhoods for VRPTW. In: *Communications in Computer and Information Science* pp. 3–19.

[12] Ribas S, Subramanian A, Machado I, et al 2011 A hybrid algorithm for the Vehicle Routing Problem with Time Windows *Proc Int Conf Ind Eng Syst Manag* 1243-52.

[13] Subramanian A, Drummond LMA, Bentes C, et al 2010 A parallel heuristic for the Vehicle Routing Problem with Simultaneous Pickup and Delivery. *Comput Oper Res* **37** 1899–1911.

[14] Silva MM, Subramanian A, Ochi LS 2015 An iterated local search heuristic for the split delivery vehicle routing problem. *Comput Oper Res* **53** 234–249.

[15] Belver M, Gomez A, Lopez J, et al 2017 Solving Vehicle Routing Problem with Multiple Trips using Iterative Local Search with Variable Neighborhood Search. In: *Int'l Conf. Artificial Intelligence | ICAI'17* pp. 286–290.

[16] Chen P, Huang H kuan, Dong XY 2010 Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem *Expert Syst Appl* **37** 1620–1627.

[17] Montoya-Torres JR, López Franco J, Nieto Isaza S, et al 2015 A literature review on the vehicle routing problem with multiple depots. *Comput Ind Eng* **79** 115–129.

[18] Geetha S, Vanathi PT, Poonthalir G 2012 Metaheuristic approach for the multi-depot vehicle routing problem. *Appl Artif Intell* **26** 878–901.

[19] Luo J, Li X, Chen M-R 2013 Multi-Phase Meta-Heuristic for Multi-Depots Vehicle Routing Problem. *J Softw Eng Appl* **06** 82–86.

[20] Kuo Y, Wang CC 2012 A variable neighborhood search for the multi-depot vehicle routing problem with loading cost. *lklklf* **39** 6949–6954.

[21] El-Sherbeny NA 2010 Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. *J King Saud Univ - Sci* **22** 123–131.