MATHEMATICAL MODELING AND COMPUTING, Vol. 8, No. 4, pp. 716–725 (2021)

A modified adaptive large neighbourhood search for a vehicle routing problem with flexible time windows

Labdiad F., Nasri M., Hafidi I., Khalfi H.

LIPIM, ENSA Khouribga, University Sultan Moulay Slimane, Bd Béni Amir, B.P. 77, Khouribga, Morocco

(Received 23 May 2021; Accepted 7 June 2021)

Vehicle routing problems are widely available in real world application. In this paper, we tackle the resolution of a specific variant of the problem called in the literature vehicle routing problem with flexible time windows (VRPFlexTW), when the solution has to obey several other constraints, such as the consideration of travel, service, and waiting time together with time-window restrictions. There are proposed two modified versions of the Multi-objective Adaptive Large Neighbourhood Search (MOALNS). The MOALNS approach and its different components are described. Also it is listed a computational comparison between the MOALNS versions and the Ant colony optimiser (ACO) on a few instances of the VRPFlexTW.

Keywords: *vehicle routing problem, flexible time window, operation research, numerical simulation, adaptive large neighbourhood search, meta-heuristic algorithm.*

2010 MSC: 90C11, 90C29, 65K05 **DOI:** 10.23939/mmc2021.04.716

1. Introduction

The Vehicle Routing Problem (VRP) is described as the problem of finding an optimal collection of routes from one or several depots to a predetermined number of scattered locations subject to side constraints enforcing some given importance criteria relative to cost, time, distance or a combination of these quantities. The basic version of the VRP problem is an extension of the Travelling Salesman problem [1]. It was originally introduced by [2] under the name of "Truck Dispatching Problem" and since then carried out the object of many intensive studies in its modeling and resolution aspects. VRPs nowadays plays a central role in many fields and in some real world application among which we cite the physical distribution and logistics, supply chain management, finance and so on. There exists a wide variety of VRPs and a broad literature on this class of problems see for example [3–6]. At its basic form, a VRP can be viewed as: a fleet of vehicles located at a central depot, that must ensure tours between several customers who have requested a certain merchandise or service. The set of customers visited by a vehicle refers to its tour and each tour starts and ends at the central depot. Each customer must be served once and only once and by one and only one vehicle. The objective of the standard VRP model is to minimize the sum of the distances travelled or the total travel time of vehicle rounds while meeting customer demand.

The model can be represented as a closed graph $G = (V, A)$ [7], where vertexes are clients $V = 0, 1, \ldots, n$ and 0 denotes the origin depot, arcs are the routes i, j linking two clients. There are m binary variables x_{ijp} used to check if a trajectory (ij) is actually travelled by the vehicle p or not. A second binary variable y_{ip} is to enforce the condition that each client will be served by only one vehicle, hence y_{ip} is equal to 1 when the vehicle p visited the node i and 0 otherwise. The mathematical model could be formulated as follows:

$$Z = \min F, \tag{1a}$$

$$\text{s.c.} \quad \sum_{p=1}^{m} \sum_{i=1}^{n-1} x_{0ip} \leqslant m, \tag{1b}$$

$$\sum_{p=1}^{m}\sum_{i=1}^{n-1} x_{i0p} \leqslant m, \tag{1c}$$

$$\sum_{p=1}^{m} y_{kp} \leqslant 1,\ \forall k = 1,\dots,n, \tag{1d}$$

$$\sum_{j=1}^{n-1} x_{ijp} = y_{ip},\ i = \{1,\dots,n\},\ p = \{1,\dots,m\}, \tag{1e}$$

$$\sum_{j=1}^{n} x_{jip} = y_{ip},\ i = \{1,\dots,n\},\ p = \{1,\dots,m\}, \tag{1f}$$

$$x_{ijp}, y_{ip} \in \{0,1\},\ \forall i,j = \{1,\dots,n\},\ p = \{1,\dots,m\}. \tag{1g}$$

The constraints (1b) and (1c) ensure that the number of vehicles leaving the depot is the same as the number of vehicles entering the depot. The constraint (1d) ensures that each city from 1 to n is visited by a maximum of one vehicle. The constraints (1e) and (1f) represent the conservation of flow for each city i and ensure that the number of vehicles crossing all the arcs entering $\{(j,i), \forall j \in A\}$ is equal to the number of vehicles crossing the outgoing arcs $\{(i,j), \forall j \in A\}$. Finally, the binary variables used x_{ijp} and y_{ip} are declared by the constraint (1g).

The vehicle routing problem is a classic extension of the travelling salesman problem. Both are part of the class of NP-complete problems. It is part of the optimization problems for which we do not know an algorithm allowing to find an exact solution quickly (polynomial time) in all cases.

As the VRP appears in real life, it may have several classes of additional constraints, such as limits on the vehicle capacity [8,9], time windows for serving customers [3–5], route lengths, or the number of hours worked by a driver or a distribution clerk. For a recent complete review on the classification of different VRP variants, see [6,10]. As with basic VRP, most VRP variants are known to be NP-hard. In this paper, we are interested in the VRP with flexible time window (VRPFlexTW). It is a relaxation of the VRPTW where time windows are considered hard constraints that should not be violated.

The layout of the paper is as follows: the next section aim to provide a brief introduction to the multi-objective VRPFlexTW problem modeling and a comprehensive overview of the popular resolution techniques used in the literature. Section 2 will present two modified versions of the ALNS Algorithms and theirs components with application to the considered VRPFLexTW problem. In section 3, numerical results will be presented and a comparison with other standard techniques such as ant colony optimization and the standard ALNS is carried out. Eventually, a summary of the results and discussion will be provided to wrap up this study.

2. Problem formulation of the VRPFlexTW

Let us consider a multi-objective VRPFlexTW formulation that seeks to optimize customer satisfaction when vehicle routes are constrained by capacity and time windows, while minimizing costs associated with the distance travelled and the number of vehicles. When one wants to extend the previous model (1) to the flexible version of the VRPFlexTW, it is necessary to make some modifications to the mathematical formulation. First of all, remember that a time window is in fact a time interval in which it is allowed to serve a customer at no additional cost. In our case, this interval is flexible, that is, with a certain penalty, it is possible to perform the service to customers outside this interval. It therefore becomes possible to increase the time windows for meeting clients from $[a_i, b_i]$, $\forall i \in N$ to $[a_i - a'_i, b_i + b'_i]$, $i \in N$. The constants a'_i and b'_i satisfy $a_i - a'_i \geqslant E_i$ and $b_i + b'_i \leqslant L_i$, where E_i and L_i are respectively tolerances for serving clients earlier or later than the appointed time interval. Although the waiting time is permitted at no cost, a client's satisfaction denoted by $\mu_i(z_i)$ will be constant on the interval $[a_i, b_i]$ but will decrease to 0 linearly when the time moves away from the agreed upon interval limits. The satisfaction function μ_i is taken as follows:

$$\mu_i(z_i) = \begin{cases} 0, & z_i < E_i, \\ \dfrac{z_i - E_i}{a_i - E_i}, & E_i \leqslant z_i < a_i, \\ 1, & a_i \leqslant z_i \leqslant b_i, \\ \dfrac{L_i - z_i}{L_i - b_i}, & b_i < z_i \leqslant L_i, \\ 0, & z_i > L_i. \end{cases} \tag{2}$$

Before presenting the mathematical formulation, we define the following notations:

- h_k is the transportation cost per unit distance of vehicle k,
- f_k is the fixed cost incurred for using vehicle k,
- c_{ij} is the distance between vertex i and vertex j,
- s_i is the service time at vertex i,
- w_i is the waiting time at vertex i,
- t_{ij} is the time required for travelling from vertex i to vertex j,
- Decision variables:

$$x_{ijk} = \begin{cases} 1, & \text{if vehicle } k \text{ travels from vertex } i \text{ to vertex } j, \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

$$y_{ik} = \begin{cases} 1, & \text{if vertex } i \text{ is served by vehicle } k, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Given the above parameters and decision variables, the problem can be formulated as follows:

$$\max \frac{1}{n} \sum_{i=1}^{n} \mu_i(z_i), \tag{5}$$

$$\min \sum_{k=1}^{m} h_k \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ijk} + \sum_{k=1}^{m} f_k \sum_{j=1}^{n} x_{0jk}, \tag{6}$$

$$\sum_{i=0}^{n} x_{ijk} = y_{jk}, \quad \forall k \in \{1, 2, \dots, m\}, \quad \forall j \in \{1, 2, \dots, n\}, \tag{7}$$

$$\sum_{j=0}^{n} x_{ijk} = y_{ik}, \quad \forall k \in \{1, 2, \dots, m\}, \quad \forall i \in \{1, 2, \dots, n\}, \tag{8}$$

$$\sum_{i=0}^{n} \sum_{j=0}^{n} x_{ijk} \left(t_{ij} + s_i + w_i\right) \leqslant r_k, \quad \forall k \in \{1, 2, \dots, m\}, \tag{9}$$

$$w_0 = s_0 = 0, \tag{10}$$

$$\sum_{k=1}^{m} \sum_{i=0}^{n} x_{ijk} \left(z_i + w_i + s_i + t_{ij}\right) = z_j, \quad \forall j \in \{1, 2, \dots, n\}, \tag{11}$$

$$E_i \leqslant z_i + w_i \leqslant L_i, \quad \forall i \in \{1, 2, \dots, n\}, \tag{12}$$

$$w_i = \max\{0, E_i - z_i\}, \quad \forall i \in \{1, 2, \cdots, n\}, \tag{13}$$

$$x_{ijk} \in \{0, 1\}, \quad \forall i, j \in \{1, 2, \dots, n\}, \quad \forall k \in \{1, 2, \dots, m\}, \tag{14}$$

$$y_{ik} \in \{0, 1\}, \quad \forall i \in \{1, 2, \dots, n\}, \quad \forall k \in \{1, 2, \dots, m\}, \tag{15}$$

$$z_i \geqslant 0, \quad \forall i \in \{1, 2, \dots, n\}. \tag{16}$$

In the model above, Objective (2) is to maximize the customer satisfaction. Objective (3) is to minimize the total routing costs, which consist of travel costs and fixed vehicle costs. Constraint (4) guarantees that the vehicle capacity is not exceeded; Constraint (5) ensures that each customer is served by exactly one vehicle; and Constraint (6) ensures that each route starts and ends at the depot. Constraints (7), (8) guarantee that each customer is served exactly once. Constraint (9) ensures that the maximum route time is not exceeded; Constraint (10) defines the waiting and service time at the depot; Constraint (11) represents the relationship between the arrival time at a vertex and the departure time from its predecessor; Constraint (12) ensures that customers are served within the required time; and Constraint (13) defines the waiting time.

3. Multi-objective ALNS Techniques for VRPFlexTW

The use of ALNS in multi-objective combinatorial optimization problems was pioneered by Schaus and Hartert [11] which emphasized the search process based on non-dominated solutions. The algorithm has been wildly used as an effective method to solve complicated neighbourhoods in tightly constrained problems, as searching small neighbourhoods may lead the algorithm to be struck in the local optima. In this class of algorithms, searching in a larger neighbourhood increases the chance of finding better solutions thanks to a variety of destroy and reconstruct methods that form an efficient adaptive search procedure balancing between intensification and diversification. The main process of multi-objective ALNS algorithm is depicted as follows:

Algorithm 1 Steps of the MOALNS algorithm.

initialize feasible solution x
set $x^* \leftarrow x$
insert x to feasible solution set
initialize adaptive weights
while the stopping criteria is not reached
    select a pair of destruction and reconstruction heuristics d_i, r_i based on the adaptive weights
    apply d_i and r_i to yield a new solution x'
    if x' can be accepted **then**
        add x' to the feasible solution set
        if x' is better than x^* **then**
            set $x^* \leftarrow x'$
        if x' is a non dominated solution **then**
            insert x' to Pareto set A
            update A
    randomly select x from A
    update the adaptive weights
return x^*

In this study, further improvement in MOALNS framework is considered to obtain the multi-objective optimal solution routes. The trade-off between objectives prevents a single unique best solution, instead it creates a set of solutions with optimal compromises of each objective. Thus, the proposed multi-objective Approach attempts to explore the neighbourhood spaces through the modification of non-dominated solutions.

We put forward two different alternatives to enhance the MOALNS process. The first approach is the modified adaptive large neighbourhood search (MALNS). It consists on a framework of meta-heuristic designed to solve the vehicle routing problem with flexible time windows. The main challenge is to highlight intensification over diversification within the heuristic search process. In this context, we incorporate the process of the choice function proposed by [12] into the ALNS algorithm. Therefore, numerous destroy/reconstruct methods are combined to explore multiple neighbourhoods within the same search which implicitly defines the large neighbourhood. The second alternative is a hierarchical approach composed of two stages as "Cluster first – Route second". In the first stage, customers are assigned to vehicles using K-medoids clustering algorithm within a spatio-temporal similarity distance. In the second stage, the VRPFlexTW is solved using two distinct routing algorithms (i.e., ALNS, GA and VNS).

3.1. The Modified MOALNS Algorithm

In this subsection, a detailed exposition of the improvements incorporated into the MOALNS algorithm for solving the VRPFlexTW is proposed. There is studied the integration of the modified choice function into the mechanism of MOALNS, in order to guide the research to areas where high-quality solutions are intended by seeking a trade-off between diversification and intensification. The selection criteria is improved instead of using a standard roulette wheel selection we use an advanced choice function taking into consideration the performance history of each applied heuristic pair of destruction and reconstruction operators.

3.1.1. The modified choice Function

The Modified Choice Function (MCF) is an efficient technique presented by [12] as an extension of the original choice function of [13]. The idea behind this method is to dynamically control the selection of heuristics on the basis of a combination of three different measures. Thereby, the heuristic to be selected must have the higher score F_t.

The first measure f_1 reflects the past performance of each single heuristic. This measure is represented by the equation:

$$f_1(h_j) = \sum_n \phi^{n-1} \frac{I_n(h_j)}{T_n(h_j)},$$

where $I_n(h_j)$ presents the change in fitness function, $T_n(h_j)$ is the time it takes the heuristic h_j to produce a solution for an invocation n, and ϕ is a parameter from the interval $[0,1]$ highlighting the recent performance.

The second measure f_2 tracks the dependency between a pair of heuristics (h_k, h_j), by considering their past performance when selected consecutively. The formula of this measure is given as follows:

$$f_2(h_j) = \sum_n \phi^{n-1} \frac{I_n(h_k, h_j)}{T_n(h_k, h_j)},$$

where $I_n(h_k, h_j)$ presents the change in fitness function, $T_n(h_k, h_j)$ is the time it takes to call the heuristic h_j immediately after h_k for an invocation n.

The third measure f_3 notes the elapsed time $(\tau(h_j))$ since an heuristic h_j was last called. This gives the heuristics which are inactive for certain time, an opportunity to be selected.

$$f_3(h_j) = \tau(h_j).$$

The formulation of the modified choice function is given as follows:

$$F_t(h_j) = \phi_t f_1(h_j) + \phi_t f_2(h_k, h_j) + \delta_t f_3(h_j),$$

where t denotes the number of invocations of heuristic h_j indicating an improvement by the used heuristic.

The measures f_1 and f_2 bring intensification to the search process while the measure f_3 supports diversification by giving a chance to inactive heuristics to be selected. This is possible by the incorporation of the parameters ϕ_t and δ_t. Where ϕ_t is an intensification parameter which weights f_1 and

f_2 respectively, and δ_t is the relative weight to f_3 and hence it is defined to control the diversification degree.

At each iteration, if the objective value improves, the value of ϕ_t is increased while δ_t is concurrently decreased. Conversely, ϕ_t is decreased and δ_t is increased when the objective value does not improve. The parameters ϕ_t and δ_t are expressed in the following way:

$$\phi_t(h_j) = \begin{cases} 0.99, & \text{if the objective value improves,} \\ \max\{\phi_{t-1} - 0.01, 0.01\}, & \text{otherwise} \end{cases}$$

and

$$\delta_t(h_j) = 1 - \phi_t(h_j).$$

3.2. A Cluster first – Route second approach for solving the VRPFlexTW

In this subsection, we propose an approach which fits into the class of Cluster first – Route second algorithms to deal with the VRPFlexTW problem. The strategy consists of two phases, clustering and routing. The first phase aims to define a set of cost-effective feasible clustering using k-medoid algorithm within an effective spatio-temporal distance similarity which is totally appropriate to the nature of the VRPFlexTW given that it considers both the spatial and temporal dimensions of the problem, while phase II is devoted to select the adequate routes, considering that each cluster corresponds to a specific VRPFlexTW subproblem. It is worth pointing that the choice of the K-Medoid was not arbitrary since it is more robust to noise and outliers and it is more flexible to be used with any similarity measure in contrast of other partitioning techniques which are not sensitive to noisy data or must be used only with distances that are consistent with the mean (e.g. K-Means).

3.2.1. Spatio-temporal distance

In practice, it is interesting to pay attention to the dynamic characteristics of the problem. Thus, assigning two customers which have a close spatial distance while their time windows of service are far is inefficient, since the related counterpart which is the waiting time will be increased and thereby missing opportunities to serve other customers. Therefore, the spatio-temporal measure seeks to explore the spatial and temporal similarities between customers in terms of both the travel distance and time windows aspects.

The generalized equation of the spatio-temporal distance is proposed as follows:

$$ST_{ij} = \alpha_1 d_{ij} + \alpha_2 T_{ij}, \tag{17}$$

$$\alpha_1 + \alpha_2 = 1, \quad \alpha_1 \geqslant 0, \quad \alpha_2 \geqslant 0, \tag{18}$$

where d_{ij} denotes the spatial distance between two customers i and j. The parameters α_1 and α_2 are weight coefficients which control how each distance, spatial d_{ij} and temporal T_ij, influence the spatio-temporal distance. It should be pointed here that before using the equation (6), both values of d_{ij} and T_{ij} must be normalized by their maximum or minimum values.

Consider $[A_{\text{start}}, A_{\text{end}}]$ and $[B_{\text{start}}, B_{\text{end}}]$ the time windows of customer A and B respectively, with $A_{\text{start}} < B_{\text{start}}$. Temporal distance between the two time windows T_{ij} used in the equation above is defined and addresses three different scenarios (see Fig. 1).

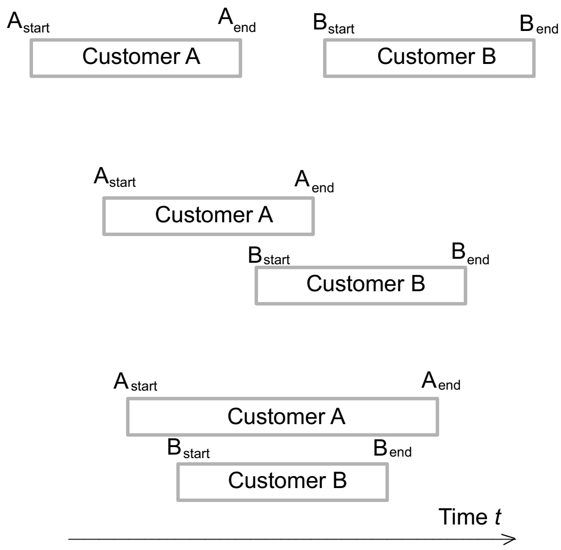

Fig. 1. Time windows overlap scenarios.

Three different situations can be considered according to the values of time windows. If $A_{\text{end}} < B_{\text{start}}$, there is therefore no overlap between the two time windows. If $A_{\text{end}} \geqslant B_{\text{start}}$ and $A_{\text{end}} < B_{\text{end}}$, there is a partial overlap between the two time windows. Finally, if $A_{\text{start}} \leqslant B_{\text{start}}$ and $A_{\text{end}} \geqslant B_{\text{end}}$, then a total overlap occurs.

Based on the presented cases, when two customers have overlapped time windows, they should be served in the same time. Then, the temporal distance between them is 0. Else, it can be determined as in Equation (8):

$$T_{ij} = \begin{cases} B_{\text{start}} - A_{\text{end}} & \text{if } A_{\text{end}} < B_{\text{start}}, \\ 0 & \text{if } A_{\text{end}} \geqslant B_{\text{start}}. \end{cases} \tag{19}$$

3.2.2. Description of our approach

Our methodology can be described on two steps. From one hand, the manner of clustering uses the K-medoid as a paradigm to tackle the pre-treatment process. On the other hand, the second step is devoted to select the adequate routes. This is the widespread ideas behind this approach. In the following, we provide more precise statements related to each step in more details:
Phase 1: It consists in identifying a set of clusters through a K-Medoid algorithm. The main idea of this iterative clustering algorithm is to divide the input data set into K distinct clusters $C_1, \ldots, C_K$.
Phase 2: It aims at finding a routing solution by solving each sub-problem related to each cluster. Then, we collect the solutions related to each sub-problem and gather them to obtain a complete solution when the sub-solutions will be the routes of the final solution. For this purpose, MOALNS is used to validate the results obtained in phase 1.

The K-Medoid algorithm used for Phase 1 is an iterative clustering algorithm which aims to divide the data set into K pre-defined distinct non overlapping clusters as such manner that the group similarity between cluster center points and data set point will be maximized, and the similarity distance between groups will be minimized. In this work, we measure the cluster similarity by the presented spatio-temporal distance between cluster medoids and data-set point. The general concept of this clustering algorithm can be outlined in the following steps:

- Select K of the N input data points as the initial medoids.
- Associate each data point to the closest medoid x by computing the spatio-temporal distance.
- Define the y point coincidence.
- If swapping x and y minimizes the cost function, swap x and y.
- Repeat the three previous steps until there is no change in the assignments.

In Phase 2 a greedy insertion heuristic introduced by (Solomon, 1987) in order to generate the initial solution for the process of the routing meta heuristics. This method consists of finding the best location of a given node by testing the different possible configurations. More explicitly, the algorithm selects the best feasible insertion place in the current route for each non inserted node considering two factors: the increase in total cost of the current route after the insertion, and the delay of service start time of the client following the new inserted client. This process ends when all deleted nodes will be inserted.

4. Numerical results

To investigate the performance of the ALNS in the context of the considered VRPFLexTW, we accomplished several computational tests. The algorithm was examined on a group of small instances in reference to the benchmark of Solomon, 1987, and its extension the instances of Gehring and Homberger, 1999. The Solomon set R containing randomized customers is used. We applied the algorithm on a number of Solomon's Sizes as a benchmark example. The algorithms were implemented in

Java 7, compiled with Intel compiler Celeron 1.80 GHz core i5 with 8GB RAM. The MALNS approach was run for 15600 iterations and was applied 10 times to each instance.

The results are showcased in the tables below. Table 1 showcases the optimal values of the Vehicle routing cost obtained through an the optimisation algorithms. Table 2 shows the optimal number of vehicles in the solution obtained for each client configuration. Lastly, Table 3 enumerates the execution time required to converge in each instance.

From the performance tables, we can see that the ALNS approaches are much more suitable to solve the considered problem in comparison to the Ant Colony Optimiser due to the nature of the population based mechanisms that seriously limit applicability to large problems and their memory and time-wise computational constraints. The standard MOALNS is a fairly efficient and fast algorithm thanks to its superior local search properties. Moreover, the quality of solutions obtained employs less vehicles to serve all target clients and take less time to compute. However, MOALNS struggles when the search spaces becomes too large as seen in the Tables when the fleet size is large, the local search becomes expansive. The modified ALNS using K-medoid is the fastest converging algorithm in this presentation, it produces smaller clusters sizes in comparison to the standard approaches but it is the less accurate and the procedure doesn't improve the optimality of the fitness function. In contrast, the Modified choice function coupled with MOALNS substantially improves the quality of solutions and reduces the optimal fleet size however it requires more computations because of its choosing mechanism that relies on a history of the performance of each optimization operator. The improvement in the quality comes at the expense of an increase in the run time of the algorithm.

Table 1. Value of the cost function for different numbers of clients.

Solomon size	ACO	ALNS	ALNS + K-medoid	ALNS + choice function
100-client	2635	1640	2021	1655
200-client	11074	4846	5887	4818
400-client	31702	12370	14078	12507
600-client	71154	26785	30368	27039
800-client	133482	51281	57444	51730
1000-client	219890	85904	95296	86761

Table 2. Optimal number of vehicles corresponding to each configuration of clients.

Solomon size	ACO	ALNS	ALNS + K-medoid	ALNS + choice function
100-client	27	10	14	10
200-client	65	12	16	12
400-client	141	24	30	25
600-client	224	40	49	41
800-client	307	65	76	67
1000-client	398	93	103	92

Table 3. Execution time of the ALNS Algorithm in seconds corresponding to each client configuration.

solomon size	ACO	ALNS	ALNS + K-medoid	ALNS + fct choice
100-client	2.47	0.96	0.54	0.59
200-client	7.9	3.86	1.11	1.24
400-client	34.2	34.41	3.78	3.51
600-client	85.53	168.01	8.43	10
800-client	187.36	489.19	21.7	29.55
1000-client	263.75	1086.41	32.41	47.74

5. Discussion and conclusion

Our main goal in this paper was to provide a comparative analysis between the proposed modified ALNS approaches using K-medoid and the choice function, in relations to the standard ALNS and the Ant Colony Optimizer for the VRPFLexTW problem. The state of the art concerning the VRPFlexTW is laid out and the versions of the modified Adaptive Large Neighbourhood Search are described and showcased. A comparison between these methods in terms of fleet size, cost optimization and time execution shows the superiority of the modified ALNS approaches in the flexible version of the VRPTW due to its interesting mechanism of construction and deconstruction operator that are capable of attaining quality solutions in shorter execution times and with less computational overhead.

[1] Dhaenens C., Espinouse M. L., Penz B. Problèmes combinatoires classiques In Recherche opérationnelle et réseaux: méthodes d'analyse spatiale. Hermès Science Publications (2002).

[2] Dantzig G., Fulkerson R., Johnson S. Solution of a large-scale travelling salesman problem. Journal of the Operations Research Society of America. **2** (4), 393–410 (1954).

[3] Baldacci R., Mingozzi A., Roberti R. Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. European Journal of Operational Research. **218** (1), 1–6 (2012).

[4] Balseiro S. R., Loiseau I., Ramonet J. An ant colony algorithm hybridized with insertion heuristics for the timedependent vehicle routing problem with time windows. Computers & Operations Research. **38** (6), 954–966 (2011).

[5] Baños R., Ortega J., Gil C., Fernández A., De Toro F. A simulated annealing-based parallel multi-objective approach to vehicle routing problems with time windows. Expert Systems with Applications. **40** (5), 1696–1707 (2013).

[6] Braekers K., Ramaekers K., Nieuwenhuyse I. V. The vehicle routing problem: State of the art classification and review. Computers & Industrial Engineering. **99**, 300–313 (2016).

[7] El-Sherbeny N. A. Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. Journal of King Saud University – Science. **22** (3), 123–131 (2010).

[8] Teymourian E., Kayvanfar V., Komaki Gh. M., Zandieh M. Enhanced intelligent water drops and cuckoo search algorithms for solving the capacitated vehicle routing problem. Information Sciences. **334-335**, 354–378 (2016).

[9] Vidal T. Technical note: Split algorithm in $O(n)$ for the capacitated vehicle routing problem. Computers & Operations Research. **69**, 40–47 (2016).

[10] Koc C., Bektas T., Jabali O., Laporte G. Thirty years of heterogeneous vehicle routing. European Journal of Operational Research. **249** (1), 1–21 (2016).

[11] Schaus P., Renaud H. Multi-objective large neighborhood search. International Conference on Principles and Practice of Constraint Programming. Springer, Berlin, Heidelberg (2013).

[12] Drake J. H., Ender O., Burke E. K. An improved choice function heuristic selection for cross domain heuristic search. International Conference on Parallel Problem Solving from Nature. Springer, Berlin, Heidelberg (2012).

[13] Cowling P. I., Kendall G., Soubeiga E. A hyperheuristic approach to scheduling a sales summit, in Selected papers from the Third International Conference on Practice and Theory of Automated Timetabling III', PATAT'00 (2001).

Модифікований адаптивний пошук великого околу для проблеми маршрутизації траспортних засобів з гнучкими часовими вікнами

Лабдіад Ф., Насрі М., Хафіді І., Халфі Х.

Національна школа прикладних наук Хурібги, Університет Султан Мулай Слімана, Bd Béni Amir, B.P. 77, Хурібга, Марокко

Задачі з маршрутизацією транспортних засобів широко доступні в сучасних застосунках. У цій статті розв'язано конкретний варіант цієї задачі, який в літературі називається задачею маршрутизації транспортних засобів з гнучкими тимчасовими вікнами (VRPFlextW), коли розв'язок має задовольняти декілька додаткових обмежень, таких як врахування подорожі, сервісу та часу очікування з обмеженнями часових вікон. Запропоновано дві модифіковані версії багатоцільового адаптивного пошуку великого околу (MOALNS), описано підходи MOALNS та його компоненти, проведено обчислювальне порівняння між версіями MOALNS та Optimiser Colony (ACO) для деяких випадків VRPFlexTW.

Ключові слова: *задача маршрутизації, гнучкі часові вікна, дослідження операцій, чисельне моделювання, адаптивний пошук великого околу, мета-евристичні алгоритми.*