BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 72(1), 2024, Article number: e148253
DOI: 10.24425/bpasts.2023.148253

Fixed fleet open vehicle routing problem: Mathematical model and a modified ant colony optimization

Majid YOUSEFIKHOSHBAKHT[1]*, Farzad DIDEHVAR[2], Farhad RAHMATI[2], and Zakir Hussain AHMED[3]

[1] Department of Mathematics, Faculty of Sciences, Bu-Ali Sina University, Hamedan, Iran
[2] Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran
[3] Department of Mathematics and Statistics, College of Science, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Kingdom of Saudi Arabia

Abstract. The fixed fleet heterogeneous open vehicle routing problem (HFFOVRP) is one of the most practical versions of the vehicle routing problem (VRP) defined because the use of rental vehicles reduces the cost of purchasing and routing for shipping companies nowadays. Also, applying a heterogeneous fleet is recommended due to the physical limitations of the streets and efforts to reduce the running costs of these companies. In this paper, a mixed-integer linear programming is proposed for HFFOVRP. Because this problem, like VRP, is related to NP-hard issues, it is not possible to use exact methods to solve real-world problems. Therefore, in this paper, a hybrid algorithm based on the ant colony algorithm called MACO is presented. This algorithm uses only global updating pheromones for a more efficient search of feasible space and considers a minimum value for pheromones on the edges. Also, pheromones of some best solutions obtained so far are updated, based on the quality of the solutions at each iteration, and three local search algorithms are used for the intensification mechanism. This method was tested on several standard instances, and the results were compared with other algorithms. The computational results show that the proposed algorithm performs better than these methods in cost and CPU time. Besides, not only has the algorithm been able to improve the quality of the best-known solutions in nine cases but also the high-quality solutions are obtained for other instances.

Keywords: open vehicle routing problem; heterogeneous; ant colony optimization; combinatorial optimization problems.

1. INTRODUCTION

One way to reduce the cost of goods is to minimize transportation prices so that the goods can be transferred from place to place with the least cost [1–6]. Therefore, nowadays, the importance of the vehicle routing problem (VRP) and its versions is not hidden from anyone, and its real applications in daily life have led researchers to pay more attention to it day by day. Figure 1 shows a feasible VRP solution that includes 46 customers and six vehicles. In this figure, vehicles have the same specific capacity, and each customer has a certain amount of demand. The horizontal and vertical axes represent the two-dimensional coordinate plane, and the total amount of customers' demands attributed does not exceed the vehicle capacity.

The OVRP issue happens in a lot of real problems, for example, assuming that a manufacturing company has signed a contract with a shipping company to distribute its goods, under which a fleet of vehicles is tasked every day to load certain goods from the company warehouse and deliver it to customers. Due to the leasing of the relevant fleet, these vehicles have only a duty to carry out the mission and no longer need to return

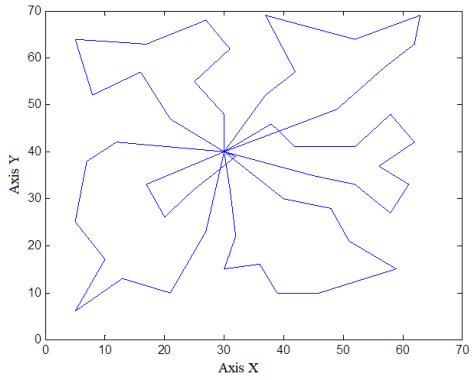

Fig. 1. A feasible solution for a VRP issue

to the warehouse. This issue aims to find routes for the fleet of vehicles at the lowest cost.

It should be noted that in these two instances, it is assumed that the capacity of all customers assigned to each vehicle is no more than its defined capacity. One of the earliest works on the OVRP problem, attempting to examine and classify this practical problem, was done by Schrage in 1981 [8]. He defines this issue as follows: A vehicle routing problem is identified based on three characteristics: capacity, cost, and being a VRP or OVRP. In a closed routing problem (VRP), vehicles are forced to return

*e-mail: khoshbakht@basu.ac.ir

Manuscript submitted 2023-05-22, revised 2023-10-01, initially accepted for publication 2023-11-01, published in February 2024.

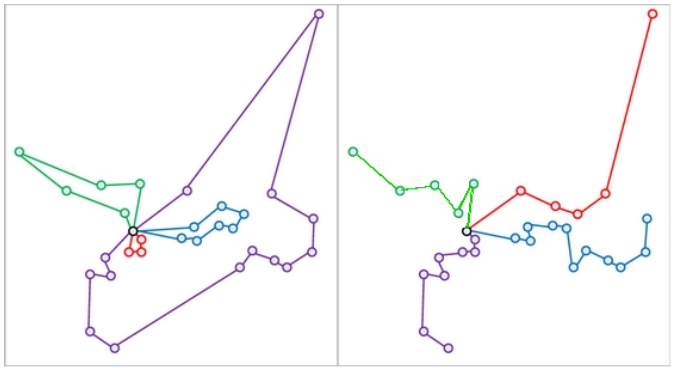

Fig. 2. The difference between VRP and OVRP [7]

to the warehouse after service. This constraint is not in the open version, and the vehicles will not return to the depot. Raff in 1983, adapted the OVRP problem in daily life and used it to solve the airmail distribution problem with several time window restrictions for receiving and delivering goods, the length of the entire route traveled by each vehicle, and the capacity of each vehicle [9]. They used a saving method to solve this problem, either if they were just a recipient or just a delivery provider. Tarantilis *et al.* proposed a population-based heuristic method [10], and in 2005, these authors examined the issue in more detail and suggested an acceptable starting point method for the problem [11]. In 2004 and 2006, the tabu search algorithm, which is one of the oldest methods of meta-heuristic and has many high-quality solutions to escape from local optimal points, was used by Brandao [12]. Pisinger and Ropke in 2007 and Lechford *et al.* in 2007 proposed a heuristic algorithm and a branch-and-cut algorithm, respectively, for the problem. These algorithms, compared to other methods presented in the literature on the relevant subject have better solutions [12].

Also, the proposed branch and bound algorithm could obtain better solutions for small and medium-sized instances and achieve optimal solutions. In the same year, Li and his colleagues implemented a hybrid algorithm for this issue. To test the efficiency of the algorithm, they generated some problems with 200 to 480 customers and reported the results of their proposed algorithm for these issues [13]. Also, a case of newspaper production and distribution into homes and work centers is considered and solved by the tabu search method. A variable neighbor search algorithm was presented in 2009 by Fleszar and his colleagues to solve the OVRP problem. This algorithm worked on edges and changing them and it was able to show efficiency and get some high-quality solutions in several standard instances [13]. Salari *et al.* presented an innovative improvement method for the OVRP problem. Their strategy was based on the linear integer programming method [14]. In 2011, MirHassani and Abolghasemi presented a particle swarm optimization method for this problem in which a new coding method was used for the issue [15]. A vector of customer position was created in a reduced order, and then each customer was selected according to the limitations of the situation in a path. Finally, the method used one-point motion to improve the solutions. A version of the OVRP problem is presented in [16] in which multiple warehouses are considered. It should be noted that in actual cases, this issue is used more, and an efficient algorithm including local search algorithms for this problem is provided. This algorithm produces very high-quality solutions using information updated every moment. The results of comparison with different algorithms show that this algorithm is very competitive compared to other algorithms.

Although OVRP is still being studied, the use of a heterogeneous fleet had not been considered in this issue until 2012, which is very practical. In other words, a homogeneous fleet cannot be used in all cases to further reduce transportation costs and traffic restrictions for vehicles. For this reason, Li considered a fixed heterogeneous fleet for OVRP (HFFOVRP) and proposed a modified meta-heuristic algorithm for this problem [17]. The proposed algorithm is based on the multi-start adaptive memory algorithm, which has obtained excellent answers compared to other algorithms. It should be noted that the reason for the efficiency of the algorithm, in addition to using appropriate solutions for intensification and diversification mechanisms, is the use of efficient adjustment for algorithm parameters.

A hybrid meta-heuristic algorithm was proposed by Yousefikhoshbakht *et al.* in 2014, using several local search algorithms to further enhance the solutions [18]. Also, by using efficient modifications to the proposed TS algorithm, a better feasible space was searched, and the algorithm became more efficient in the global search of the problem. In other words, the tabu period was used to improve the hybrid algorithm, and the minimum and maximum values for the tabu tenure are considered for the intensity and diversity policy. Besides, the same authors first proposed a mixed-linear programming model for this problem [19] and showed that due to the complexity of the problem, the exact algorithm could only solve problems up to 16 nodes optimally, but with an increasing number of nodes, it is better to use meta-heuristic algorithms. For this reason, a ranking ant system is presented in this paper [20]. The algorithm used diversification mechanisms for global search and an intensification mechanism for local search. In the article, a mixed integer linear programming model for this problem is presented. Then an exact and meta-heuristic combination method of column production is raised to solve this problem. In the proposed method, a method of elite ant system as diversification issue is presented to better search for possible space. Also, three local search algorithms of insert, swap, and 2-opt were used as three intensification methods. The results showed the efficiency of this method in comparison with the column generation and ant colony optimization (ACO).

Although the solution algorithms of this problem, like other optimization, are divided into two exact and heuristic categories [21–24], the difficulty of this problem has caused to lose efficiency of exact algorithms. In other words, when the size of the problem is increased, these algorithms need more time to find the optimal solution. Therefore, scientists use heuristic algorithms to solve these problems to obtain a near-optimal solution despite not getting the optimal solution to the issue at an acceptable time. Therefore, a modified version of ACO, called MACO, is proposed in this paper after presenting a new mixed-integer programming model. The proposed MACO uses some effective

modifications and local search algorithms to improve the diversification and intensification of the algorithm, including a new global update, an influential state transition rule, and a modified ranking method for ants. Comparing the results of MACO with some approximation and meta-heuristics algorithms shows that the algorithm can provide better solutions compared to these algorithms within a comparatively shorter CPU time.

In this paper, a mixed-integer linear programming model is presented in Section 2, and then Section 3 describes the proposed method. Section 4 offers at first sensitive analysis, and then the computational results performed on standard instances available in the literature. Section 5 shows the conclusions and future orientations.

2. PROBLEM DESCRIPTION AND FORMULATION

To present the HFFOVRP model, the $G(V,A)$ graph is used where $V=\{0,1,\ldots,n\}$ is the nodes set and $A=\{(i,j)\,|i,j\in V$ and $i\neq j\}$ is the set of edges. In this issue, each node i, except node 0 which shows the warehouse, represents the customers and has the demand for p_i goods. The distance c_{ij} corresponds to each arc in A, where $c_{ii}=0$ for each $i\in V$. A fleet of K different vehicle types is located at the depot so that each vehicle with type k has capacity Q_k, fixed cost f_k, and variable cost α_k. The number of k-th device shown by n_k is also available in the fleet. If α_k is the cost of each vehicle distance during one unit, the cost of navigating each arc (i,j) is equal to $c_{ij}^k=c_{ij}\times\alpha_k$ for the k-type vehicle. Therefore, there are K numbers of symmetrical cost matrices in the problem. If for $(i,j=0,1,2,\ldots,n\ ;i\neq j)$, k-th vehicle moves directly from nodes i to j, $x_{ij}^k=1$ and otherwise $x_{ij}^k=0$. Also, if for $i=0,1,2,\ldots,n$, k-th vehicle moves from customer i, $h_{ik}=1$ and otherwise $h_{ik}=0$. y_{ij}^k is the amount of goods that k-th vehicle carries when traveling from node i to j node.

The *HFFOVRP* problem involves determining a set of routes and allocating them to existing vehicles. The aim is to underestimate the total cost of the routes so that the following conditions are established:

- Meet the demands of all customers.
- Only use existing vehicles in customer service.
- Each customer's request is met once by a vehicle.
- No customer request $p_i$should be more than Q_k.
- No k-service is allowed along the way to upload more than the specified Q_k capacity.
- All vehicles must be at point 0 at the beginning and end their routes at arbitrary customers.

The mixed-integer linear programming model of HFFOVRP is as follows.

$$\text{Min}\sum_{k=1}^{K}f_k\sum_{j=1}^{n}x_{0j}^k+\sum_{k=1}^{K}\sum_{i=0}^{n}\sum_{j=0}^{n}c_{ij}^kx_{ij}^k$$

subject to

$$\sum_{i=0}^{n}p_ih_{ik}\leq Q_k\quad\forall k=1,2,\ldots,K, \tag{1}$$

$$\sum_{k=1}^{K}h_{ik}=1\qquad\forall i=1,2,\ldots,n, \tag{2}$$

$$\sum_{i=0}^{n}x_{ij}^k=h_{ik}\qquad\forall j=1,2,\ldots,n,\ \ \forall k=1,2,\ldots,K, \tag{3}$$

$$\sum_{j=1}^{n}x_{ij}^k=h_{ik}\qquad\forall i=0,1,\ldots,n,\ \forall k=1,2,\ldots,K, \tag{4}$$

$$\sum_{j=1}^{n}x_{0j}^k\leq n_k\qquad\forall k=1,2,\ldots,K, \tag{5}$$

$$\sum_{j=1}^{n}x_{i0}^k=0,\qquad\forall k=1,2,\ldots,K, \tag{6}$$

$$x_{ij}^k,h_{ik}\in\{0,1\}\quad\forall i,j=0,1,\ldots,n,\ i\neq j,\ \forall k=1,2,\ldots,K, \tag{7}$$

$$y_{ij}^k\geq 0\qquad\forall i,j=0,1,\ldots,n,\ \ \forall k=1,2,\ldots,K. \tag{8}$$

In this proposed model, the objective function is composed of two parts. In the first part, the total fixed cost for the use of vehicles is taken into account, while the second part considers the sum of all the edges met by the vehicles. Here, the objective of the problem is to minimize the sum of these two functions. Restrictions (1) prevent vehicle capacity from being breached, and vehicles can meet more vehicles as long as possible. Also, the limitations (2) make each node demand answered by a vehicle only once visited. Constraints (3) and (4) are very important in this model, which together ensure the continuity of the path for a vehicle. In other words, the vehicle that enters the node exits the same node. Also, the number used by vehicles should not be greater than the number available based on constraints (5). As a solution to this issue, there should be no way from any customer to the warehouse, which is guaranteed by (6). Binary variables must be used to calculate only the costs of the scrolling edges in the objective function. These limitations are considered by (7). If the edge is met, the amount 1 for this variable is devoted. Otherwise, the price is estimated to zero. Finally, restrictions (8) ensure that the flow of goods between nodes should never be negative.

3. THE PROPOSED METHOD

Nowadays, the VRP problems are significant, and these issues exist in daily life, and solving them solves socio-economic problems. These problems cannot be solved in practical sizes by exact optimization methods at an acceptable time. Because these problems have many local optimizations, innovative algorithms cannot get away from these local optimizations and achieve global optimizations. Therefore, researchers are looking for algorithms that have a suitable way to get away from local optimal points and reach a high-quality answer at an acceptable time. Metaheuristic methods can respond to this need and achieve a high-quality solution at a good CPU time. These algorithms, when trapped in local optimization, can escape them as much as possible, achieve other spatial solutions, and examine more areas. Therefore, these algorithms achieve better solutions by searching more and using more efficient procedures. In this sec-

tion, we present the proposed MACO for solving HFFOVRP with n customers.

In this method, a feasible solution to the problem is created as stated, which uses insert and exchange algorithms. The procedure is that the number of vehicles is considered first, and then the full 360 degrees of 2D space is divided into this number. Now, the algorithm starts from the maximum capacity of the vehicle and uses these two algorithms in which the number of demands is no greater than the ability of the vehicle. For each partition, the Hamilton path problem is created and solved by using the ACO described before. Therefore, the proposed method is a constructive algorithm, including random and probabilistic elements that cause the initial solution. Then, the initial value of the pheromone is attributed ten values to the edges of the feasible solution. Now, m different initial solutions, as a diversification strategy, are generated for n groups of ants. In this algorithm, each ant plays the vehicle role and builds a Hamiltonian route step by step. The probability of ant k is presented in formula (9) which moves from customer i to customer j. Besides, customer j has not been visited yet and its ability does not exceed the capacity of the corresponding ant.

$$P_{ij}^{k} = \begin{cases} 1 & \text{if } g \le q_0 \ \ \& \ \ i = j^*, \\ 0 & \text{if } g \le q_0 \ \ \& \ \ i \ne j^*, \\ \dfrac{\tau_{ir}^{\alpha}(t)\eta_{ir}^{\beta}(t)}{\sum\limits_{r \in J_i^k} \tau_{ir}^{\alpha}(t)\eta_{ir}^{\beta}(t)} & \text{otherwise,} \end{cases} \tag{9}$$

where:

$j^* = \arg\max_{r \in J_i^k} \tau_{ir}^{\alpha}(t)\eta_{ir}^{\beta}(t)$ identifies the unvisited node in J_i^k maximizing $P_{ij}^{k}(t)$.

$\tau_{ij}(t)$: The value of pheromone on arc (i, j).

$\eta_{ij}(t)$: The heuristic information arc (i, j) is defined as the reciprocal of the distance between node i and node j.

Parameters α and β determine the relative importance of pheromone level versus distance.

q : A uniformly distributed random number between 0 to 1.

q_0 : A variable assumed 0.2 at the beginning of the algorithm.

In every iteration of the algorithm, this variable is increased by 0.01 compared to the previous iteration until q_0 is equal to 0.9. The smaller the $0 \le q_0 \le 1$, the higher the probability of making a random choice.

α, β : The controlling parameters by the user.

To consider the solutions obtained in new iterations by the proposed ACO algorithm, which usually has better quality, the pheromone of the paths traveled by the ants should be updated at each iteration. In other words, in this algorithm, adaptive learning is done to make ants more inclined to higher-quality solutions. This is done by combining two operations to pour the pheromone in the new edges belonging to high-quality solution in the current repetition, and evaporation of the pheromone on all edges of the problem graph. This leads to new edges belonging to high-quality solutions, like all edges, losing some of their pheromones due to evaporation but in the same repetition, they receive a considerable amount of pheromone on their edges. The more time passes since the algorithm starts, the more pheromone difference will be created, which will attract more attention in subsequent repetitions of these solutions, and their neighbors.

For this purpose, the proposed algorithm only uses global updating, and unlike the classic ACO mode, local updating is not used. This strategy brings more attention to high-quality solutions, and two types of updating are used in the algorithm (formula (10)). The best answer in the current repetition is encouraged, and the pheromones on its edges are increased based on the amount of the quality solution obtained. On the other hand, the algorithm should consider perfect solutions and try to make a good balance between local and global search mechanisms. For this purpose, $\sigma - 1$ of the best solutions obtained so far are considered, and in addition to being updated in each iteration, the pheromones of their mane are updated. This will re-emphasize the excellent answers in each iteration mechanism and its neighbors will be searched more.

Indeed, to prevent premature convergence, an attempt to avoid other solutions soon is made. In this way, at least a minimum amount of pheromone is considered on the edges, equal to the *Map* value of the primary pheromone. In other words, whenever the amount of pheromone on the edge is less than Map the initial pheromone according to the evaporation, the value of the pheromone is replaced.

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{\mu=1}^{\sigma-1} \Delta\tau_{ij}^{\mu}(t) + \Delta\tau_{ij}^{gb}(t), \tag{10}$$

where: ρ – a parameter called evaporation rate in the range $[0, 1]$ regulating the reduction of pheromone on the edges.

$$\Delta\tau_{ij}^{\mu}(t) = \begin{cases} (\sigma - \mu)\dfrac{Q}{L^{\mu}(t)} & (i, j) \in S^{\mu}, \\ 0 & (i, j) \notin S^{k}, \end{cases} \tag{11}$$

Q : A constant coefficient determined by the user.

σ : The number of ranked solutions in which the pheromone has been deposited on their edges.

μ : The variable indicating ranking index from 1 to $\sigma - 1$.

S^{μ} : The traversed edges belonged to the μ-th rank.

$L^{\mu}(t)$: The length of the paths traversed by the μ-th ant.

The amount of the global pheromone release for the best solution is calculated by equation (12):

$$\Delta\tau_{ij}^{gb}(t) = \begin{cases} \sigma\dfrac{Q}{L^{gb}(t)} & (i, j) \in \text{best route}, \\ 0 & (i, j) \notin \text{best route}. \end{cases} \tag{12}$$

Although the ACO algorithm generally has good efficiency in global search, using local search algorithms is recommended to combine with this algorithm. Local search algorithms are commonly used in many articles to improve an initial solution and usually get high-quality solutions. Therefore, for a more efficient and diverse search in problem space, three local search algorithms, including insert move, exchange move, and 2-opt algorithm, are used in the proposed algorithm. In this algorithm, when the best solution is improved, it is better to use these algorithms to search more around the found solutions. Because

the implementation time of the algorithm is critical, in addition to quality, a percentage of the best solutions are considered to reduce CPU run time. Also, each of these local search algorithms is done with a probability of 1/3. It is logical to use this strategy at this time because better solutions in its neighbors are more likely to happen when a high-quality solution is obtained. It should be noted that the use of local search algorithms is only acceptable when the previous solution is replaced by an improved one.

In the insert algorithm, one node is selected and moves to a place in the current path or another path. Besides, in the exchange move, two nodes are selected from one or two paths and changed with each other. Finally, in the 2-opt local search, two edges are selected and replaced by the other two. In this algorithm, two edges can be found to be intersecting together, they are replaced with two other ones. These three local search algorithms are acceptable when all limitations are not violated, and the new solution has better quality than the previous one.

4. COMPUTATIONAL RESULTS

The proposed algorithm was coded in Matlab 2016 and all the experiments were implemented on a Laptop with core 3 2.6 GHZ, 4GB RAM, and Windows 7 Home Basic Operating system. In this section, the parameter tuning is presented, and the results of the MACO are shown.

4.1. Parameter tuning

Metaheuristic algorithms can obtain excellent solutions in a limited time because they use randomness to create solutions. Also, to develop this concept, an algorithm must use many parameters and there are different ways to find optimal solutions for them. Because this paper aims to find answers in an adequate time, an attempt is made to use a more uncomplicated strategy to adjust the parameters in this section. For this purpose, based on the results obtained in other articles, the essential parameters, including α, β, σ, and *Mao*, are considered in Table 1. In this table, the candidate values, and problem a5 are considered. Besides, this test problem is solved ten times for each parameter combination, and the best results are reported in the table below.

Table 1
Range of parameters of the proposed algorithm

Parameters	Range	The best value
alpha	0.5 1.5 2 2.5 3 3.5 4 4.5 5	1
beta	1 1.5 2 2.5 3 3.5 4 4.5 5	1
sigma	1 2 3 4 5 6 7 8 9	5
Mao	0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9	0.5

As mentioned above, the algorithm results with different parameters are displayed in Fig. 3 in more detail. In these forms, the amplitude of the parameters considered in the horizontal axis is shown, while the vertical axis represents the Gap between the best result obtained compared to the best-known solution (BKS). The Gap for the *best solution* obtained is computed by using formula (13):

$$Gap = \frac{\text{Best solution} - \text{BKS}}{\text{BKS}} \times 100. \tag{13}$$

It should be noted that since the quality of the algorithm is often lower than BKS, the values of these Gaps are positive numbers. On the other hand, a zero Gap shows that the algorithm has a high performance, and achieves BKS. By comparing the results in Table 1 for *alpha*, it can be concluded that based on Fig. 7, the Gap change diagram is almost ascending based on its values. In other words, the higher the amount of *alpha*, the lower the quality of the solution. Therefore, value 1 for this parameter causes the importance of pheromone information not to affect choosing a new path, especially at the beginning of the algorithm, which has not yet obtained reasonable solutions to the problem. Also, obtaining value 1 of beta has led the algorithm to seek more global searches in initial iterations and prevent early convergence at a slow rate to increase the pheromone on the edges. Also, for the best solutions obtained so far that they must have an updating pheromone at each iteration, value 5 for *sigma* has been obtained. Finally, the minimum pheromone of each edge for this problem is 0.5. In other words, if the pheromone of each edge is less than 0.5, the amount of the pheromone on the edge is increased to 0.5. It should be noted that high values cause the difference between the edges to increase slowly, and the algorithm does not have enough time to search locally.

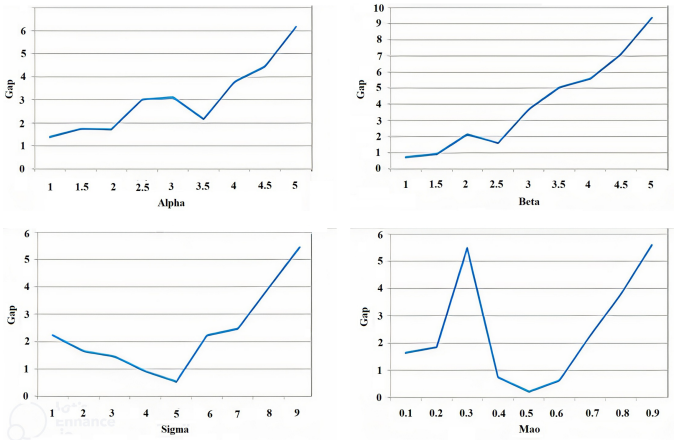

Fig. 3. Parameter tuning of the proposed algorithm

4.2. Comparison to other algorithms

The numerical experiment is performed using three sets of problem instances available in the literature. The first set was introduced in [25], and (0,0) is considered a depot for all cases. This data set was derived from the well-known Taillard's benchmark for the HFFVRP and consists of eight tests numbered from a1 to a8 with sizes ranging from 11 to 50 customers. The second data set is provided by Taillard [26] and consists of eight instances from a13 to a20 with sizes ranging from 50 to 100 customers. The third data set was introduced in [25], and its size is from 10 to 100 customers.

A simple criterion to measure the efficiency and the quality of an algorithm is to compute the Gap of its solutions from the exact solution on specific benchmark instances. Therefore, the Gap of solutions of ant colony optimization (ACO) and modified ant colony optimization (MACO) are compared to the IBM ILOG CPLEX Optimization Studio (often informally referred to simply as CPLEX) solver results on AIMMS software as an exact algorithm over the new eight small benchmark instances in Table 2. The default stop condition is that AIMMS will let the CPLEX solve the problem to local optimality. In this table, n is the number of customers for each example, and the results of three algorithms, including CPLEX solver, ACO, and MACO are shown. The last column presents BKSs for each instance in the literature. It is noted that each algorithm has three characteristics; the best solution (cost), CPU time (Time), and Gap. Since the ACO and MACO are meta-heuristic algorithms, the algorithms are run ten independent times, and the Gap and time of the best result are reported here.

Following Table 2, CPLEX can reach optimal solutions for only two small-scale instances but in other cases, this algorithm traps on local optimum points, and the lower bounds found are worse than their best solution results of other metaheuristic algorithms. The results indicate that ACO has been able to find the best solutions in two examples, including a1 and a2, of the eight examples. Furthermore, this algorithm has failed to improve the solutions in only one instance in the name of a4 compared to CPLEX and has come up with solutions similar to the ones found. Hence, it can be concluded that ACO is more efficient than CPLEX in finding high-quality solutions. On the other hand, the MACO has been able to find better solutions than CPLEX in five out of eight examples. In other words, the MACO algorithm has been capable of improving the solutions gained from CPLEX in instances a4, a5, a6, a7, and a8. Furthermore, the MACO can improve solutions obtained by the ACO in most instances, including a3, a4, a5, a6, a7, and a8. As a result, the proposed algorithm yields better solutions than ACO and CPLEX.

To assess the effectiveness of the MACO algorithm compared to ACO in large-size instances, the results of the proposed algorithm and ACO are shown in Table 3. This table contains eight standard HFFVRP problems which possess a fair number of customers whose sizes are between 50 and 100 called a13 to a20. For more information regarding the provided examples, visit:
http://mistic.heig-vd.ch/taillard/problemes.dir/vrp.dir/vrp.html

In Table 3, columns 3, 4, 5, and 6 show the results of ACO and MACO, and their CPU times. From the comparison between ACO and MACO, it can be seen that the ACO has been able to find near solutions with a gap of less than one percent in three examples compared with MACO. In more detail, ACO has produced an equaled solution only in a14, while MACO has generated better solutions in the remaining seven examples. It should be noted that ACO has had a weak performance in general, and has not produced the best solutions in any of the examples except for a14. Therefore, MACO can escape local optimum points and it makes a satisfactory improvement in the performance compared to the ACO algorithm.

Table 3
Comparing ACO and MACO for HFFOVRPS

Instance	N	ACO	Time of ACO	MACO	Time of MACO	BKS
a13	50	992.76	98.15	**990.11**	94.32	990.11
a14	50	**448.25**	101.42	**448.25**	104.11	448.25
a15	50	714.67	88.73	**709.31**	92.22	709.31
a16	50	802.95	99.52	**791.01**	106.31	791.01
a17	75	823.56	123.52	**815.05**	129.11	815.05
a18	75	1635.67	135.78	**1601.55**	145.04	1601.55
a19	100	938.62	172.77	**900.11**	189.78	900.11
a20	100	1088.62	178.02	**1038.58**	179.81	1038.58

To provide a better comparison between the classic ACO algorithm and the proposed algorithm, see Fig. 4. In this figure, several examples are shown in the horizontal axis, while the Gap

Table 2
Comparison results for small-size problems

	n	CPLEX			ACO			MACO			BKS
		Cost	Cap	Time	Cost	Gap	Time	Cost	Gap	Time	
a1	11	400.53	0	8	400.53	0	4.19	400.53	0	3.91	400.53
a2	16	416.80	0	5299	416.80	0	6.12	416.80	0	5.76	416.80
a3	21	723.90	0.54	14528	721.35	0.19	9.93	719.989	0	8.22	719.989
a4	26	734.99	0.26	18953	738.45	0.73	30.45	733.12	0	29.97	733.12
a5	31	827.51	0.14	11734	827.51	0.14	46.11	826.33	0	48.22	826.33
a6	36	530.18	2.13	131144	529.12	1.93	93.43	519.12	0	99.12	519.12
a7	41	–	–	267161	785.92	1.29	99.19	775.92	0	98.23	775.92
a8	50	–	–	645442	658.77	2.47	90.12	642.88	0	88.32	642.88

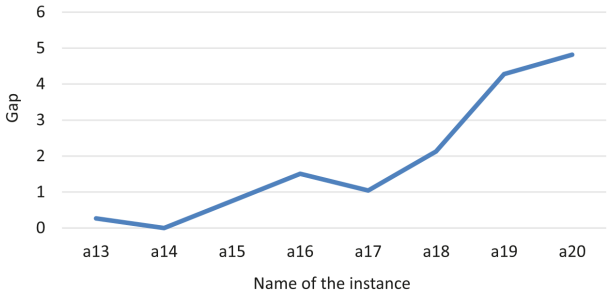

Fig. 4. The Gap of solutions by ACO compared to the MACO

obtained by the ACO algorithm compared to the best MACO solutions is presented in the vertical axis. Note that in this form, there is only one specific curve that is ascending. This shows that with the increasing number of customers, the difference between the two algorithms grows. Therefore, the proposed algorithm is more efficient than the classic type of algorithm in large-size problems. Also, to determine the efficiency of the two algorithms, and the obtained solutions during the implementation of the algorithm, Fig. 5 shows the value of the solutions of the two algorithms for instance a19. It should be noted that in this figure, the horizontal axis shows the repeat number of the algorithm, and the vertical axis shows the repeat number of the best solution obtained in that repetition. This is why the curves of both algorithms are descending, although the quality of the solutions has not grown in some repeats. In this figure, it can be seen that although the ACO algorithm has good performance in the tenth repetition of the implementation, and achieved 938.62, in subsequent iterations it has not been able to escape from this optimal local point and achieve better solutions. Therefore, it has suffered a premature convergence and lost many repetitions to improve. On the other hand, the MACO algorithm obtained a much better solution than ACO and has been able to achieve BKS at 16 repetitions with a good speed. The obtained result for this problem is 900.11, which is 4% better than the ACO solution.

From the statistical viewpoint in Table 4 there is no statistically significant difference between MACO and ACO (p-value = 0.908687). Furthermore, it can be observed from both the Mean and Standard Deviation values that the proposed algorithm is better than the ACO algorithm. In more detail, the Mean values are 302.82 for ACO and 292.81 for MACO. Also, the results indicate that these two algorithms have perfect ability for small-size problems, have approximately similar behavior, and can converge to the best solutions. However, the ability of ACO, against the proposed MACO, decreases when the number of nodes and solutions increases.

Table 4
ANOVA table

Source	SS	df	MS	F	Prob
Between-treatments	1186.953	1	1186.953	0.013	0.909
Within-treatments	2661572.123	30	88719.071		
Total	2662759.076	31			

To confirm the results obtained by MACO, two obtained solutions are shown in Fig. 6. It should be noted that in this example,

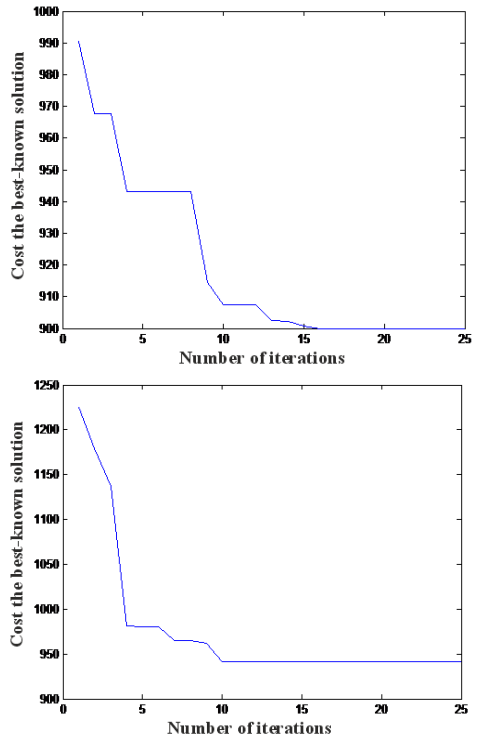

Fig. 5. Comparison of MACO (top) and ACO (bottom) in example a13

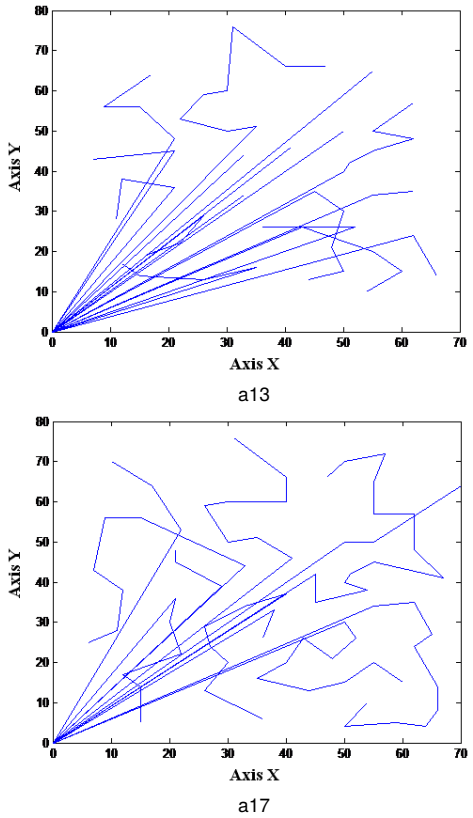

Fig. 6. Some of the HFFOVRP solutions found by MACO

the origin of the coordinates is considered as the depot, and the rest of the customers are scattered in the first quarter of the coordinates. In this figure, the horizontal and vertical axes show the coordinates of the customers and the warehouse, and the routes are obtained openly, indicating the correctness of the solutions obtained. The MACO algorithm in these two forms has better quality than ACO.

Although direct comparisons of the required computational times cannot be generally conducted for various algorithms in different computers, comparing the CPU time of the mentioned algorithms is possible here because they run in the same system. Figure 7a shows the CPU times of CPLEX compared to the proposed algorithm and the CPU comparison of the ACO and MACO algorithms are shown in Fig. 7b. Because the CPU time of CPLEX could not be reported for the large-size instances, only eight instances, including a1, a2, a3, a4, a5, a6, a7, and a8 are considered for the first comparison. For the second comparison, all instances including small- and large-size instances are considered for comparing MACO against ACO. By comparing the obtained results in these figures, it is concluded that the proposed MACO algorithm can obtain high-quality solutions in sufficient CPU time.

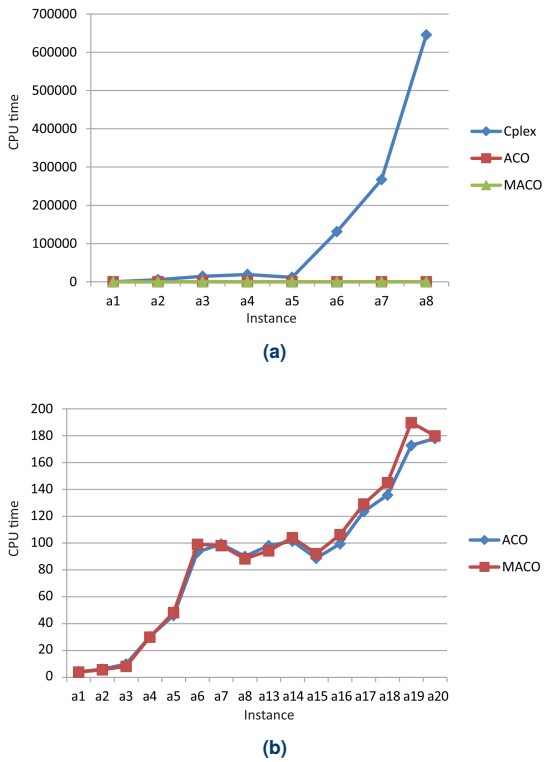

Fig. 7. Comparison of the CPU time between CPLEX, ACO, and MACO

The third category of instances, which includes a range between 10 and 100 customers, is derived from reference [25]. In this set, the customers are scattered around the depot randomly. Also, since the coordinates of customers and warehouses exist as two-dimensional points as data it is easy to calculate their Euclidean distances. In Table 4, the specifications of these customers along with the proposed algorithm results are presented, and compared to the results of the elite ant system (EAS), genetic algorithm (GA), column generation (CG), and column generation with modified ant system mixed with local search (SISEC). It should be noted that this table presents the number of customers of each example, the amount of CPU time of each algorithm for each example, the BKS results, and the Gap between the results of the proposed algorithm compared to BKSs. In other words, the solution to the CG and SISEC algorithms will be improved sequentially until the software itself stops it. On the other hand, EAS, GA, and MACO methods have user-controlled execution time and their results have been considered in ten repetitions, and the best ones were reported.

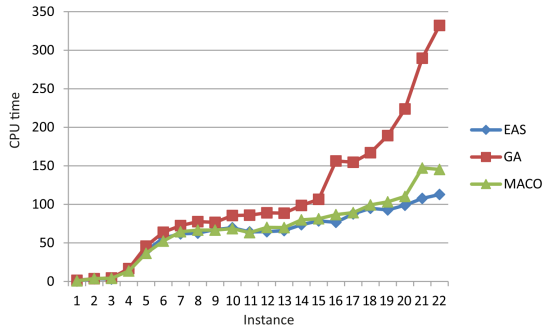

Fig. 8. Comparison of the CPU time between metaheuristics

According to these results, it can be seen that the proposed algorithm has perfect performance compared to other methods and has been able to improve BKS values in nine instances. Apart from two examples 6 and 12, algorithms have achieved BKSs in the rest of the instances. This algorithm has been able to significantly improve the quality of the solutions compared to the two EAS and GA meta-heuristic methods. In addition, these two algorithms have been able to compete closely with the proposed algorithm and achieve better solutions in less time compared to the SISEC algorithm which has the best solutions. Figure 8 presents the comparison of *CPU* time between meta-heuristic algorithms, including EAS, GA, and MACO for the 22 instances. In this figure, the horizontal axis shows the names of instances, and the vertical axis indicates the *CPU* time of each algorithm.

5. CONCLUSION AND FUTURE WORK

In this paper, the HFFOVRP problem was investigated, as a new version of the OVRP problem and a hybrid meta-heuristic algorithm was presented. The MACO algorithm had excellent efficiency in exploration and exploitation mechanisms and was able to get high-quality solutions compared to other presented methods for this problem (Table 5). It should be noted that several effective solutions were presented to better search the neighbors of high-quality solutions, and therefore it has an outstanding performance in finding high-quality answers. It can be predicted that using this algorithm for new versions of VRP will lead to finding high-quality solutions. Besides, other methods can be used as local search algorithms. On the other hand, paying

Table 5
Comparison results of MACO with other algorithms

Instance	n	EAS	T(EAS)	GA	T(GA)	CG	T(CG)	SISEC	T(SISEC)	MACO	T(MACO)	BKS	Gap
1	10	**191.10**	0.55	**191.10**	1.55	**191.10**	0.36	**191.10**	0.21	**191.10**	0.85	191.10	0
2	15	**282**	3.61	**282**	3.69	**282**	4.28	**282**	2.34	**282**	3.91	282	0
3	20	**379.63**	3.12	**379.63**	4.62	**379.63**	5.44	**379.63**	3.51	**379.63**	4.11	379.63	0
4	25	439.12	15.53	441.56	16.73	**437.79**	9.78	**437.79**	7.54	**437.79**	13.53	437.79	0
5	30	473.31	39.65	475.61	45.87	473.31	16.46	**471.81**	10.26	**471.81**	36.61	471.81	0
6	35	349.95	56.31	351.73	63.81	349.95	370.08	**345.81**	165.45	346.42	52.34	345.81	-0.18
7	40	607.99	61.68	611.67	72.39	600.99	767.42	**589.41**	457.94	587.64	64.62	589.41	0.30
8	45	676.04	62.64	687.46	77.67	676.04	2408.40	**671.91**	1157.81	**671.91**	66.72	671.91	0
9	50	915.63	67.27	924.73	76.72	907.3	256.72	**899.51**	98.64	895.75	66.73	899.51	0.42
10	50	537.18	69.76	537.18	85.62	507.58	2253.26	**451.21**	1784.54	443.31	68.59	451.21	1.75
11	50	829.24	64.31	829.24	86.11	826.19	840.18	**798.61**	621.39	790.86	63.29	798.61	0.97
12	50	952.41	64.51	963.61	89.12	947.81	698.96	**924.18**	584.74	926.11	70.11	924.18	-0.21
13	55	1086.65	65.98	1084.16	88.52	1074.91	1940.82	**998.81**	1254.93	**998.81**	69.90	998.81	0
14	60	1984.09	73.71	1984.09	98.62	1937.03	8807.25	**1842.81**	5694.87	1803.56	79.83	1842.81	2.13
15	65	1598.23	78.61	1599.18	106.67	1563.33	4203.14	**1498.28**	3549.14	**1498.28**	81.31	1498.28	0
16	70	974.27	76.73	983.52	156.27	962.57	6357.70	**925.48**	3614.87	**925.48**	86.82	925.48	0
17	75	1399.17	87.72	1421.60	154.72	1356.67	6021.97	**1326.36**	4987.31	1307.67	89.28	1326.36	1.41
18	80	1321.44	94.93	1387.29	167.13	1285.74	8331.64	**1248.21**	5927.98	**1248.21**	99.38	1248.21	0
19	85	1295.58	92.72	1341.52	189.32	1295.58	13731.35	**1254.33**	8642.29	1225.63	103.30	1254.33	2.29
20	90	1685.51	98.83	1783.29	223.69	1645.21	62748.94	**1512.84**	14621.62	**1512.84**	110.38	1512.84	0
21	95	1998.62	107.72	2210.27	289.73	1951.45	124548.98	**1789.34**	36124.87	1734.76	147.27	1789.34	3.05
22	100	2198.56	112.82	2287.39	332.27	2154.36	195648.84	**2054.11**	32548.18	2001.11	145.28	2054.11	2.58

attention to another version of this problem called HFFOVRP with time windows is very useful, which can be utilized in the delivery of perishable goods or the time limit of the recipients of the goods. The application of these ideas and the presentation of this new version will be postponed to future work.

ACKNOWLEDGEMENTS

The authors extend their appreciation to the Deputyship for Research and Innovation, Ministry of Education, in Saudi Arabia for funding this research through the project number IFP-IMSIU-2023104. The authors also appreciate the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University (IMSIU) for supporting and supervising this project.

REFERENCES

[1] Z.H. Ahme and M. Yousefikhoshbakht, "An improved tabu search algorithm for solving heterogeneous fixed fleet open vehicle routing problem with time windows," *Alex. Eng. J.*, vol. 64, pp. 349–363, 2023, doi: 10.1016/j.aej.2022.09.008.

[2] M. Yousefikhoshbakht, N. Malekzadeh, and M. Sedighpour, "Solving the traveling salesman problem based on the genetic reactive bone route algorithm whit ant colony system," *Int. J. Prod. Manag. Engineering*, vol. 4, no. 2, pp. 65–73, 2016.

[3] F. Maleki and M. Yousefikhoshbakht, "A hybrid algorithm for the open vehicle routing problem," International Journal of Optimization in Civil Engineering, vol. 9, no. 2, pp. 355–371, 2019.

[4] B. Guan, Y. Zhao, and Y. Li, "An improved ant colony optimization with an automatic updating mechanism for constraint satisfaction problems," *Expert Syst. Appl.*, vol. 164, p. 114021, Feb. 2021, doi: 10.1016/j.eswa.2020.114021.

[5] M. Yousefikhoshbakht and A. Dolatnejad, "A column generation for the heterogeneous fixed fleet open vehicle routing problem," *Int. J. Prod. Manag. Eng.*, vol. 5, no. 2, pp. 55–71, 2017.

[6] M. Yousefikhoshbakht, E. Mahmoodabadi, and M. Sedighpour, "A modified elite ACO based avoiding premature convergence for traveling salesmen problem," *J. Ind. Eng. Int.*, vol. 7, no. 15, pp. 68–75, 2011.

[7] T. Pichpibul, and R. Kawtummachai, "A heuristic approach based on clarke-wright algorithm for open vehicle routing problem," *Sci. World J.*, vol. 2013, p. 874349, 2013, doi: 10.1155/2013/874349.

[8] L. Schrage, "Formulation and structure of more complex/realistic routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 229–232, 1981, doi: 10.1002/net.3230110212.

[9] S.J. Raff, "Routing and scheduling of vehicles and crews," *Comput. Oper. Res.*, vol. 10, no. 2, pp. 63–211, Jan. 1983, doi: 10.1016/0305-0548(83)90030-8.

[10] C.D. Tarantilis, D. Diakoulaki, and C.T. Kiranoudis, "Combination of geographical information system and efficient routing algorithms for real life distribution operations," *Eur. J. Oper. Res.*, vol. 152, no. 2, pp. 437–453, Jan. 2004, doi: 10.1016/s0377-2217(03)00035-3.

[11] C.D. Tarantilis, G.N. Ioannou, C.T. Kiranoudis, and G.P. Prastacos, "Solving the open vehicle routeing problem via a single parameter metaheuristic algorithm," *J. Oper. Res. Soc.*, vol. 56, no. 5, pp. 588–596, May 2005, doi: 10.1057/palgrave.jors.2601848.

[12] A. Mor and M. G. Speranza, "Vehicle routing problems over time: a survey," *Ann. Oper. Res.*, vol. 314, no. 1, pp. 255-275, 2022.

[13] F. Li, B. Golden, and E. Wasil, "The open vehicle routing problem: Algorithms, large-scale test problems, and computational results," *Comput. Oper. Res.*, vol. 34, no. 10, pp. 2918–2930, Oct. 2007, doi: 10.1016/j.cor.2005.11.018.

[14] K. Corona-Gutiérrez, S. Nucamendi-Guillén, and E. Lalla-Ruiz, "Vehicle routing with cumulative objectives: A state of the art and analysis," *Comput. Ind. Eng.*, vol. 169, p. 108054, 2022.

[15] M. Salari, P. Toth, and A. Tramontani, "An ILP improvement procedure for the Open Vehicle Routing Problem," *Comput. Oper. Res*, vol. 37, no. 12, pp. 2106–2120, Dec. 2010, doi: 10.1016/j.cor.2010.02.010.

[16] S.A. MirHassani and N. Abolghasemi, "A particle swarm optimization algorithm for open vehicle routing problem," *Expert Syst. Appl.*, vol. 38, no. 9, pp. 11547–11551, Sep. 2011, doi: 10.1016/j.eswa.2011.03.032.

[17] J. Brandão, "A memory-based iterated local search algorithm for the multi-depot open vehicle routing problem," *Eur. J. Oper. Res.*, vol. 284, no. 2, pp. 559–571, Jul. 2020, doi: 10.1016/j.ejor.2020.01.008.

[18] X. Li, S.C.H. Leung, and P. Tian, "A multi start adaptive memory-based tabu search algorithm for the heterogeneous fixed fleet open vehicle routing problem," *Expert Syst. Appl.*, vol. 39, pp. 365–374, 2012.

[19] M. Yousefikhoshbakht, F. Didehvar, and F. Rahmati, "Solving the heterogeneous fixed fleet open vehicle routing problem by a combined metaheuristic algorithm," *Int. J. Prod. Res.*, vol. 52, no. 9, pp. 2565–2575, Nov. 2013, doi: 10.1080/00207543.2013.855337.

[20] M. Yousefikhoshbakht, F. Didehvar, and F. Rahmati, "A mixed integer programming formulation for the heterogeneous fixed fleet open vehicle routing problem," *J. Optim. Ind. Eng.*, vol. 8, no. 18, pp. 37–46, 2015.

[21] F. Nakhaei, M. Irannajad, and M. Yousefikhoshbakht, "Flotation column performance optimisation based on imperialist competitive algorithm," *Int. J. Min. Miner. Eng.*, vol. 7, no. 1, pp. 1–17, 2016.

[22] F. Nakhaei, M. Irannajad, and M. Yousefikhoshbakht, "Simultaneous optimization of flotation column performance using genetic evolutionary algorithm," *Physicochem. Probl. Miner. Proc.*, vol. 52, no. 2, pp. 874–893, Jun. 2016, doi: 10.5277/ppmp160228.

[23] A. Rahmani, and M. Yousefikhoshbakht, "Capacitated facility location problem in random fuzzy environment: using (α, β)-cost minimization model under the Hurwicz criterion", *J. Intell. Fuzzy Syst.*, vol. 25, no. 4, pp. 953–964, 2013.

[24] J. Kochańska, A. Burduk, D. Łapczyńska, and K. Musiał, "The solution of MRSLP with the use of heuristicalgorithms," *Bull. Pol. Acad. Sci. Tech. Sci.*, vol. 72, no. 1, p. e146407, doi: 10.24425/bpasts.2023.146407.

[25] M. Yousefikhoshbakht, A. Dolatnejad, F. Didehvar, and F. Rahmati, "A Modified Column Generation to Solve the Heterogeneous Fixed Fleet Open Vehicle Routing Problem," *J. Eng.*, vol. 2016, p. 5692792, 2016, doi: 10.1155/2016/5692792.

[26] É. Taillard, "Parallel iterative search methods for vehicle routing problems," *Networks*, vol. 23, no. 8, pp. 661–673, Dec. 1993, doi: 10.1002/net.3230230.