International Journal on Advanced Science Engineering Information Technology

Vol.12 (2022) No. 4
ISSN: 2088-5334

Multiobjective Memetic GRASP to Solve Vehicle Routing Problems with Time Windows Size

Carlos Molano [a], Manuel Lagos [a], Carlos Cobos [a,*]

[a] *Computer Science Department, Universidad del Cauca, Sector: Tulcan - Building: FIET - Office: 422, Popayán, 190001, Colombia*
Corresponding author: [*]ccobos@unicauca.edu.co

***Abstract*—The Vehicle Routing Problem with Time Windows is a complete NP combinatorial problem in which product deliveries to customers must be made under certain time constraints. This problem can be solved from a single objective approach, well studied in the state of the art, in which the objective of the total travel distance or the size of the fleet (number of vehicles) is generally minimized. However, recent studies have used a multiobjective approach (Multiobjective Vehicle Routing Problem with Time Windows, MOVRPTW) that solves the problem from a viewpoint closer to reality. This work presents a new multiobjective memetic algorithm based on the GRASP (Greedy Randomized Adaptive Search Procedures) algorithm called MOMGRASP for the minimization of three objectives in MOVRPTW (total travel time, waiting time of customers to be attended, and balance of total travel time between routes). The results of the experimentation carried out with 56 problems proposed by Solomon and 45 problems proposed by Castro-Gutiérrez show that the proposed algorithm finds better solutions in these three objectives and competitive solutions than those reported by Zhou (compared to LSMOVRPTW algorithm and optimizing 5 objectives: number of vehicles, total travel distance, travel time of the longest route, total waiting time due to early arrivals, and total delay time due to late arrivals) and by Baños (versus the MMOEASA algorithm in two scenarios; case 1: total travel distance and balance of distance and case 2: total travel distance and balance of workload).**

***Keywords*— Vehicle routing problem; grasp; metaheuristic algorithm; combinatorial optimization; multiobjective; memetic.**

Manuscript received 13 Aug. 2021; revised 29 Oct. 2021; accepted 10 Nov. 2021. Date of publication 31 Aug. 2022.

I. INTRODUCTION

The Vehicle Routing Problem (VRP) occurs in several different logistics environments. In the transport of goods and supplies, optimal routes are sought while meeting the needs of all customers and coping with a restriction on the number of vehicles available (fleet size). In a real setting, VRP has some or all of the following objectives: reduction of costs, reduction of the distances traveled by the vehicles, reduction of the time of travel and service of the vehicles, reduction in the size of the fleet, and a balance in the distances that vehicles are required to travel [1][2].

VRP can be classified into the following main categories: Capacitated Vehicle Routing Problem (CVRP), in which there is a fleet of vehicles with the same characteristics (capacity and speed), which, starting from a depot, must deliver some products to certain customers located in different parts of a region with a specific number of products per customer (product demand) and that as a whole (sum of demands for that delivery) should not exceed the capacity of the vehicle. The cost is determined solely by the distance traveled from the vehicles to the clients, and clients can be served in any time frame [3]. Heterogeneous Vehicle Routing Problem (HVRP) is a problem similar to CVRP but in which the fleet has vehicles of different characteristics. Therefore, variations in the travel costs of the vehicles, the speed of the vehicles, the maximum travel time, and the availability of these can occur; And Vehicle Routing Problem with Time Window (VRPTW), which is an extension of CVRP adding a time window in which customers must be served (service time), which is generally different for each one. The time window or time interval has a beginning and an end, which must be respected, with the purpose that it is in this interval where the goods (products) can be delivered to each specific client. There is also a service time that corresponds to the service time of a vehicle for a customer that must be met when the product is delivered. In VRPTW, the travel cost is the cost of the distance traveled. There can also be vehicles with heterogeneous characteristics, in which case HVRP would be combined with VRPTW [2].

VRPTW can be solved from a single objective optimization approach (much studied in the state of the art) where the minimization of travel distances and the number of

vehicles (fleet size) is generally sought, or from a multiobjective optimization approach (MOVRPTW). The latter has recently attracted attention because it allows defining objectives that are more applicable to real problems, such as, for example, in addition to the classic objectives: improve service quality (earlier delivery of products), minimize the variance in the distances traveled by vehicles, minimize variance in vehicle load, among many others.

Companies that within their business need to transport goods to their customers, mainly seek to retain them through the timely delivery of their products and reduce their distribution costs by reducing, for example, the total transport distance. Carrying out the process in an efficient way allows, in addition to the direct benefits to the company, making a contribution to the conservation of the environment by reducing the emission of carbon dioxide, which is highly polluting [4].

The algorithms that have been used to solve MOVRPTW, VRPTW, or similar can be organized into four main categories:

- Simple state metaheuristics such as Simulated Annealing and Tabu search, among others [5]–[11],
- Population-based metaheuristics such as Genetic Algorithms, Ant Colony Optimization algorithms, and Particle Swarm Optimization, among others [12]–[21],
- Hybrid algorithms [9], [22]–[26], and
- Exact algorithms [27], [28].

The different proposals use specialized operators that seek to solve specific problems such as, for example, reducing the longest route, reducing the time in the beginning to serve customers, balancing the load of the routes (homogeneity in the amount of product delivered by vehicles), minimizing the total distance traveled by all routes, among others. It should be noted that the intelligent initialization of the solutions is also necessary to obtain results that solve the real needs of customers and companies (good results are not always obtained with totally random initial solutions; in addition, the convergence time of the algorithms is increased and the time windows, for example, are not considered as a priority).

The main contribution of this work includes 1) a new way to convert the original version of GRASP (Greedy Randomized Adaptive Search Procedures) algorithm to a multi-objective version including in addition of specific knowledge of the application context to solve MOVRPTW problems. This way of converting GRASP can also be used for other single objective algorithms and in other optimization problems, and 2) a new multiobjective memetic algorithm based on the GRASP algorithm called MOMGRASP.

MOMGRASP has the following main components: i) an initialization of solutions based on two methods that always generate feasible solutions and that take advantage of prior knowledge of the state of the art to start with good quality solutions; ii) the flexibility of using different operators to create neighboring solutions according to the objectives to be optimized; iii) the use of a local optimizer that performs various enhancement operations on a solution to find the best possible neighbor within a maximum number of iterations, iv) the use of a single-state GRASP to evolve a solution avoiding being trapped in local optima; and v) the use of a population ordered by the non-dominance of the solutions and the Crowding distance that evolves over time. MOMGRASP was evaluated in four test scenarios against two algorithms reported in the state of the art with superior or competitive results in three of these scenarios.

The rest of the document is organized as follows, Section 2 presents the proposed algorithm, starting with the formal definition of MOVRPTW and the three objectives of interest in the proposal, then explains in detail the proposed algorithm starting with the Population envelope continuing with the simple state and ending with the operators. Then in Section 3, the results of the experiments carried out in four different scenarios (objective configurations) are presented. Finally, the conclusions of the research and the future work that the research group hopes to carry out in the short term are presented.

II. Materials And Method

The proposed algorithm called MOMGRASP searches for solutions for MOVRPTW with a population of fixed size that provides the user with the set of non-dominated solutions that make up the Pareto Front. In the evolutionary process, MOMGRASP ends its execution based on a maximum execution time defined by the user and uses Crowding distance to compare undominated solutions with each other, prioritizing the diversity of the solutions. MOMGRASP executes a single-state GRASP algorithm as an optimizer that uses operators that integrate specific knowledge of the problem (memetic approach).

The proposed algorithm allows to solve MOVRPTW problems with the objectives proposed by Baños [12] (total travel distance and balance of distance; total travel distance and load balance), Zhou [5] (number of vehicles, total travel distance, travel time of the longest route, total waiting time due to early arrivals, and total delay time due to late arrivals), the three objectives of interest of this work (total travel time, waiting time of customers to be attended, and balance of total travel time between routes) or combinations of these. To achieve this, specific operators are used to improve each of these objectives by integrating knowledge of the problem.

A. Formal Definition of MOVRPTW

The MOVRPTW problem can be described by 3 main matrices, namely: the adjacency matrix, the service time matrix, and the matrix with the solution of the routes.

Bearing in mind that it seeks to serve M clients with R vehicles, each of which performs its own route (R routes), then the adjacency matrix is represented as $A = (a_{i,j})$ of size M+1 * M+1, which stores the distances between the depot (position 0 of the matrix in rows and columns, that is, base index 0 of the matrix) and each customer and the distance of all customers from each other, in where $a_{i,j} = a_{j,i}$ y $a_{i,i} = 0$, which implies that the transpose of the matrix (A^T) is equal to the original matrix A.

The service time matrix $C = (c_{i,k})$ of size M x 3 (base index 1) that represents information from each customer c_i as a triple (ts_i, twi_i, twf_i) o $(c_{i,1}, c_{i,2}, c_{i,3})$ with the values of service time (time it takes to service any vehicle to customer i), start time of the service window for customer i and the end time of the service window for customer i. This matrix

establishes the restrictions of service for each of the clients and must be strictly observed to obtain a feasible solution.

The solution matrix $S = (s_{r,i})$ of R rows (base index 1) and variable size of columns that corresponds in each row to the value δ_r, where row r of S represents the route made by vehicle r starting and ending at the depot and going through all your customers in order, that is, a variable length array {customer A, customer H…, customer C}.

Given the above, the travel time of route r to customer l ($TR_{r,l}$) is defined as the travel time from the depot through customers of the route to customer l with the recursive expression of **Eq. (1)**.

$$TR_{r,l} = \begin{cases} c_{s_{r,l},2} + c_{s_{r,l},1} & si\ TR_{r,l-1} + a_{s_{r,l-1},s_{r,l}} \leq c_{s_{r,l},2} \\ TR_{r,l-1} + a_{s_{r,l-1},s_{r,l}} + c_{s_{r,l},1} & de\ otro\ modo \end{cases} \quad (1)$$

Subject to:

- $l \geq 1$ (the base index of S is one)
- $TR_{r,l-1} + a_{s_{r,l-1},s_{r,l}} \leq c_{s_{r,l},3}$ (the vehicle must arrive before the time window for the customer ends)
- $TR_{r,0} = a_{0,s_{r,l}}$ (Corresponds to the displacement of vehicle *r* of the last customer to the depot)

The total travel time of route r (TTR_r) is defined as the travel time from the depot, passing through all its clients and returning to the depot, plus the service time of each client in the route, plus the vehicle waiting time ($TE_{r,l}$) when it reaches each customer before the start of the service window, which corresponds to: $TTR_r = TR_{r,\delta_r} + a_{s_{r,\delta_r},0}$.

The route vehicle waiting time (r) for a customer (l) is defined as the time a vehicle must wait to serve a customer when it arrives before the service start window, which is formally expressed in the **Eq. (2)**.

$$TE_{r,l} = \begin{cases} 0 & si\ TR_{r,l-1} + a_{s_{r,l-1},s_{r,l}} > c_{s_{r,l},2} \\ c_{s_{r,l},2} - (TR_{r,l-1} + a_{s_{r,l-1},s_{r,l}}) & de\ otro\ modo \end{cases} \quad (2)$$

The version of MOVRPTW that is sought to be solved in this work, seeks the optimization (minimization) of three objectives (total travel time, waiting time of customers to be attended, and balance of total travel time between routes). That is, $\boldsymbol{Minimize\ f} = \{\boldsymbol{f_1}, \boldsymbol{f_2}, \boldsymbol{f_3}\}$, where $\boldsymbol{f_1}$ corresponds to the first objective to be minimized, which corresponds to the total travel time of all routes (vehicles) to their clients as expressed in **Eq. (3)**.

$$f_1 = \sum_{r=1}^{R} (TTR_r) \quad (3)$$

$\boldsymbol{f_2}$ corresponds to the second objective to be minimized, which corresponds to the waiting times of the clients when being served as expressed in **Eq. (4).**

$$f_2 = \sum_{r=1}^{R} (TTE_r) \quad (4)$$

$\boldsymbol{f_3}$ corresponds to the third objective, which searches for homogeneous routes, that is, it seeks to minimize the difference in travel times between the routes as expressed in **Eq. (5)**, in which TPS corresponds to the average service time of the routes.

$$f_3 = \frac{1}{R} \sum_{r=1}^{R} (TTR_r - TPS)^2 \text{ where } \boldsymbol{TPS} = \frac{1}{R} \sum_{r=1}^{R} TTR_r \quad (5)$$

B. MOMGRASP

Fig. 1 presents the pseudocode of the MOMGRASP algorithm. Its execution requires a problem pv (read from a file or dataset), the objectives to be minimized (o), the initial seed of random values (s) that ensures the repeatability of the experiments, the maximum execution time (tm) and other parameters that are used when the method Single State GRASP is called. Its operation is divided into two main parts.

```
MOMGRASP (problem pv, objectives o, seed s,
          maximum execution time tm, probability pl,
          restricted list size rls, maximum number of
          optimization iterations mnoi, maximum
          number of operations mno, operator's
          configuration oc, population size Psize,
          population of Nondominated solutions
          NDsize)
1.  do
2.      p = p U SingleStateGRASP (pv, o, s, tm, pl, rls,
            mnoi, oc, mno, null, null, null) if it is valid
3.  while (SizeOf (p) < Psize)
4.  ReduceNDSUsingCrowding (p, NDsize)
5.  do
6.      sol = SelectRandomSolution (p)
7.      amountOfRC = NRU (sol) * FCR
8.      rtp = RoutesToPreserve (sol, amountOfRC)
9.      cv = CustomersVisited (sol, rtp)
10.     p = p U SingleStateGRASP (pv, o, s, tm, pl, rls,
            mnoi, oc, mno, sol, rtp, cv) if it is not
            dominated
11. while (tm has not been reached)
12. ReduceNDSUsingCrowding (p, NDsize)
13. return Non-dominated-set
```

Fig. 1 MOMGRASP algorithm pseudo-code

The first part (lines 1 to 4) delegates to Single State GRASP (a method detailed below) the process of construction and improvement of solutions that make up the initial population (p) with size Psize (parameter defined by the user). Then (line 4) is reduced based on Crowding distance to a NDsize size (population of Nondominated solutions), a parameter that the user also defines. In the population, each solution has defined the routes. The values obtained for each of the objectives, among other data are used to calculate the ranking on the Pareto front and the Crowding distance of the solutions that share the same Ranking.

The second part (lines 5 to 11) corresponds to an iterative process of improvement of the solutions already built, which is repeated if the maximum execution time (tm) defined by the user is not reached. The cycle begins by randomly taking a non-dominated solution from the population. It then determines the number of routes that will be preserved intact (amountOfRC) of this solution in the improvement process that will be carried out using Single State GRASP. This number is obtained (line 7) from the integer division of the number of routes in use (NRU) in the solution and the factor of routes to conserve (FCR, user-defined parameter), which allows the use of prior knowledge of the solutions already found in the iterative improvement process. Then, in line 8, which routes are to be kept are defined and they are included

in the rtp list that is then used to make a list of the clients that have already been visited (cv) with these routes (line 9). With the selected solution, the routes to keep rtp from this solution and the clients already visited by these cv routes, Single State GRASP is executed seeking to obtain a better solution by modifying the other routes that were not marked and visiting the clients that were not marked, which causes solutions to be exploited in different regions of the search space (line 10). If the solution obtained is not dominated by some solution in the population, it is included within it.

C. Single State GRASP method

Fig. 2 shows the general pseudo-code of the Single State GRASP method. In line 1 is the construction of the initial solution (best solution, bestSol). This construction is carried out by choosing one of two initialization methods that are explained later in section D. Choosing the first method of initialization is done based on a percentage (parameter pl) that corresponds to a parameter defined by the user. In initialization regardless of method, the rls parameter that corresponds to the restricted list size is used when creating a solution with classic GRASP metaheuristics. If Single State GRASP receives a solution (sol other than null), it does not perform the initialization process and takes that solution as a starting point.

A check is carried out in line 2 that the initial solution is valid (if it was started from an existing solution in the population, the validation is not carried out since the solutions are always built in the space of feasible solutions). This means that it complies with the restrictions of the problem such as capacity per route, customer service times (time windows), the arrival of vehicles at the depot within the stipulated times, that all customers are served and the other restrictions inherent to the VRPTW problem. If an initial invalid solution is found, null is returned.

Between lines 3 and 8 the optimization cycle is carried out, which is executed until a maximum number of optimization iterations (mnoi) is reached. In line 5, a copy of the best current solution (bestSol) is made in a variable called currentSol. This variable goes to the local optimization process in line 5, which is modified a specific number of times (om parameter). This process is explained in more detail later. In line 6, if the current solution dominates the best solution, it replaces the best one in such a way that this new solution is the one to which optimization is applied in the next iteration. Otherwise, the one that was already the best solution is left. If the algorithm's execution time (from the start of MOMGRASP) exceeds the maximum execution time defined by the user (line 7), the cycle is broken to exit from Single State GRASP method and return the best solution found (line 9) to the MOMGRASP method.

Fig. 3 shows the pseudocode of the Single State GRASP local optimization method (Local Optimizer). This method receives a solution from Single State GRASP and assumes that it is the best solution (bestSol) from the input parameter. In the loop of lines 1 to 7, an optimization operation is executed and if the solution obtained is better, it replaces it to make the next cycle of improvement on this one. Otherwise, the one that is currently considered the best is continued (line 5). If the execution time of the algorithm (from the start of MOMGRASP) exceeds the maximum execution time defined by the user (line 6), the cycle is broken to exit from Local Optimizer method and return the best solution (line 8) found to the Single State GRASP method.

```
SingleStateGRASP (problem pv, objectives o, seed s,
        maximum execution time tm, probability pl,
        restricted list size rls, maximum number of
        optimization iterations mnoi, maximum
        number of operations mno, operator's
        configuration oc, solution sol, routes to
        preserve rtp, clients visited cv)
1.  if sol is null then bestSol = InitializeRoutes (pl, rls)
        else bestSol = sol
2.  if sol is null and Validate (bestSol) = false then
        return null
3.  for i = 1 to mnoi do
4.      currentSol = CopyOf (bestSol)
5.      LocalOptimizer (currentSol, mno, oc, rtp, cv)
6.      if (currentSol dominates to bestSol) then
            bestSol = currentSol
7.      if tm has been reached, then break for
8.  end for
9.  return bestSol
```

Fig. 2 Single State GRASP method pseudo-code

```
LocalOptimizer (problem pv, objectives o, seed s,
        maximum execution time tm, maximum
        number of operations mno, operator's
        configuration oc, solution bestSol, routes to
        preserve rtp, clients visited cv)
1.  for j = 1 to mno do
2.      if oc = 1 then currentSol =
            ApplyGRASPOperators (bestSol, rtp, cv)
3.      if oc = 2 then currentSol =
            ApplyLSMOVRPTWOperators (bestSol, rtp,
            cv)
4.      if oc = 3 then currentSol =
            ApplyMMOEASAOperators (bestSol, rtp,
            cv)
5.      if (currentSol dominates to bestSol) then
            bestSol = currentSol
6.      if tm has been reached then break for
7.  end for
8.  return bestSol
```

Fig. 3 Local Optimizer method pseudo-code used by Single State GRASP

The Local Optimizer is configured to use the operators of the state of the art algorithm LSMOVRPTW (Local Search-Based Multiobjective Optimization Algorithm for Multiobjective Vehicle Routing Problem with Time Windows) [5]. If the parameter oc (operators configuration) is equal to 2 (line 3), or use the operators of the state of the art algorithm MMOEASA (a multi-start multi-objective evolutionary algorithm with simulated annealing) [12] If the parameter oc is equal to 3 (line 4), or use the operators presented later in section E which are defined for the 3 objectives of interest in this work (line 2). Existing operators and new ones include specific knowledge of the problem (memetic approach).

D. Initializing a Solution

Considering that most of the state-of-the-art articles express that the creation of the initial solutions should be done based on the early attention of the clients who have a closer start window. In Single State GRASP (line 1, Initialize Routes method), this approach is adapted to create two initialization methods. The first initialization that is executed with a probability pl, seeks to build routes sequentially and has the following steps:

1. Customers are sorted in a list by their start time in the service window.
2. The depot at the beginning and at the end is added to all routes. The first customer from the list built-in step 1 is added to the first route, and that customer is deleted from that list.
3. The last customer of the current route is taken (route 1 at the beginning). An order is made of the customers that remain to be attended (list in step 1) by their proximity in travel time to this last customer, that is, the distance to the customer plus the waiting time to serve them, from the list a customer is taken by raising who is in the first positions based on the parameter rls (size of the restricted list). If that customer can be added to the route complying with the restrictions, it is added and removed from the list in step 1. Now this client (last of the current route) is taken, and the same process is applied again until no more clients can be added to the route current.
4. Step 3 is repeated with the following routes (2, 3 and so on).
5. The attention of all the clients to the problem is verified to make sure to deliver a feasible solution.

In the second initialization that is executed with a probability 1-pl, it is sought to assign clients to the routes where the last client is closest and is carried out with the following steps:

1. Customers are sorted in a list by their start time in the service window.
2. The depot at the beginning and at the end is added to all routes. The first customer from the list built in step 1 is added to the first route and that customer is deleted from that list.
3. Take the next customer on the list and create a list L1 of how close this customer is to the last customer on each route (at first it is compared against the first customer on route 1 and for the other routes with the depot because no clients have been assigned). From this L1 list, a route from the closest ones is randomly chosen (based on the restricted list size parameter, rls) and the client is assigned to that route if they comply with the restrictions, if not, the process is repeated.
4. Step 3 is repeated with the remaining clients (3, 4, and so on).
5. The attention of all the clients of the problem is verified, to make sure to deliver a feasible solution.

E. Operators

The operators of the algorithm proposed in this research are three (3), namely:

- **Operator 1:** Takes a random route and from this a random client takes afterwards it evaluates different positions of other routes to relocate it to the position where it achieves the best fitness of the objective to be optimized.
- **Operator 2**: Takes a specific number of clients from a route chosen at random and proceeds to relocate them to the best positions on other routes, bearing in mind that one of the three objectives to be optimized can be improved.
- **Operator 3:** Takes a sequence of a random number of clients, from a position of a client chosen at random, from a route also selected at random, and proceeds to test the insertion of this ordered sequence of clients in another route. The Insertion must preserve order and the location will be where one of the three objectives can be minimized to the greatest extent.

F. Experimental setup

The proposed algorithm was evaluated and compared with two state-of-the-art algorithms, LSMOVRPTW [5] and MMOEASA 14], with datasets that the authors of these algorithms used to evaluate their proposals using four test scenarios (A, B, C, and D). The measure used to evaluate the quality of the results obtained was Hypervolume (higher values are better), and it was obtained as the average of 31 executions (repetitions) of each algorithm in each dataset (problem) using different seeds of random numbers to thereby generate different initial solutions.

1) Test problems (datasets): For this work, the problems proposed in [29] (available at https://github.com/psxjpc) were selected for the comparison of Population GRASP against LSMOVRPTW. These problems are organized into three groups by the number of clients: 50, 150, and 250.

The problems proposed by Solomon [30] (available at http://w.cba.neu.edu/~msolomon/problems.htm) were also used, which are divided into 6 categories, random R1 (12 in total) and R2 (11 in total), grouped C1 (9 in total) and C2 (8 in total) and semi-grouped RC1 (8 in total) and RC2 (8 in total) that give a total of 56 instances. These were used for the evaluation of the proposed algorithm versus the MMOEASA and LSMOVRPTW algorithms.

2) Optimum values for test problems: Considering that the datasets reported in the literature do not have the optimal reference values for the objectives (maximum and minimum), it was necessary to calculate these. To achieve this, the three algorithms (MOMGRASP, LSMOVRPTW and MMOEASA) were executed 30 times each for 100 seconds. All the initial and final solutions of the same problem (dataset) were organized in two files. The non-dominated solutions (Pareto front) were selected from the second file. Then, for each objective, the arithmetic mean ($\bar{x}$) and the standard deviation (σ) of the values of the objectives were calculated both for the solutions resulting from the initialization process (to calculate the maximums), and for the non-dominated solutions (for calculate the minimums). The maximum values of the objectives were calculated as follows: $\max = \bar{x} + (3.72 * \sigma)$, while the minimum values were calculated as follows: $\min = \bar{x} - (3.72 * \sigma)$. The 3.72 makes it possible to have a virtually 100% probability of taking all the data based on a normal distribution of the data.

3) Test scenarios: The proposed algorithm (MOMGRASP) was evaluated in four (4) scenarios, namely: A - against LSMOVRPTW algorithm over 5 objectives (number of vehicles, total travel distance, travel time of the longest route, total waiting time due to early arrivals, and total delay time due to late arrivals) and using the operators of LSMOVRPTW for a fair comparison; B – against MMOEASA algorithm over 2 objectives (total travel distance and balance of distance between routes - case 1) and using the MMOEASA operators for a fair comparison; C - against MMOEASA algorithm over 2 objectives (total travel distance and balance of workload - case 2) and using the MMOEASA operators; D - against LSMOVRPTW and MMOEASA algorithms over 3 objectives (total travel time, waiting time of customers to be attended, and balance of total travel time between routes) and using the MOMGRASP operators in all algorithms for a fair comparison.

4) Software for experimentation: Considering the large amount of time it takes to execute the experiments (approximately 28 days on a single computer), a task distribution scheme was implemented with a server that deploys a web service that allows assigning MOVRPTW optimization tasks to customer computers and another web service to record the results reported by those clients. The client computers execute an optimization algorithm on a dataset as defined in the task, and when it finishes it requests the results record and if there are still tasks to be executed on the server it takes one and solves it. On the server computer, the tasks that are expected to be developed are configured, storing them in the database, and then when the client computers request tasks, the server delivers them, marking them so as not to assign them later to other client computers. Then, when a client computer finishes the task, it registers in the database through the other web service. With the above, the database has at the end, the record of the results of all the tasks, tasks that include the algorithm to be executed, the dataset or specific problem to be solved, the objectives to be optimized, the specific parameters with which the algorithm is executed, the value of the hypervolume (HV) and the solutions of the Pareto front. The client computers were a total of 18 computers with an AMD A10 PRO-7800B R/ processor, 12 Compute Cores C + 8G 3.50GHz and 8 GB RAM.

5) Parameter tuning: In the proposed algorithm 3 parameters were tuned, the percentage of use of the first initialization method (probability pl), the restricted list size (rls) and the factor of routes that are preserved (FCR) for the second cycle of improvement. To tune these parameters, Covering Arrays (CAs) were used, avoiding exhaustive tests, and thus reducing the parameter tuning time. This fine-tuning method has been used with success in different previous works [31], [32]. CA was defined as strength 2 and with an alphabet of 4 for each parameter. For the first parameter, the values 0.3, 0.4, 0.5, 0.6; for the second parameter 3, 4, 5, 6 and for the third parameter the values 1.42, 1.66, 2, 2.5. The exhaustive tests that were going to be 64 (4 * 4 * 4) trials were reduced to 17. In addition, only 6 datasets were used (reducing the possible overfitting of the results), 2 for each category and the processing was parallelized in the 18 client computers. Table I shows the parameters used for each operator's configuration and experimentation scenarios.

III. Results And Discussion

Table II shows the values obtained for the average hypervolume for the two algorithms (MOMGRASP and LSMOVRPTW) on test scenario A. In this scenario, the MOMGRASP algorithm had better (higher) hypervolume than LSMOVRPTW in 35 datasets out of the 45 (77.8%). MOMGRASP had better results in all class of datasets (50 clients, 150 clients and 250 clients), winning roughly in 12 out of 15 of each (approximately in 80% of each class). Friedman nonparametric test reports that MOMGRASP algorithm is better than LSMOVRPTW with a p-value of 0.00019 (<0.05) and a Friedman statistic of 13.88888.

TABLE I
PARAMETERS USED FOR EXPERIMENTS

Objectives	pl	rls	FCR
5 objectives of LSMOVRPTW	0,3	5	0,602
2 objectives of MMOEASA - case 1	0,6	3	0,602
2 objectives of MMOEASA - case 2	0,6	3	0,602
3 objectives of MOMGRASP	0,6	3	0,602

TABLE II
GENERAL RESULTS FOR MOMGRASP, LSMOVRPTW AND MMOEASA IN THREE SCENARIOS

Test Scenario	Measure	Populational GRASP	LSMOVRPTW	MMOEASA
A	Avg. HV	0.195806	0.154911	-
	Best 50	11	4	-
	Best 150	12	3	-
	Best 250	12	3	-
	Best	35	10	-
B	Avg. HV	0.814427	-	0.807543
	Best C	14	-	3
	Best R	12	-	11
	Best RC	3	-	13
	Best	29	-	27
C	Avg. HV	0.729781		0.763323
	Best C	1	-	16
	Best R	7	-	16
	Best RC	4	-	12
	Best	12	-	44

Table II also shows the obtained average hypervolume values for MOMGRASP and MMOEASA algorithms on test scenario B. For this case, the MOMGRASP algorithm had higher hypervolume than MMOEASA in only 29 instances of the 56 (51.8%). MOMGRASP obtains better results in instances C (grouped type and easy to solve) and in instances R (random type and more difficult to solve), where it was better in 12 of 23 (52.2%), but the results are poor in instances RC (semi-grouped type). The Friedman nonparametric test reports that there is no significant difference in mean values (p-value of 0.78927) of this scenario.

Table II also shows the values obtained for average hypervolume for MOMGRASP and MMOEASA algorithms on test scenario C. For this case, the MMOEASA technique

had higher hypervolume than MOMGRASP in 44 datasets of the 56 (78.6%) and in all dataset types (random, grouped, and semi-grouped) it was better in 12 datasets or more. The Friedman nonparametric test reports that MMOEASA is better than MOMGRASP with a p-value of 1.901E-5 (<0.05) and a Friedman statistic of 18.28571.

Table III shows the mean hypervolume values for the three algorithms on test scenario D. In each row, the highest value for the respective dataset (problem) is presented with an asterisk. In this case, the MOMGRASP technique was the best technique with the highest hypervolume in 41 datasets of the 56 (73.2%), and in problems R (random type and with greater difficulty to solve), it was better in 21 of 23 (91.3%). In the second place, MMOEASA was found with the highest hypervolume in 12 of the 56 problems (21.4%), and in type R problems it obtained 2 hypervolume values that were better (8.7%). Finally, the algorithm with the lowest hypervolume value was LSMOVRPTW, winning in 3 of the 56 datasets (5.4%) and in none of the R-type datasets.

In addition to this, it is evident that the datasets for which LSMOVRPTW or MMOEASA wins are mostly grouped in category C (Clustered Clients) - with 6 cases for MMOEASA and none for LSMOVRPTW - and RC (Semi Clustered Clients) - with 3 cases in which MMOEASA wins and 3 in which LSMOVRPTW wins. These results may be due to the simpler structure of LSMOVRPTW and MMOEASA that allows them to create and evaluate more solutions simultaneously than MOMGRASP, which helps them when the problem is simpler to resolve. However, the extra time it takes for MOMGRASP to search different regions of the solution space allows it to get better results on the most difficult category R problems.

The Friedman nonparametric test reports that the results of MOMGRASP come in first in the ranking, followed by MMOEASA and LSMOVRPTW with a p-value of 1.979E-14 (<0.05) and a Friedman statistic of 63.107. The Wilcoxon signed test shows that the results of MOMGRASP are statistically better than those obtained by MMOEASA and LSMOVRPTW, and the results obtained by MMOEASA are better than those obtained by LSMOVRPTW.

Based on the results presented in the previous tables, it is evident that the Population GRASP algorithm is, on average, better at solving MOVRPTW problems than the LSMOVRPTW and MMOEASA algorithms by obtaining better hypervolume values in 3 of the four evaluated scenarios. It is important to note that the results were better (statistically speaking) in 2 of the 3 scenarios evaluated.

TABLE III
MEAN HYPERVOLUME FOR GRASP, LSMOVRPTW AND MMOEASA WITH 3 OBJECTIVES (TEST SCENARIO D)

Dataset	Mean hypervolume		
	Populational GRASP	LSMOVRPTW	MMOEASA
c101	0.528260	0.395171	0.572061 *
c102	0.600100 *	0.379695	0.564227
c103	0.562528 *	0.424531	0.486334
c104	0.655225 *	0.463461	0.509355
c105	0.519515 *	0.414634	0.489361
c106	0.522354 *	0.345378	0.465299
c107	0.456147	0.354747	0.464556 *
c108	0.499508 *	0.378500	0.422958
c109	0.516369 *	0.413881	0.443786
c201	0.619604	0.456444	0.707160 *
c202	0.583739 *	0.570456	0.577368
c203	0.600100 *	0.544819	0.539503
c204	0.597830 *	0.494517	0.442941
c205	0.618218	0.506289	0.689901 *
c206	0.606931	0.516212	0.658527 *
c207	0.607466 *	0.518960	0.592833
c208	0.616000	0.474917	0.645634 *
r101	0.566541	0.258998	0.580007 *
r102	0.548540 *	0.173072	0.327957
r103	0.515323 *	0.182282	0.326287
r104	0.520792 *	0.184636	0.338129
r105	0.575079	0.465485	0.613484 *
r106	0.587514 *	0.337585	0.417290
r107	0.661440 *	0.376360	0.487487
r108	0.680890 *	0.390823	0.513461
r109	0.754385 *	0.600738	0.728779
r110	0.740928 *	0.534670	0.588620
r111	0.538709 *	0.278150	0.513030
r112	0.829284 *	0.536445	0.570785
r201	0.633107 *	0.534328	0.435269
r202	0.633651 *	0.492317	0.507840
r203	0.592544 *	0.475165	0.475679
r204	0.562594 *	0.288127	0.419086
r205	0.630430 *	0.499747	0.619663
r206	0.643737 *	0.537211	0.552724
r207	0.625506 *	0.497641	0.556514
r208	0.598985 *	0.334917	0.449362
r209	0.643405 *	0.491637	0.565669
r210	0.596368 *	0.328392	0.463461
r211	0.604551 *	0.546226	0.567945
rc101	0.587630	0.482087	0.634470 *
rc102	0.573753 *	0.346066	0.433272
rc103	0.661290 *	0.411882	0.526581
rc104	0.522674 *	0.271624	0.518995
rc105	0.496100 *	0.287144	0.262144
rc106	0.703848	0.583576	0.728921 *
rc107	0.709239 *	0.521584	0.545812
rc108	0.590758	0.405941	0.591275 *
rc201	0.562897	0.609762 *	0.477835
rc202	0.596940 *	0.592544	0.469020
rc203	0.558211 *	0.488075	0.470694
rc204	0.545555 *	0.389162	0.414218
rc205	0.512501	0.521544 *	0.347995
rc206	0.599962	0.523235	0.609238 *
rc207	0.619014	0.625470 *	0.524144
rc208	0.595777 *	0.579961	0.550126
Avg. HV	0.596970 *	0.439950	0.517769
Best R	21	0	2
Best	41	3	12

IV. CONCLUSION

The proposed way of adapting the original version of GRASP to solve a problem with multi or many objectives using specific knowledge of the problem (memetic approach) - in this case, MOVRPTW - can be used to convert other single objective metaheuristics and use them to solve different problems from a multiobjective memetic approach.

The proposed multiobjective memetic adaptation of GRASP is a good option for solving MOVRPTW problems and obtaining competitive or better results than those reported in the state of the art for LSMOVRPTW and MMOEASA. The Friedman nonparametric test shows with 95% confidence that MOMGRASP obtained better results in two (2) out of four (4) test scenarios, similar results to MMOEASA in one (1) test scenario and worse results in another scenario against MMOEASA. The experiments showed that MOMGRASP

beat LSMOVRPTW and that it overwhelmingly beat both algorithms in its own experimentation scenario (three objectives) of initial interest of the work.

As future work, it is expected to study the behavior of the proposed algorithm by including an adaptive parameter that defines the percentage of times that an operator should be used in a specific problem, seeking to use more times the operator that makes the greatest improvements to the solutions. The results obtained are expected to be transferred to a company with a MOVPRTW problem in Colombia. It is also expected to compare the result of the work carried out with the most recent publications in the field, for example, against the use of specific variation operators in NSGA-II and the hybrid genetic algorithm of two phases (ruin and recreation).

ACKNOWLEDGMENTS

The Universidad del Cauca partially funded this work. The authors thank Colin McLachlan for translating this document.

REFERENCES

[1] J. Caceres-Cruz, P. Arias, D. Guimarans, D. Riera, and A. A. Juan, "Rich vehicle routing problem: Survey," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 1–28, 2014, doi: 10.1145/2666003.

[2] A. Dixit, A. Mishra, and A. Shukla, "Vehicle routing problem with time windows using meta-heuristic algorithms: A survey," in *Advances in Intelligent Systems and Computing*, 2019, vol. 741, pp. 539–546, doi: 10.1007/978-981-13-0761-4_52.

[3] S. Ben Hamida, R. Gorsane, and K. Gorsan Mestiri, "Towards a better understanding of genetic operators for ordering optimization: Application to the capacitated vehicle routing problem," in *15th International Conference on Software Technologies, ICSOFT 2020*, 2020, pp. 461–469.

[4] R. S. Kumar, K. Kondapaneni, V. Dixit, A. Goswami, L. S. Thakur, and M. K. Tiwari, "Multi-objective modeling of production and pollution routing problem with time window: A self-learning particle swarm optimization approach," *Comput. Ind. Eng.*, vol. 99, pp. 29–40, 2016, doi: 10.1016/j.cie.2015.07.003.

[5] Y. Zhou and J. Wang, "A Local Search-Based Multiobjective Optimization Algorithm for Multiobjective Vehicle Routing Problem with Time Windows," *IEEE Syst. J.*, vol. 9, no. 3, pp. 1100–1113, 2015, doi: 10.1109/JSYST.2014.2300201.

[6] H. Yousefi, R. Tavakkoli-Moghaddam, M. Taheri Bavil Oliaei, M. Mohammadi, and A. Mozaffari, "Solving a bi-objective vehicle routing problem under uncertainty by a revised multichoice goal programming approach," *Int. J. Ind. Eng. Comput.*, vol. 8, no. 3, pp. 283–302, 2017, doi: 10.5267/j.ijiec.2017.1.003.

[7] E. T. Yassen, M. Ayob, M. Z. A. Nazri, and N. R. Sabar, "An adaptive hybrid algorithm for vehicle routing problems with time windows," *Comput. Ind. Eng.*, vol. 113, pp. 382–391, 2017, doi: 10.1016/j.cie.2017.09.034.

[8] V. F. Yu, T. Iswari, N. M. E. Normasari, A. M. S. Asih, and H. Ting, "Simulated annealing with restart strategy for the blood pickup routing problem," in *IOP Conference Series: Materials Science and Engineering*, 2018, vol. 337, no. 1, doi: 10.1088/1757-899X/337/1/012007.

[9] J. Wang, W. Ren, Z. Zhang, H. Huang, and Y. Zhou, "A Hybrid Multiobjective Memetic Algorithm for Multiobjective Periodic Vehicle Routing Problem with Time Windows," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 50, no. 11, pp. 4732–4745, 2020, doi: 10.1109/TSMC.2018.2861879.

[10] A. Agárdi, L. Kovács, and T. Bányai, "Optimization of multi-depot periodic vehicle routing problem with time window," *Acad. J. Manuf. Eng.*, vol. 17, no. 4, pp. 96–108, 2019.

[11] M. Gmira, M. Gendreau, A. Lodi, and J.-Y. Potvin, "Tabu search for the time-dependent vehicle routing problem with time windows on a road network," *Eur. J. Oper. Res.*, vol. 288, no. 1, pp. 129–140, 2021, doi: https://doi.org/10.1016/j.ejor.2020.05.041.

[12] R. Baños, J. Ortega, C. Gil, A. L. Márquez, and F. De Toro, "A hybrid meta-heuristic for multi-objective Vehicle Routing Problems with Time Windows," *Comput. Ind. Eng.*, vol. 65, no. 2, pp. 286–296, 2013, doi: 10.1016/j.cie.2013.01.007.

[13] A. B. Pratiwi, A. Pratama, I. Sa'diyah, and H. Suprajitno, "Vehicle routing problem with time windows using natural inspired algorithms," in *Journal of Physics: Conference Series*, 2018, vol. 974, no. 1, doi: 10.1088/1742-6596/974/1/012025.

[14] J. Chen and J. Shi, "A multi-compartment vehicle routing problem with time windows for urban distribution – A comparison study on particle swarm optimization algorithms," *Comput. Ind. Eng.*, vol. 133, pp. 95–106, 2019, doi: 10.1016/j.cie.2019.05.008.

[15] N. Rezaei, S. Ebrahimnejad, A. Moosavi, and A. Nikfarjam, "A green vehicle routing problem with time windows considering the heterogeneous fleet of vehicles: Two metaheuristic algorithms," *Eur. J. Ind. Eng.*, vol. 13, no. 4, pp. 507–535, 2019, doi: 10.1504/EJIE.2019.100919.

[16] L. Deng and J. Zhang, "A Hybrid Ant Colony Optimization for Bi-Objective VRP with Time Windows," *Complex Syst. Complex. Sci.*, vol. 17, no. 4, pp. 73–84, 2020, doi: 10.13306/j.1672-3813.2020.04.009.

[17] M. Song, J. Li, Y. Han, Y. Han, L. Liu, and Q. Sun, "Metaheuristics for solving the vehicle routing problem with the time windows and energy consumption in cold chain logistics," *Appl. Soft Comput.*, vol. 95, p. 106561, 2020, doi: https://doi.org/10.1016/j.asoc.2020.106561.

[18] G. Srivastava, A. Singh, and R. Mallipeddi, "NSGA-II with objective-specific variation operators for multiobjective vehicle routing problem with time windows," *Expert Syst. Appl.*, vol. 176, p. 114779, 2021, doi: https://doi.org/10.1016/j.eswa.2021.114779.

[19] T. S. Khoo and B. B. Mohammad, "The parallelization of a two-phase distributed hybrid ruin-and-recreate genetic algorithm for solving multi-objective vehicle routing problem with time windows," *Expert Syst. Appl.*, vol. 168, p. 114408, 2021, doi: https://doi.org/10.1016/j.eswa.2020.114408.

[20] H. Zhang, Q. Zhang, L. Ma, Z. Zhang, and Y. Liu, "A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows," *Inf. Sci. (Ny).*, vol. 490, pp. 166–190, 2019, doi: https://doi.org/10.1016/j.ins.2019.03.070.

[21] J. C. Molina, J. L. Salmeron, and I. Eguia, "An ACS-based memetic algorithm for the heterogeneous vehicle routing problem with time windows," *Expert Syst. Appl.*, vol. 157, p. 113379, 2020, doi: https://doi.org/10.1016/j.eswa.2020.113379.

[22] A. K. Ariyani, W. F. Mahmudy, and Y. P. Anggodo, "Hybrid genetic algorithms and simulated annealing for multi-trip vehicle routing problem with time windows," *Int. J. Electr. Comput. Eng.*, vol. 8, no. 6, pp. 4713–4723, 2018, doi: 10.11591/ijece.v8i6.pp.4713-4723.

[23] J. Euchi, S. Zidi, and L. Laouamer, "A Hybrid Approach to Solve the Vehicle Routing Problem with Time Windows and Synchronized Visits In-Home Health Care," *Arab. J. Sci. Eng.*, vol. 45, no. 12, pp. 10637–10652, 2020, doi: 10.1007/s13369-020-04828-5.

[24] P. Jiang, J. Men, H. Xu, S. Zheng, Y. Kong, and L. Zhang, "A Variable Neighborhood Search-Based Hybrid Multiobjective Evolutionary Algorithm for HazMat Heterogeneous Vehicle Routing Problem with Time Windows," *IEEE Syst. J.*, vol. 14, no. 3, pp. 4344–4355, 2020, doi: 10.1109/JSYST.2020.2966788.

[25] M. Liu, Y. Shen, Q. Zhao, and Y. Shi, "A Hybrid BSO-ACS Algorithm for Vehicle Routing Problem with Time Windows on Road Networks," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, 2020, pp. 1–8, doi: 10.1109/CEC48606.2020.9185868.

[26] Y. Shen, M. Liu, J. Yang, Y. Shi, and M. Middendorf, "A hybrid swarm intelligence algorithm for vehicle routing problem with time windows," *IEEE Access*, vol. 8, pp. 93882–93893, 2020, doi: 10.1109/ACCESS.2020.2984660.

[27] A. Pessoa, R. Sadykov, and E. Uchoa, "Enhanced Branch-Cut-and-Price algorithm for heterogeneous fleet vehicle routing problems," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 530–543, 2018, doi: https://doi.org/10.1016/j.ejor.2018.04.009.

[28] R. F. Fachini and V. A. Armentano, "Logic-based Benders decomposition for the heterogeneous fixed fleet vehicle routing problem with time windows," *Comput. Ind. Eng.*, vol. 148, p. 106641, 2020, doi: https://doi.org/10.1016/j.cie.2020.106641.

[29] J. Castro-Gutierrez, D. Landa-Silva, and J. Moreno Pérez, "Nature of real-world multi-objective vehicle routing with evolutionary algorithms," in *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 2011, pp. 257–264, doi: 10.1109/ICSMC.2011.6083675.

[30] M. M. Solomon, "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints," *Oper. Res.*, vol. 35, no. 2, pp. 254–265, 1987, doi: 10.1287/opre.35.2.254.

[31] C. Cobos, A. Paz, J. Luna, C. Erazo, and M. Mendoza, "A Multi-Objective Approach for the Calibration of Microscopic Traffic Flow Simulation Models," *IEEE Access*, vol. 8, pp. 103124–103140, 2020, doi: 10.1109/ACCESS.2020.2999081.

[32] E. Ruano-Daza, C. Cobos, J. Torres-Jimenez, M. Mendoza, and A. Paz, "A multiobjective bilevel approach based on global-best harmony search for defining optimal routes and frequencies for bus rapid transit systems," *Appl. Soft Comput. J.*, vol. 67, pp. 567–583, Jun. 2018, doi: 10.1016/j.asoc.2018.03.026.