Computer Science • 18(3) 2017 http://dx.doi.org/10.7494/csci.2017.18.3.1809

Krzysztof Podlaski
Grzegorz Wiatrowski

MULTI-OBJECTIVE OPTIMIZATION OF VEHICLE ROUTING PROBLEM USING EVOLUTIONARY ALGORITHM WITH MEMORY

Abstract *The idea of a new evolutionary algorithm with a memory aspect included is proposed to find a multi-objective optimized solution for the vehicle routing problem with time windows. This algorithm uses a population of agents that individually search for optimal solutions. The agent memory incorporates the process of learning from the experience of each individual agent as well as from the experience of the whole population. This algorithm uses crossover operation to define each agent's evolution. In this paper, we choose the Best Cost Route Crossover (BCRC) operator as a base. This operator is well-suited for VPRTW problems; however, it does not treat both parents symmetrically, which is not natural for general evolutionary processes. A part of the paper is devoted to finding an extension of the BCRC operator in order to improve the inheritance of chromosomes from both parents. Thus, the proposed evolutionary algorithm is implemented with the use of two crossover operators: BCRC and its extended-modified version. We analyze the results obtained from both versions applied to Solomon's and Gehring & Homberger's instances. We conclude that the proposed method with the modified version of the BCRC operator gives statistically better results than those obtained using the original BCRC. It seems that an evolutionary algorithm with the memory and modification of the Best Cost Route Crossover Operator leads to very promising results when compared to other solutions presented in the literature.*

Keywords vehicle routing problem, time windows, evolutionary algorithms, multi-objective optimization

Citation Computer Science 18(3) 2017: 269–286

1. Introduction

The Vehicle Routing Problem (VRP) is one of the most-recognized of all of the known NP-hard problems in computer science. The problem was proposed by Dantzig and Ramser in 1959's [6]. Since then, many versions of VRP have been introduced to the literature; e.g., with a time window, truck capacity, trip multiplicity, multi-depots, and other constraints [30]. This paper focuses on the Vehicle Routing Problem with Time Windows (VRPTW).

VRPTW can be described as a task for a single depot with an associated fleet of trucks delivering cargo to many customers with respect to their time constraints. Here are the following rules that must be taken into account during the search for an optimal solution:

- each customer can be visited only once,
- each customer has defined a demand of cargo,
- each customer has defined an opening time window (ready and due time),
- each customer has defined a service time,
- the time of travel between customers is equal to the distance,
- each truck has defined a capacity and has to deliver supplies to customers within the defined time windows,
- a vehicle that arrives too early to a customer has to wait until service would start at the open time for a given customer,
- the customer cannot be serviced if a vehicle arrives after the due time.

Many different approaches to solve VRP have already been proposed: exact [8,12, 31], tabu search [5,20], heuristic [1,14], genetic algorithms [2–4,19,21,29], and swarm intelligence [9,17,23,29]. In order to be able to compare the numerical results of different methods, some classified benchmark tasks were defined. The best-known benchmarks are known as Solomon's benchmarks [27] and Gehring and Homberger's [10].

The complexity of VRPTW problems makes all approaches that use the exact methods computationally difficult. For large instances with 100 customers or more, it is rarely possible to obtain an optimal route in a reasonable amount time [16]. Moreover, none of existing exact methods can optimally solve all VRPTW benchmark instances with at least 100 customers. For these reasons, the meta-heuristic methods seems to be a better choice for these types of problems.

In this paper, we present an evolutionary algorithm with memory that can be used to solve VRPTW problems. The algorithm uses a population of agents to search for an optimal solution for a given task; each agent stores an actual solution and, additionally, keeps the best historical results already obtained in its memory. The actual solution is the position of an agent in the search space of a VRPTW problem, and an agent movement in this space is obtained due to agent evolution. During the exploration and exploitation processes, the agents evolve and search for better solutions; the new solutions for an agent are created on the basis of its actual solution and historical information gathered using crossover operations. This approach has

been applied to the known benchmarks. After the first numerical experiments, we have discovered that the used crossover operator (Best Cost Route Crossover Operator BCRC) can be modified in order to obtain better results. The mentioned modification is also presented in this paper. The experimental results presented in this paper are interesting and comparable with best-known results from the literature.

The article is organized as follows. Section 2 describes how the aspect of memory can be introduced into the population of agents used in evolutionary algorithm. Section 3 is devoted to the BCRC crossover operator for VRPTW and its modification. Section 4 summarizes the obtained numerical results, while Section 5 focuses on their statistical analysis. Section 6 concludes the paper.

2. Evolutionary algorithm for VRPTW

Evolutionary Algorithms (EA) are based on the evolution of a population of agents, and this evolution should provide a way to search for at least a near-optimal solution for a given problem. A fitness measure is implemented in order to control the evolution of the population in the right direction. The task is to steer the population to search for the most-optimal solution to a given problem. Usually, Evolutionary Algorithms are connected strictly to Genetic Algorithms (GA), but this is not the only possible case. In this paper, we propose an evolutionary algorithm with the use of its memory designed specifically to solve VRPTW tasks.

The algorithm is based on a population of agents, each of which lives in the space of solutions for a given VRPTW instance. The agent stores information about its actual solution (position in search space); additionally, it remembers the best solution already found by this agent and the best solution found by any member of the population. During the evolution process, an agent changes its actual solution (move in search space). The evolutionary changes of the solution for an agent takes into account its actual solution and the historical data stored (memory effect). The solutions of the VRPTW instance is given by chromosome representation known from the usual GA approaches to VRPTW, and the changes of a solution is implemented using a genetic crossover operator.

2.1. Chromosome representation of VRPTV solution

Solution s of the VRPTV problem has to contain information about the routes of all vehicles v_a used in the solution. Each of the vehicles starts and ends at the depot. Then, it services its customers, which means it forms an ordered list of all customers to be visited by a given vehicle.

The customers are denoted by an unique integer number ($c > 0$), while the depot is denoted by 0. Each solution is a sequence of serviced customers for all vehicles separated with 0. For example, sequence s in form $s = \{0, 2, 4, 5, 0, 1, 7, 0, 3, 6, 0\}$ describes a solution of the VRPTW problem that uses three trucks to service seven customers.

The overall length of route s is a sum of the distances covered by all vehicles used in the solution.

$$\begin{aligned} |s| &= \textstyle\sum_a |v_a|, \\ |v_a| &= \textstyle\sum_b dist(c_a^b, c_a^{b+1}) \end{aligned} \tag{1}$$

where c_a^b denotes a customer serviced by vehicle v_a, and function $dist(,)$ is the distance between two customers. The graphical representation of such a solution is presented in Figure 1. Additionally, solution s has to fulfill all additional constraints, such as truck capacity and the time constraints of the customers.

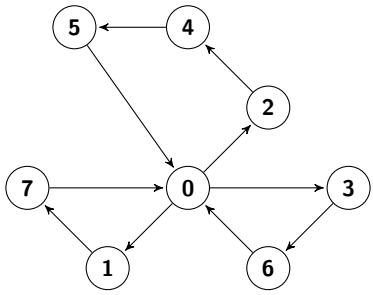

Figure 1. Graphical representation of solution $s = \{0, 2, 4, 5, 0, 1, 7, 0, 3,\ 6, 0\}$ for VRPTW problem with seven customers and three vehicles.

2.2. Evolution of the population

As mentioned above, we record actual solution s_k and best historical solution s_k^{ab} for each member of the population; additionally, we also record the population's best solution (denoted as s^{pb}).

The evolution of the k-th agent is ruled by the following equation:

$$s_{k,i} = \begin{cases} s_{k,i-1} \otimes s_k^{ab}, & \text{with probability } \nu^{ab} \\ s_{k,i-1} \otimes s^{pb}, & \text{with probability } \nu^{pb}, \end{cases} \tag{2}$$

where $\otimes$ denotes the crossover operator.

Probabilities ν^{ab} and ν^{pb} depend on the overall lengths of s_k^{ab} and s^{pb}:

$$\begin{aligned} \nu^{ab} &= \tfrac{|s^{pb}|}{|s_k^{ab}|+|s^{pb}|}, \\ \nu^{pb} &= \tfrac{|s_k^{ab}|}{|s_k^{ab}|+|s^{pb}|}, \\ \nu^{ab} &+ \nu^{pb} = 1, \end{aligned} \tag{3}$$

where $s_{k,i}$ is the solution obtained by the k-th agent in the i-th iteration of the algorithm. In the presented approach, genetic crossover operator $\otimes$ is used for the generation of a new solution for an agent and represents a rule that defines the evolution of the agent. There are many available crossover operators in the literature. As the basis for our approach, we have chosen the Best Cost Route Crossover Operator (BCRC) [21].

From (2) and (3), we can observe that, when the historically-best solution found by agent s_k^{ab} is much longer than the best solution found by population s^{pb}, the agent prefers to operate with the population's best solution. Thus, the proposed evolutionary algorithm for the VRPTW problem has the following form.

Algorithm 1 Evolutionary algorithm

```
1: Begin
2: initialize the population,
3: repeat
4:    for all agents in the population, do
5:       count new solution of an agent using (2),
6:       update an agent best solution if needed,
7:       update the population best solution if needed,
8:    end for
9: until the maximal number of iterations is reached
10: End
11: Algorithm result: return the population best solution as an optimization result.
```

It is worth mentioning that the evolutionary algorithm for solving problems by searching needs to balance between the exploration and exploitation of the search space [7]. The former is the process of penetrating entirely new regions, while the latter addresses the local search close to the previously found solutions. The presented evolutionary approach incorporates the learning from the experience of each individual agent as well as from the experience of the population (see (2)). In this way, the exploitation process is being incorporated into the method. On the other hand, we will see below that the idea of the modified Best Cost Route Crossover Operator will lead us to the exploration procedure (Sec. 3.2).

2.3. Multi-objectivity and fitness function

In a proposed evolutionary algorithm, very often we have to decide if an actual solution is better than the best from an agent's historical results or the historically best from the population. For the VRPTW problem (which is a multi-objective optimization problem), we should take into consideration both the number of vehicles involved in the solution as well as the route length of all of these vehicles. Thus, we can distinguish between four different fitness criteria:

- Distance – sum of the distances travelled by the vehicles in the selected solution. The lower the value, the better the solution.
- Distance and Vehicle Count – we try to minimize both of these values, but we set a higher priority on distance.
- Vehicle Count and Distance – similar to the previous criterion, but distance has a lower priority.
- Weighted Method – we try to find the minimum of fitness function $F(s) = \alpha|v_s| + \beta|s|$, where $|v_s|$ denotes the vehicle count in solution s and $|s|$ is the distance travelled by the vehicles in s. Parameters α and β are usually established empirically, and the values are most-often set to: $\alpha = 100$ and $\beta = 0.001$ [19,21].

During our experiments, we decided to use the Weighted Method as the best suited model to represent needs of logistic companies. Analyzing the cost of such types of activities, one can stress that overall distance is connected to the costs of fuel, and the number of vehicles is connected to the costs of the workforce. The choice of the Weighted Method and parameters α and β express the idea that, for a logistic company, it is better (in economical aspects) to spend a little more on fuel than on an additional truck and driver. On the other hand, we should emphasize that the choice of a fitness policy has an important impact on the final results of the multi-objective decisions.

3. Crossover operators

In the literature, many crossover operators that can be used for combinatorial problems exist; some of them have already been used in VRP [22]. Unfortunately, most of these operators were designed for other combinatorial problems (like TSP), and when applied to VRPTW, one looses' main information about customers distribution among vehicles in chromosome. This is why such operators enforce the use of additional repair mechanisms in order to obtain a proper VRPTW solution. On the other hand, the Best Cost Route Crossover Operator was designed especially for VRP and does not require any additional repair procedures [21].

3.1. Best Cost Route Crossover Operator

The BCRC recombination operator can be defined as follows. Let us have the chromosomes of two parents $P1$ and $P2$; we can compute $P1 \otimes P2$ using algorithm 2. In order to obtain the final result, we should generate two offspring ($O12$ and $O21$), exchanging the role of parents $P1$ and $P2$ in algorithm 2 and choosing the best according to the chosen fitness function.

The presented procedure ensures that, if parents $P1$ and $P2$ respect the conditions on time and capacity, the obtained offspring also will respect these constraints. A simple example of such a procedure is presented below.

Example *BCRC operator* (see Fig. 2)

Assume that we have two parents of the following forms:

$$P1 = \{0, 2, 4, 5, 0, 1, 7, 0, 3, 6, 0\},$$
$$P2 = \{0, 1, 6, 0, 5, 7, 2, 0, 3, 4, 0\}.$$

Suppose that we randomly select vehicle $v = \{5, 7, 2\}$ from $P2$. At the beginning, we copy $P1$ into offspring $O12$ and erase from $O12$ all customers from selected vehicle v; at this point, the offspring would have the following form:

$$O12 = \{0, 4, 0, 1, 0, 3, 6, 0\}.$$

Now, we start the procedure of reinserting the missing customers serviced by vehicle v into offspring $O12$. We randomly select an element from v (let us assume that it is customer 5) then we find the best place in $O12$ where we can insert this customer.

In the example, the best choice is to insert customer 5 after customer 4 serviced by the second car. We have to repeat the same operation for the rest of the elements in v.

Algorithm 2 BCRC Crossover

```
1: Task: Create offspring chromosome O12.
2: Inputs: parent chromosomes P1,P2, fitness function fit(x).
3: Begin
4: O12 = P1
5: selectedVehicle = Rand(P2vehicles.size)
6: candidates = P2vehicles[selectedVehicle].customers
7: for all customer in candidates do
8:    Erase customer from O12
9:    Update cargo free space for all vehicles in O12
10: end for
11: for all customer in candidates do
12:    min = infinity
13:    placeid = −1
14:    for all place in places in O12 do
15:       TempO = O12
16:       insert customer in place to TempO
17:       if TempO respects constraints then
18:          actualLength = fit(TempO)
19:          if actualLength < min then
20:             placeid = place
21:             min = actualLength
22:          end if
23:       end if
24:    end for
25:    if placeid == −1 then
26:       add new vehicle with customer to O12
27:    else
28:       insert customer in placeid into O12
29:    end if
30: end for
31: End
32: Algorithm result: New offspring O12.
```

At the end, we obtain offspring of the following form (see Figure 2):

$$O12 = \{0, 4, 5, 0, 1, 7, 0, 2, 3, 6, 0\}$$

One can observe that most of the edges from parent $P1$ were copied into offspring $O12$ and none of edges from parent $P2$ were copied into offspring $O12$. A similar procedure is used to create a second offspring ($O21$). At the end, the best-suited offspring between $O12$ and $O21$ is used as a result of the presented recombination procedure. The best-worse relationship for two chromosomes is described in detail in Section 2.3.

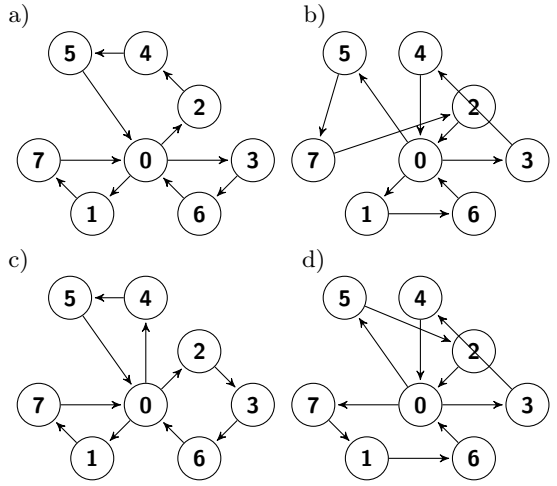

Figure 2. Graphical representation of BCRC operator example: a) parent $P1$; b) parent $P2$; c) offspring $O12$; d) offspring $O21$.

3.2. Modification of BCRC operator for VRPTW

The first numerical experiments of the proposed algorithm with the use of the BCRC crossover operator shows that the BCRC operator can be modified in order to obtain better results. The chromosome of the offspring obtained from the application of BCRC on the parent chromosomes shows a similarity to only one of the two parents. This behavior is natural if we analyze the structure of the BCRC operator; the offspring contains parts copied from only one of the parents, and the other parent only introduces some random mixing. This observation leads to a new Modified BCRC operator that incorporates parts of both of the parents into the offspring chromosome.

First, we introduce mean route length d_m for vehicle v as an overall length of the route travelled by a vehicle v divided by the number of visited customers. We denote the distance function between customers c_a and c_b as $dist(c_a, c_b)$. Assuming that vehicle v visits n customers in sequence $c_1, c_2, ..., c_n$, we can write the formula for mean route length as follows:

$$d_m(v) = \frac{dist(0, c_1) + \left[\sum_{i=1}^{n-1} dist(c_i, c_{i+1})\right] + dist(c_n, 0)}{n}. \tag{4}$$

Using the proposed mean route length, we can order all vehicles in a solution from the lowest to the highest values of d_m. Then, we denote the vehicle with the lowest mean route length as the best vehicle in the solution and the vehicle with the highest d_m as the worst vehicle in the solution.

The modified BCRC operator applied to parents $P1$ and $P2$ is presented in Algorithm 3. As in the case of BCRC, we should have to choose the best from offspring $O12$ and $O21$ obtained by using 3.

Algorithm 3 Modified BCRC Crossover

1: **Task:** Create offspring chromosome $O12$.
2: **Inputs:** parent chromosomes $P1$,$P2$, fitness function $\underline{fit}$(x).
3: **Begin**
4: **for all** vehicle in $P1vehicles, P2vehicles$ **do**
5: count value of mean length for vehicle
6: **end for**
7: identify the vehicles v_1^b, v_1^w, v_2^b and v_2^w
8: $O12 = P1$
9: $candidates = P1vehicles[v_1^w]$
10: $candidates.add(P2vehicles[v_2^b])$
11: $selectedVehicle = \text{Rand}(P2vehicles.size)$
12: $candidates.add(P2vehicles[selectedVehicle].customers)$
13: **for all** customer in candidates **do**
14: Erase customer from $O12$
15: Update cargo free space for all vehicles in $O12$
16: **if** customer in $P2vehicles[v_2^b]$ **then**
17: customers.erase(customer)
18: **end if**
19: **end for**
20: $O12.vehicles.add(P2vehicles[v_2^b])$
21: **for all** customer in candidates **do**
22: min = infinity
23: placeid = -1
24: **for all** place in places in $O12$ **do**
25: TempO = $O12$
26: insert customer in place to TempO
27: **if** TempO respects constraints **then**
28: actualLength = $\underline{fit}$(TempO)
29: **if** actualLength < min **then**
30: placeid = place
31: min = actualLength
32: **end if**
33: **end if**
34: **end for**
35: **if** placeid == -1 **then**
36: add new vehicle with customer to $O12$
37: **else**
38: insert customer in placeid into $O12$
39: **end if**
40: **end for**
41: **End**
42: **Algorithm result:** New offspring $O12$.

The proposed Modified BCRC operator incorporates elements of chromosomes from both parents and, at the same time, finds the best-possible places for some of the customers (line 21 in 3). An example of such a procedure is presented below.

Example *Modified BCRC operator* (see Fig. 3)

Taking into account the same parents as in the example for BCRC of forms:

$$P1 = \{0, 2, 4, 5, 0, 1, 7, 0, 3, 6, 0\},$$
$$P2 = \{0, 1, 6, 0, 5, 7, 2, 0, 3, 4, 0\}$$

we can identify the best and worst vehicles in both parents: $v_1^b = \{1, 7\}$, $v_1^w = \{2, 4, 5\}$, $v_1^b = \{1, 6\}$, $v_1^w = \{5, 7, 2\}$. At the beginning, we copy $P1$ into offspring $O12$ and exchange vehicle v_1^w with v_2^b; thus, it has the following form: $O12 = \{0, 1, 6, 0, 7, 0, 3, 0\}$. Suppose that we randomly select vehicle $v = \{3, 4\}$ from $P2$; in that case, we temporarily have $O12 = \{0, 1, 6, 0, 7, 0\}$.

Now, we start the procedure of reinserting missing customers $2, 3, 4, 5$ into offspring $O12$. Assuming that we look for the best place for all of the elements in random order $3, 4, 2, 5$, we obtain new solution $O12 = \{0, 1, 6, 3, 0, 2, 4, 7, 0\}$. A similar procedure is used to create the second offspring $O21$ (see Fig. 3).

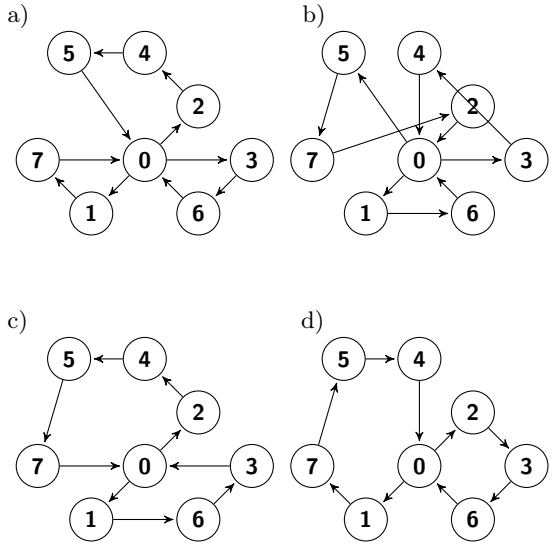

Figure 3. Graphical representation of modified BCRC operator example: a) parent $P1$; b) parent $P2$; c) offspring $O12$; d) offspring $O21$.

4. Experimental results

The presented evolutionary algorithm was implemented with each of the two crossover operators (BCRC and Modified BCRC). The implementation was applied to the known Solomon's and Ghering & Homberger's benchmarks with 50, 100, and 200 customers. During our experiments, we used a population of 30 agents. In each algorithm run, we set the maximal number of iterations to 2500. The results presented bellow

suggest that, the bigger the task considered, the more-efficient the results obtained from the Modified BCRC crossover operator are as compared to the original BCRC operator (see Tables 1, 2, and 3).

Table 1

Results for Solomon's VRPTW benchmark instances with 50 customers. Best means the best result obtained in 100 algorithm runs, Worst is the worst result. Results are presented in the form of #V/#D, where #V is the number of vehicles used and #D is the overall distance. The best-known results are taken from [11, 13, 15].

Problem	BCRC		Modified BCRC		Best known
	Best #V/#D	Worst #V/#D	Best #V/#D	Worst #V/#D	#V/#D
R101	12/ 1057.0	13/ 1109.8	12/ 1058.5	14/ 1111.4	12/ 1044 [13]
R201	4/ 817.7	6/ 873.5	4/ 817.7	7/ 883.0	6/ 791.9 [11]
R202	4/ 729.1	5/ 880.4	4/ 730.7	6/ 825.6	5/ 689.5 [11]
C101	5/ 363.2	6/ 403.6	5/ 363.2	6/ 406.8	5/ 362.5 [13]
C201	3/ 361.8	4/ 459.5	3/ 361.8	4/ 469.2	3 360.2 [15]
RC101	8/ 949.8	10/ 1067.4	8/ 949.8	10/ 1065.4	8/ 944 [13]

Table 2

Results for Solomon's VRPTW benchmark instances with 100 customers. Best means the best result obtained in 100 algorithm runs, Worst is the worst result. Results are presented in the form of #V/#D, where #V is the number of vehicles used and #D is the overall distance. The best-known results are taken from [24, 25, 28].

Problem	BCRC		Modified BCRC		Best known
	Best #V/#D	Worst #V/#D	Best #V/#D	Worst #V/#D	#V/#D
R101	20/ 1692.6	21/ 1758.2	19/ 1689.8	20/ 1775.8	19/ 1650.8 [24]
R102	18/ 1541.4	19/ 1581.5	17/ 1562.8	20/ 1600.7	17/ 1486.1 [24]
R105	15/ 1447.4	17/ 1582.5	15/ 1434.6	17/ 1642.7	14/ 1377.1 [25]
C101	10/ 828.9	11/ 872.4	10/ 828.9	11/ 869.3	10/ 828.9 [24]
RC101	15/ 1706.5	17/ 1811.2	15/ 1684.0	18 1894.4	14 1696.9 [28]

Table 3

Results for Ghering & Homberger's VRPTW benchmark instances with 200 customers. Best means the best result obtained in 100 algorithm runs for each class of instances. Results are presented in the form of #V/#D, where #V is the number of vehicles used and #D is the overall distance. The best-known results are taken from [26].

Problem	BCRC	MBCRC	Best known [26]
	#V/#D	#V/#D	#V/#D
C1	18.9/ 2722.06	18.9/ 2718.48	18.9/ 2718.41
C2	6.0/ 1850.07	6.0/ 1833.25	6.0/ 1831.59
R1	18.2/ 3642.22	18.2/ 3629.17	18.2/ 3611.86
R2	4.0/ 2932.18	4.0/ 2930.12	4.0/ 2929.41
RC1	18.0/ 3185.37	18.0/ 3179.20	18.0/ 3176.23
RC2	4.3/ 2539.52	4.3/ 2537.19	4.3/ 2535.88

The computation times for both of the methods are similar; thus, the better results obtained using Modified BCRC are not due to a longer computation time. In the case of RC101 with 100 customers, our result has a higher number of vehicles and shorter overall distance travelled. However, according to the economic decisions presented in Section 2.3 and the chosen fitness criteria, the best-known solution is still better than the one we obtained. In most of the cases, the number of vehicles for Modified BCRC is optimal; however, the overall distance is only slightly longer than for the best-known solutions.

Our best results are also very similar to the best-known solutions from the literature.In each case, the best-known solution and its reference is given.

5. Statistical comparison of obtained results

The distribution of results obtained in the numerical experiments suggest that the approach utilizing MBCRC gives better results than that with BCRC (Figs. 4–8). In order to prove this hypothesis, we can use the Mann-Whitney U test [18]. This nonparametric statistical test allows us to decide if two independent statistical samples come from the same population (have the same statistical distribution). The test defines a statistical number called U based on elements from both samples; the value of U describes a mixing of results in both samples (for details, see original paper [18], where the construction of U is defined explicitly). In many cases, the Mann-Whitney U test allows us to prove if one sample represents the population that has statistically larger values than the other.

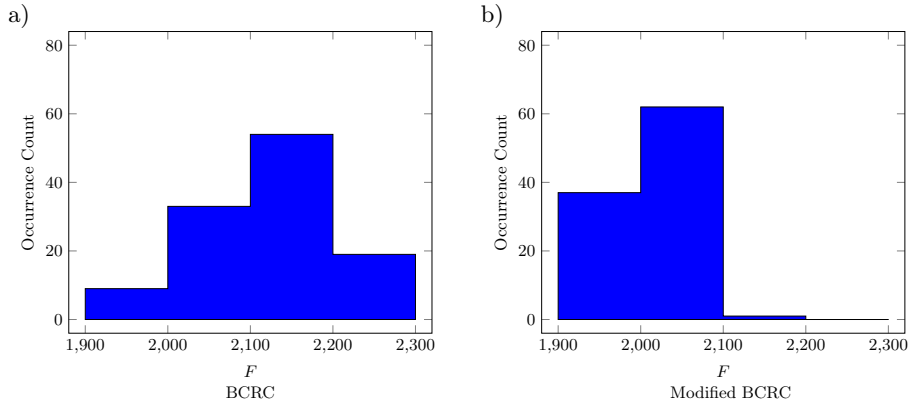

Figure 4. Histogram of distribution of the the weighted fitness function (F) values for instance R101 with 100 customers using BCRC (a) and Modified BCRC (b) operators. $F(p) = \alpha|v_p| + \beta|p|$, where $\alpha = 100$ and $\beta = 0.001$.

At first, we have to formulate our null hypothesis (`H0`) and appropriate alternative hypothesis (`H1`); then, we apply the Mann-Whitney U test in order to decide if we can accept hypothesis `H0`.

`Hypothesis H0` The distributions of the results for both crossover operators BCRC and MBCRC are statistically the same.

`Hypothesis H1` The distributions of the results for both crossover operators BCRC and MBCRC are statistically different.

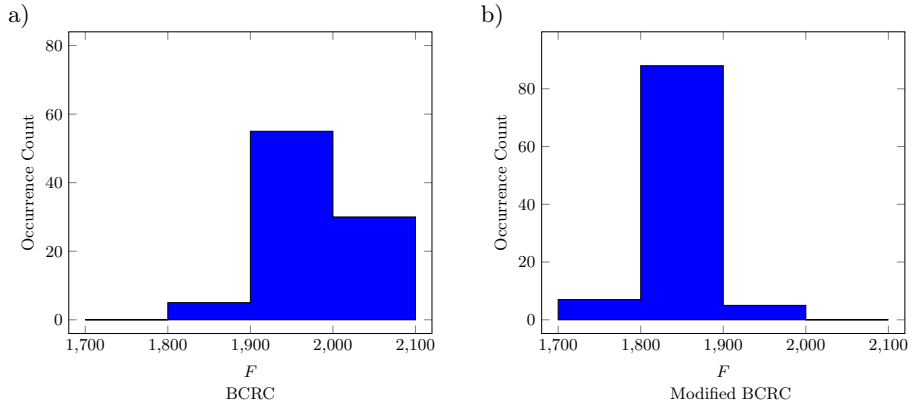

Figure 5. Histogram of distribution of the the weighted fitness function (F) values for instance R102 with 100 customers using BCRC (a) and Modified BCRC (b) operators. $F(p) = \alpha|v_p| + \beta|p|$, where $\alpha = 100$ and $\beta = 0.001$.

For each of obtained result distribution, we have calculated statistical value U and the appropriate z_U given by (5):

$$z_u = \frac{|U - (n_1 n_2/2)|}{\sqrt{\frac{n_1 n_2 (n_1+n_2+1)}{12}}}, \tag{5}$$

where n_1, n_2 – are sizes of statistical ensembles.

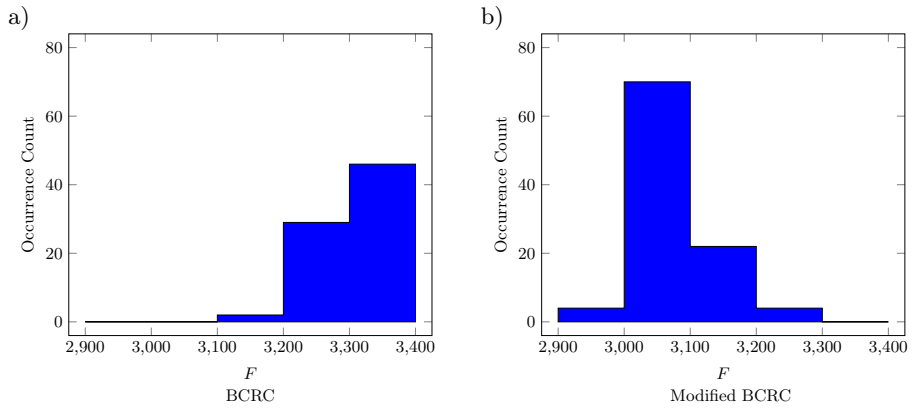

Figure 6. Histogram of distribution of the the weighted fitness function (F) values for instance R105 with 100 customers using BCRC (a) and Modified BCRC (b) operators. $F(p) = \alpha|v_p| + \beta|p|$, where $\alpha = 100$ and $\beta = 0.001$.

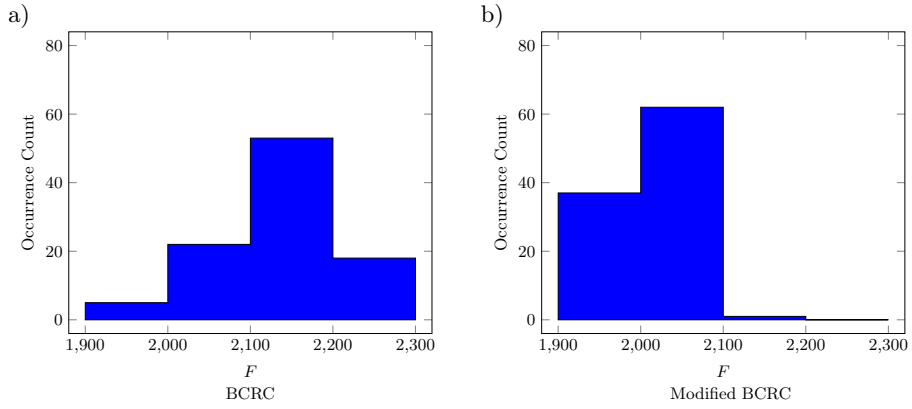

Figure 7. Histogram of distribution of the the weighted fitness function (F) values for instance RC101 with 100 customers using BCRC (a) and Modified BCRC (b) operators. $F(p) = \alpha|v_p| + \beta|p|$, where $\alpha = 100$ and $\beta = 0.001$.

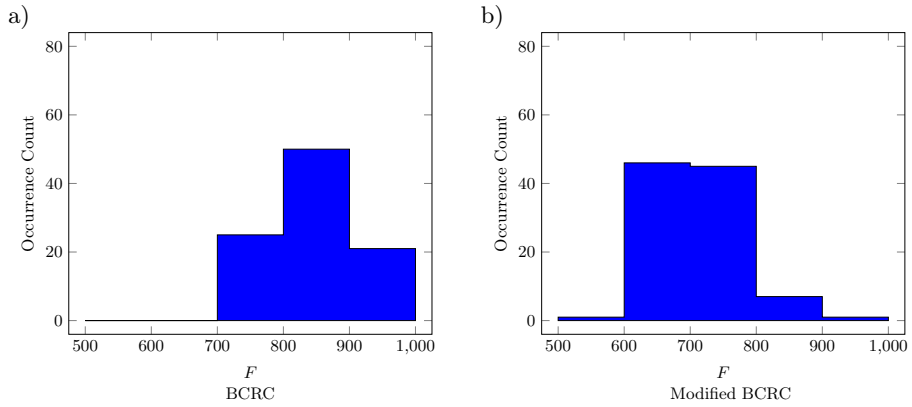

Figure 8. Histogram of distribution of the the weighted fitness function (F) for results obtained for instance RC201 with 100 customers using BCRC (a) and Modified BCRC (b) operators. $F(p) = \alpha|v_p| + \beta|p|$, where $\alpha = 100$ and $\beta = 0.001$.

Hypothesis H0 can be accepted with a significance set to 0.05 if $z_U < 2.575$. The results of our statistical tests for instances with 100 customers are presented in Table 4. We can conclude that, for all considered cases, Hypothesis H0 must be rejected.

Table 4

Results of Mann-Whitney U test calculated for distributions of obtained results for instances with 100 customers.

Problem	U	z_U
R101	963	9.86
R102	137	11.88
R105	43	12.11
RC101	968	9.85
RC201	250	11.60

6. Conclusions

In this paper, we present an evolutionary approach to the vehicle routing problem with time windows. This algorithm uses agents equipped with private and social memory. This is implemented for numerical experiments with two crossover operators: the Best Cost Route Crossover Operator, and its modified version described explicitly in Section 3.2. The original BCRC operator was specially derived for the VRPTW task; the introduction of the presented modification ensures that the parts of chromosomes from both parents can be found in an offspring in contrast to the case with original operator. The results show that, for the analyzed Solomon's and

Ghering & Homberger's benchmark tasks, the introduced Modified BCRC operator gives better results than the original one when used in the proposed algorithm for instances with 100 and 200 customers. No mutation operation is included in the presented algorithm due to the conclusion obtained during the preliminary numerical experiments. No significant contribution to algorithm efficiency was observed with mutation applied. Comparing the obtained results with the known optimal results and other presented in the literature, we conclude that the proposed method gives satisfying results statistically better than those obtained using the BCRC operator (Tabs. 3, 4). However, there is still room for improvement; the next modifications of the presented algorithm will be investigated in future works.

References

[1] Battarra M.: Exact and heuristic algorithms for routing problems, *4OR*, vol. 9(4), pp. 421–424, 2011. http://dx.doi.org/10.1007/s10288-010-0141-9.

[2] Berger J., Barkaoui M.: A parallel hybrid genetic algorithm for the vehicle routing problem with time windows, *Computers & Operations Research*, vol. 31(12), pp. 2037–2053, 2004.

[3] Braysy O., Gendreau M.: Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms, *Transportation Science*, vol. 39(1), pp. 104–118, 2005. http://dx.doi.org/10.1287/trsc.1030.0056.

[4] Braysy O., Gendreau M.: Vehicle Routing Problem with Time Windows, Part II: Metaheuristics, *Transportation Science*, vol. 39(1), pp. 119–139, 2005. http://dx.doi.org/10.1287/trsc.1030.0057.

[5] Cordeau J.F., Laporte G., Mercier A.: A unified tabu search heuristic for vehicle routing problems with time windows, *Journal of the Operational Research Society*, vol. 52(8), pp. 928–936, 2001.

[6] Dantzig G.B., Ramser J.H.: The Truck Dispatching Problem, *Management Science*, vol. 6(1), pp. 80–91, 1959.

[7] Eiben A.E., Schippers C.A.: On Evolutionary Exploration and Exploitation, *Fundamenta Informaticae*, vol. 35(1–4), pp. 35–50, 1998. http://dl.acm.org/citation.cfm?id=297119.297124.

[8] Fisher M.L.: Optimal Solution of Vehicle Routing Problems Using Minimum K-Trees, *Operations Research*, vol. 42(4), pp. 626–642, 1994. http://dx.doi.org/10.1287/opre.42.4.626.

[9] Geetha S., Poonthalir G., Vanathi P.T.: A Hybrid Particle Swarm Optimization with Genetic Operator for Vehicle Routing Problem, *Journal of Advances in Information Technology*, vol. 1(4), 2010.

[10] Gehring H., Homberger J.: A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. In: *Proceedings of EUROGEN99*, vol. 2, pp. 57–64, Springer, Berlin, 1999.

[11] Kallehauge B., Larsen J., Madsen O.B.: Lagrangian duality applied to the vehicle routing problem with time windows, *Computers & Operations Research*, vol. 33(5), pp. 1464–1487, 2006.

[12] Kohl N.: *Exact methods for time constrained routing and related scheduling problems*. Ph.D. thesis, Technical University of Denmark, 1995.

[13] Kohl N., Desrosiers J., Madsen O.B., Solomon M.M., Soumis F.: 2-path cuts for the vehicle routing problem with time windows, *Transportation Science*, vol. 33(1), pp. 101–116, 1999.

[14] Laporte G., Gendreau M., Potvin J.Y., Semet F.: Classical and modern heuristics for the vehicle routing problem, *International Transactions in Operational Research*, vol. 7(4-5), pp. 285–300, 2000. http://dx.doi.org/10.1111/j.1475-3995.2000.tb00200.x.

[15] Larsen J.: *Parallelization of the vehicle routing problem with time windows*. Ph.D. thesis, Technical University of Denmark, Danmarks Tekniske Universitet, Department of Informatics and Mathematical Modeling, Institut for Informatik og Matematisk Modellering, 1999.

[16] Lenstra J.K., Kan A.: Complexity of vehicle routing and scheduling problems, *Networks*, vol. 11(2), pp. 221–227, 1981.

[17] Liu B., Wang L., Jin Y.H.: An effective hybrid PSO-based algorithm for flow shop scheduling with limited buffers, *Computers & Operations Research*, vol. 35(9), pp. 2791–2806, 2008.

[18] Mann H.B., Whitney D.R.: On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other, *The Annals of Mathematical Statistics*, vol. 18(1), pp. 50–60, 1947. http://dx.doi.org/10.1214/aoms/1177730491.

[19] Minocha B., Tripathi S.: Two Phase Algorithm for Solving VRPTW Problem, *International Journal of Artificial Inteligence and Expert Systems*, vol. 4, 2013.

[20] Moccia L., Cordeau J.F., Laporte G.: An incremental tabu search heuristic for the generalized vehicle routing problem with time windows, *Journal of the Operational Research Society*, vol. 63(2), pp. 232–244, 2012.

[21] Ombuki B., Ross B.J., Hanshar F.: Multi-objective Genetic Algorithms for Vehicle Routing Problem with Time Windows, *Applied Intelligence*, vol. 24(1), pp. 17–30, 2006.

[22] Puljić K., Manger R.: Comparison of eight evolutionary crossover operators for the vehicle routing problem, *Mathematical Communications*, vol. 18(2), pp. 359–375, 2013.

[23] Reimann M., Doerner K., Hartl R.F.: D-ants: Savings based ants divide and conquer the vehicle routing problem, *Computers & Operations Research*, vol. 31(4), pp. 563–591, 2004.

[24] Rochat Y., Taillard É.D.: Probabilistic diversification and intensification in local search for vehicle routing, *Journal of Heuristics*, vol. 1(1), pp. 147–167, 1995.

[25] Shaw P.: Using constraint programming and local search methods to solve vehicle routing problems. In: *Principles and Practice of Constraint Programming – CP98*, pp. 417–431. Springer, 1998.

[26] SINTEF: top VRPTW web page. https://www.sintef.no/projectweb/top/vrptw/.

[27] Solomon M.: Solomon's VRPTW Benchmark Problems, 1999. http://w.cba.neu.edu/~msolomon/problems.htm.

[28] Taillard É., Badeau P., Gendreau M., Guertin F., Potvin J.Y.: A tabu search heuristic for the vehicle routing problem with soft time windows, *Transportation Science*, vol. 31(2), pp. 170–186, 1997.

[29] Thangiah S.R., Nygard K.E., Juell P.L.: GIDEON: a genetic algorithm system for vehicle routing with time windows. In: *Proceedings. The Seventh IEEE Conference on Artificial Intelligence Application*, vol. i, pp. 322–328, 1991. http://dx.doi.org/10.1109/CAIA.1991.120888.

[30] Toth P., Vigo D. (eds.): *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.

[31] Zhang Z., Qin H., Lim A., Guo S.: Branch and Bound Algorithm for a Single Vehicle Routing Problem with Toll-by-Weight Scheme. In: García-Pedrajas N., Herrera F., Fyfe C., Benítez J., Ali M. (eds.), *Trends in Applied Intelligent Systems*, *Lecture Notes in Computer Science*, vol. 6098, pp. 179–188, Springer, Berlin–Heidelberg, 2010. http://dx.doi.org/10.1007/978-3-642-13033-5_19.

Affiliations

Krzysztof Podlaski
University of Lodz, Faculty of Physics and Applied Informatics, Łodź, Poland,
podlaski@uni.lodz.pl

Grzegorz Wiatrowski
University of Lodz, Faculty of Physics and Applied Informatics, Łodź, Poland,
wiatr@uni.lodz.pl

Received: 2.12.2015
Revised: 16.05.2017
Accepted: 20.05.2017