DOI: 10.2298/CSIS100425067K

A Genetic Algorithm for the Routing and Carrier Selection Problem*

Jozef Kratica[1], Tijana Kostić[2], Dušan Tošić[3], Djordje Dugošija[3], and Vladimir Filipović[3]

[1] Mathematical Institute, Serbian Academy of Sciences and Arts,
Kneza Mihajla 36/III, 11 000 Belgrade, Serbia
jkratica@mi.sanu.ac.rs
[2] UCLA Department of Mathematics
Box 951555 Los Angeles, CA 90095-1555, USA
kostict@math.ucla.edu
[3] Faculty of Mathematics, University of Belgrade,
Studentski trg 16/IV, 11 000 Belgrade, Serbia
{dtosic | dugosija | vladaf}@matf.bg.ac.rs

Abstract. In this paper we present new evolutionary approach for solving the Routing and Carrier Selection Problem (RCSP). New encoding scheme is implemented with appropriate objective function. This approach in most cases keeps the feasibility of individuals by using specific representation and modified genetic operators. The numerical experiments were carried out on the standard data sets known from the literature and results were successful comparing to two other recent heuristic for solving RCSP.

Keywords: vehicle routing problems, genetic algorithm, evolutionary computation, combinatorial optimization.

1. Introduction

The delivery of goods from a warehouse to local customers is an important and practical problem of the logistics management. It has to decide which customers are to be served by the heterogeneous internal fleet and to route the vehicles of the internal fleet. A internal vehicle allows a company to consolidate several shipments, going to different destinations, in a single route. This problem with internal fleet and external carriers has also known as VRPPC (vehicle routing problem with private fleet and common carriers) in the literature ([4])

Demand of remaining customers must be served by external carriers. An external carrier usually assumes the responsibility for routing each shipment from origin to destination. The freight charged by a external carrier is usually much higher than the cost of a vehicle in internal fleet. Therefore, in some cases, it may be more economical to use external carriers instead of using an internal

* This research was partially supported by Serbian Ministry of Education and Science under the grant no. 174010.

vehicle to serve one or very few customers. Also, if total demand is greater than whole capacity of the heterogeneous internal fleet, logistics managers have to consider using an external carrier. Selecting the right mode to transport a shipment may yield significant cost savings to the company.

Routing and Carrier Selection Problem (RCSP) consists of the routing fixed number of vehicles with the limited capacity from the central warehouse to the customers with known demand. The objective is to minimize overall cost, which consists of the three parts: fixed cost of the using necessary vehicles in the internal fleet, variable transportation cost of routing every vehicle and a cost of freight for remaining customers (not served by internal fleet) charged by external carriers. RCSP is a NP-hard problem, as an generalization of the well known vehicle routing problem.

2. Related work

The routing problems are well-known combinatorial optimization problems and many approaches for solving these have been proposed. Good survey of new contributions can be found in [10] and for multi-objective case in [11].

However, only in several papers the problem with the vehicle routing when external carrier services are available, was considered. In [2] a fleet planning problem for long-haul deliveries with fixed delivery locations and an option to use an external carrier is considered. Static problem with a fixed fleet size and optional use of a outside carrier is considered in [1]. In [12] is described a methodology to address the fleet size planning and to route limited vehicles from a central warehouse to customers with random daily demands is developed. Paper [7] considered the problem where the company has only one vehicle.

In [5] selected customers were served by the external carriers, then used the modified version of heuristic proposed by [6] to construct the routes to serve the remaining customers and finally uses the local search (steepest descent heuristics) to improve the obtained solution.

The method proposed in [3], is named SRI (Selection, Routing and Improvement) heuristic, and it is composed of the following steps: (a) select customers to be served by the external carrier, (b) construct a first initial solution, (c) improve the obtained solution, (d) construct another initial solution, and (e) improve the second solution. Then, the best obtained solution is retained as a final solution. Use of two initial solutions increases chance to get good solutions within a very reasonable computation time.

Paper [4] describes metaheuirstic that uses perturbation procedure in the construction and improvement phases. It also performs exchanges between the sets of customers served by the private fleet and the common carrier. The obtained results clearly indicate that perturbation metaheuirstic is useful tool for solving RCSP.

A tabu search heuristic with a neighborhood structure based on ejection chains is described in [20]. In is empirically demonstrated that proposed algorithm slightly outperforms previous approaches reported in the literature.

3. Mathematical formulation

In our problem one central warehouse is considered. All vehicles start at the warehouse and return back to it. Also, the demands of all customers are known in advance and they cannot exceed the vehicle capacity. Each customer also has to be served by exactly one vehicle (either from internal fleet or by external carrier).

In the following an integer programming model given in [5] is presented:

$$\min \left(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{m} c_{ijk}x_{ijk} + \sum_{i=1}^{n} e_i z_i\right) \tag{1}$$

$$\sum_{k=1}^{m} y_{1k} = m \tag{2}$$

$$z_i + \sum_{k=1}^{m} y_{ik} = 1 \qquad i = 2, n; \tag{3}$$

$$\sum_{i=1}^{n} q_i \cdot y_{ik} \leq Q_k \qquad k = 1, m; \tag{4}$$

$$\sum_{j=1}^{n} x_{ijk} = y_{ik} \qquad i = 1, n; \;\; k = 1, m; \tag{5}$$

$$\sum_{j=1}^{n} x_{jik} = y_{ik} \qquad i = 1, n; \;\; k = 1, m; \tag{6}$$

$$\sum_{i\in S}\sum_{j\in S} x_{ijk} \leq |S| - 1 \qquad for\ all\ subsets\ S \subseteq \{2, 3, ..., n\}, \;\; k = 1, n; \tag{7}$$

$$x_{ijk} \in \{0, 1\}; \;\; y_{ik} \in \{0, 1\}; z_i \in \{0, 1\}; \quad i = 1, n; \;\; j = 1, n; \;\; k = 1, m; \tag{8}$$

In relations (1)-(8) the number of customers is denoted by n and number of vehicles by m. The coefficients f_k and Q_k are fixed cost and capacity of vehicle k ($k = 1, .., m$). q_i and e_i represent demand and cost charged by external carrier (if it is used) of customer i ($i = 1, .., n$). Transportation cost of truck k traveling from customer i to customer j is represented by c_{ijk}. Variable $z_i = 1$ if customer i is served by external carrier, and 0 otherwise. Similarly, $y_{ik} = 1$ if customer i is

served by vehicle k and $x_{ijk} = 1$ if vehicle k travels from customer i to customer j.

Constraint (2) ensures that all vehicles have been assigned to customers, (3) ensures that every customer is served by internal fleet or external carriers, (4) denotes vehicle capacity constraint, while (5) and (6) ensure that a vehicle arrives to a customer and also leaves that location. Subtour breaking constraints are given in (7).

There is another ILP formulation (with polynomial number of constraints) for RCSP, proposed in [4]. Moreover, we use ILP formulation only to explain problem, but without applying it in our algorithm.

4. Proposed genetic algorithm

Genetic algorithms (GAs) are robust stochastic search techniques which imitate some spontaneous optimization processes in the natural selection and reproduction. At each iteration (generation) GA manipulates a set (population) of encoded solutions (individuals), starting from either randomly or heuristically generated one. Individuals from the current population are evaluated using a fitness function to determine their qualities. Good individuals are selected to produce the new ones (offspring), applying operators inspired from nature (crossover and mutation), and they replace some of the individuals from the current population. Genetic algorithms are easily hybridized with other methods (for example see [16, 18]).

A global description of used genetic algorithm is given by pseudo-code in Figure 1.

$Npop$ denotes the number of individuals in a population and $val(i)$ is objective value of i-th individual.

Detailed description of GAs is out of this paper's scope and it can be found in [19]. Extensive computational experience on various optimization problems shows that GA often produces high quality solutions in a reasonable time. Some of recent applications are [8, 14, 17].

In this section we shall describe a GA implementation for solving the Routing and Carrier Selection Problem - RCSP. Computational results summarized in Tables 1 and 3 are very encouraging and give further justification of GA robustness.

4.1. Representation and objective function

The representation of individuals in this GA implementation is completely different from the other GA approaches for vehicle routing problems. In other GA methods, representation of particular gene include customer number in current problem instance. Customer's number usually is not an important value in the particular problem instance. For example, if customers are permuted in problem instance, solution value will remain the same.

```
Input_Data();
Population_Init();
while not Finish() do
  for i:=1 to Npop do
       if(Exist_in_Cache(i))
          then
               val(i) := Get_Value_From_Cache();
          else
               val(i) := Objective_Function(i);
        endif
        Update_Cache_Memory();
  endfor
  Fitness_Function();
  Selection();
  Crossover();
  Mutation();
endwhile
Output_Data();
```

Fig. 1. The basic scheme of our GA implementation

In order to use problem instance's characteristics in better way, in our GA implementation each gene represents relative distance between current customer and other unserved customers. This idea emerges from fact that in the optimal route of every vehicle consecutive customers are often close to each other. This means, if we increase chances of choosing relatively close customers in route of current vehicle, that will lead genetic algorithm toward promising regions of search space.

In the initial part of the GA, for every customer, we arrange other customers in the non-decreasing order of their distances (see Example 1). Genetic code of each individual consists of the $n-2$ genes, since the customer 1 is warehouse and it has not demand. The last customer is determined by all previous. Procedure starts from the warehouse and the first vehicle. We take the gene that corresponds to current vehicle and current customer from the genetic code. If that gene has value r, we take the $r+1$th closest unserved customer (warehouse is also not counted) from the matrix described above. If that customer has demand that can be served by current vehicle, we add it to the route of the current vehicle. If its demand exceeds capacity of current vehicle, the vehicle returns to warehouse. In that case, next vehicle is used, current customer is moved to warehouse and previous procedure is repeated. Due to the fact that load of vehicles is determined in sequential way, subsequent vehicles are empty. When there are no more vehicles available, demands of remaining customers are served by external carriers.

Example 1. Instance $chu1$ from [5] has n=6 customers and m=2 vehicles:

customers:					vehicles:			Customers arranged
i	xcoord	ycoord	q	e	k	Q	f	by distance:
1	35	35	0	0	1	40	60	1: 2 3 6 4 5
2	41	49	10	90	2	30	50	2: 4 1 5 6 3
3	35	17	7	108				3: 1 5 6 2 4
4	55	45	13	132				4: 2 1 5 3 6
5	55	20	19	150				5: 3 4 1 2 6
6	15	30	26	120				6: 1 3 2 5 4

Vehicles of the internal fleet have transportation costs $1.5/per mile (number of miles is rounded to integer). Suppose that the genetic code is 3 1 0 1. The first gene 3 means that the 4th closest (unserved) customer to open warehouse is chosen (customer 4). The second gene 1 is applied to customer 4 and its 2nd closest unserved customer is chosen (customer 5, since open warehouse is not count into the account). The third gene 0 denotes that the closest customer is chosen (customer 3). Since the 2nd closest unserved customer is 2 (1 and 5 are served) and its demand exceeds the capacity of vehicle 1 ($13+19+7+10 = 49 > 40$), vehicle must go to the warehouse (route of the 1st vehicle is: 1 4 5 3 1). We continue with the next (second) vehicle. The 2nd closest unserved customer of the warehouse is 6 (2 is closest, 3 is previously served). Since the demand of last customer 2 exceeds the capacity of vehicle 2 ($26 + 10 > 30$) and there are no more vehicles, customer 2 is served by an external carrier. Then, the route of vehicle 2 is: 1 6 1. This solution is optimal (see [3] or [5]) and has total cost of 387.5 .

4.2. Genetic operators

Our GA implementation experimented with tournament and fine-grained tournament selection - FGTS (described in [9]). The FGTS depends on the real parameter F_{tour} - the desired average tournament size that takes real values. Actually, the average tournament size should be as close as possible to F_{tour}. It is implemented using two types of tournaments. During one generation, tournaments are held with different number of competitors. The first tournament type is held k_1 times and its size is $\lfloor F_{tour} \rfloor$. The second type is performed k_2 times with $\lceil F_{tour} \rceil$ individuals participating ($\lfloor x \rfloor = r$ and $\lceil x \rceil$ = s $\Longleftrightarrow r \leq x \leq s$ and $r, s \in Z,\ x \in R$) that implies $F_{tour} \approx \frac{k_1 \cdot \lfloor F_{tour} \rfloor + k_2 \cdot \lceil F_{tour} \rceil}{k_1 + k_2}$.

The crossover operator is applied on a selected pair of parents producing two offspring. The one-point crossover is applied by randomly choosing crossover points and simply exchanging the segments of the parents' genetic codes. Crossover points are chosen on gene borders to prevent disruption of good genes. For example, if every gene has length 4, only possible crossover points are 4, 8, 12, etc.

The standard simple mutation operator is performed by changing a randomly selected gene in the genetic code of the individual, with a certain mutation rate. Since the number of genes in this GA implementation is $n-2$, the mutation rate

is $\frac{0.4}{n-2}$. Extensive computational experience on various optimization problems shows that this rate is appropriately chosen.

4.3. Caching GA

The running time of the GA is improved by caching (see [13]). The evaluated objective functions are stored in the hash-queue data structure, with the corresponding genetic codes. When the same genetic code is obtained again during the GA, the objective value is taken from the hash-queue table, instead of computing the objective function. The Least Recently Used (LRU) strategy is applied for caching GA. The number of cached function values is limited to 5000 in this implementation.

4.4. Other GA aspects

The population numbers 150 individuals and in the initial population (the first generation) is randomly generated. This approach provides maximal diversity of the genetic material and better gradient of objective function. A steady-state generation replacement with elitist strategy is used. In this replacement scheme only N_{nonel} = 50 individuals are replaced in every generation, while the best N_{elite} = 100 individuals are directly passed in the next generation preserving highly fitted genes. The elite individuals do not need recalculation of objective value since each of them is evaluated in one of the previous generations.

Duplicated individuals are removed from each generation. Their fitness values are set to zero, so that selection operator prevents them from entering into the next generation. This is very effective method for saving the diversity of genetic material and keeping the algorithm away from premature convergence. The individuals with the same objective function but different genetic codes in some cases may dominate the population. If their codes are similar, the GA can lead to local optimum. For that reason, it is useful to limit their appearance to some constant. In this GA application this constant is set to 40.

5. Computational results

In this section the computational results of the GA method and comparisons with existing algorithms are presented. All tests were carried out on an Intel 1.4 GHz with 256 MB memory. The algorithms were coded in C programming language. We tested our GA method on all RCSP instances proposed in [5] and [3].

The finishing criterion of GA is the maximal number of generations N_{gen} = 5000. The algorithm also stops if the best individual or best objective value remains unchanged through N_{rep} = 2000 successive generations. Since the results of GA is nondeterministic, method was applied 20 times on each problem instance.

The Table 1 summarizes the GA result on instances described above and is organized as follows:

- the first three columns contain the test instance name, the number of customers and vehicles respectively;
- the fourth column contains the optimal solution, named Opt for all solutions that are proved to be optimal. For instances chu5 and new5 optimal solution is not known because CPLEX in [3] was stopped after 150 hours. For this reason best known solution of CPLEX for these instances is noted with $*$.;
- the best solution obtained by GA in 20 runs, named GA_{best}, is given in fifth column;
- the average running time (t) used to reach the final GA solution for the first time is given in the sixth column, while the seventh and the eighth column (t_{tot} and gen) show the average total running time and the average number of generations for finishing GA, respectively;
- in the last two columns $eval$ represents the average number of the objective function evaluations, while $cache$ displays savings (in percent) achieved by using the caching technique.

Routes of internal vehicles and which customers are served by external carriers are presented in Table 2. The first column display instance's name, while the next two columns, presents information about optimal and GA solutions, respectively.

Next, in Table 3, we compare results of the GA method with Chu heuristic from [5] and SRI from [3]. The first two columns in Table 3 display instance's name and optimal solution. Next three columns contain results of GA: best GA solution in 20 runs, average time t in seconds needed to detect the best GA solutions and t_{tot} represents the total time (in seconds) needed to reach finishing criterion. Next two columns taken from [5] represent best solution and running time of the his heuristic. Since running times of SRI heuristic is not reported in [3], last column represents only solution values obtained by SRI heuristic.

Table 1. GA results

$Inst$	n	m	Opt	GA_{best}	t (sec)	t_{tot} (sec)	gen	$eval$	$cache$ (%)
chu1	6	2	387.5	opt	0.002	0.910	2001	3025	97.0
chu2	11	2	586.0	opt	0.075	1.357	2107	71695	32.1
chu3	16	3	823.5	opt	0.199	1.667	2261	85229	24.7
chu4	23	2	1389.0	1407.0	0.784	2.923	2742	125115	8.9
chu5	30	3	1441.5^*	1461.0	2.222	4.780	3556	167732	5.7
new1	6	2	423.5	opt	0.002	0.903	2005	2947	97.1
new2	11	2	476.5	opt	0.022	1.316	2032	65566	35.6
new3	16	3	777.0	opt	0.647	2.113	2838	105460	25.8
new4	23	2	1521.0	1545.0	0.449	2.603	2422	111690	8.0
new5	30	3	1609.5^*	b.k.	1.652	4.372	3189	149655	6.3

Optimal solutions in Tables 1 and 3 are noted by opt, instances that are not tested by $n.t.$ and best known solutions by $b.k.$ Although, Table 3 does not

Table 2. Routes of vehicles

Inst	*Optimal solution*	*GA solution*
Chu1	Route 1:1-3-5-4-1 Route 2:1-6-1 External carrier:2	Route 1:1-4-5-3-1 Route 2:1-6-1 External carrier:2
Chu2	Route 1:1-4-3-10-11-5-1 Route 2:1-2-9-8-7-1 External carrier:6	Route 1:1-4-3-10-11-5-1 Route 2: 1-7-8-9-2-1 External carrier:6
Chu3	Route 1:1-14-16-6-3-2-7-1 Route 2:1-8-12-15-9-1 Route 3:1-4-10-11-13-1 External carrier:5	Route 1: 1-7-2-3-6-16-14-1 Route 2: 1-8-12-15-9-1 Route 3: 1-13-11-10-4-1 External carrier:5
Chu4	Route 1:1-8-22-5-6-9-10-14-12-13-1 Route 2:1-7-2-3-4-17-16-15-18-23-21-20-19-1 External carrier:11	Route1:1-19-20-23-21-18-15-16-17-4-3-2-7-14-12-13-1 Route 2: 1-22-5-6-9-10-8-1 External carrier:11
Chu5	Route 1:1-21-23-3-6-5-4-20-1 Route 2:1-16-17-14-8-18-10-15-9-13-12-11-24-19-1 Route 3:1-27-28-29-30-26-25-2-7-1 External carrier:22	Route 1: 1-20-21-4-5-6-2-7-25-26-30-28-29-27-1 Route 2: 1-19-24-15-9-18-10-8-14-17-16-13-12-11-1 Route 3: 1-13-11-10-4-1 External carrier:22
New1	Route 1:1-6-4-1 Route 2:1-2-5-1 External carrier:3	Route 1: 1-6-4-1 Route 2: 1-2-5-1 External carrier:3
New2	Route 1:1-2-5-8-11-6-10-1 Route 2:1-7-4-9-1 External carrier:3	Route 1: 1-8-2-5-11-6-10-1 Route 2: 1-7-4-9-1 External carrier:3
New3	Route 1: 1-9-11-14-3-5-1 Route 2: 1-13-8-16-12-6-1 Route 3: 1-2-10-4-15-1 External carrier:7	Route 1: 1-5-3-14-11-9-1 Route 2: 1-6-12-16-8-13-1 Route 3: 1-2-10-4-15-1 External carrier:7
New4	Route 1: 1-16-2-9-7-21-18-5-8-1 Route 2: 1-23-6-20-19-4-22-10-13-3-12-15-14-17-1 External carrier:11	Route 1: 1-17-14-15-12-3-13-10-4-22-19-20-6-5-23-1 Route 2: 1-8-18-21-7-9-2-16-1 External carrier:11
New5	Route 1:1-23-26-4-22-16-19-5-7-9-24-14-11-1 Route 2: 1-18-12-29-2-17-28-1 Route 3: 1-25-10-8-15-30-21-27-13-20-6-1 External carrier:3	Route 1:1-23-26-4-22-16-19-5-7-9-24-14-11-1 Route 2: 1-25-10-8-15-30-21-27-13-20-6-1 Route 3: 1-1-18-12-29-2-17-28-1 External carrier:3

Table 3. GA compared with Chu and SRI heuristic

$Inst$	Opt	GA			Chu		SRI
		Sol	$t[s]$	$t_{tot}[s]$	Sol	$t[s]$	Sol
chu1	387.5	opt	0.002	0.910	opt	3.14	opt
chu2	586.0	opt	0.075	1.357	631	4.58	opt
chu3	823.5	**opt**	0.199	1.667	900.0	5.88	826.5
chu4	1389.0	1407.0	0.784	2.923	1681.5	8.42	**opt**
chu5	1441.5^*	1461.0	2.222	4.780	1917	11.06	**1444.5**
new1	423.5	opt	0.002	0.903	n.t.	n.t.	opt
new2	476.5	opt	0.022	1.316	n.t.	n.t.	opt
new3	777.0	**opt**	0.647	2.113	n.t.	n.t.	804.0
new4	1521.0	**1545.0**	0.449	2.603	n.t.	n.t.	1564.5
new5	1609.5^*	b.k.	1.652	4.372	n.t.	n.t.	b.k.

contain complete comparisons on all instances, because instances $new1$-$new5$ are proposed in [3], after publication of the paper [5]).

The data from Table 3 show that the GA method reached optimal solution (or best known solution for $chu5$ and $new5$ instances) in 7/10 cases, SRI in 6/10 cases and Chu heuristic only in 1/5 cases. For more clear comparison, solutions in Table 3, that are strictly better than solutions of other methods are bolded and underlined. We can see that GA is strictly better than other methods in 3 cases ($chu3$, $new3$, $new4$), SRI is strictly better than other methods in 2 cases ($chu4$, $chu5$), while Chu heuristic is never strictly better than other methods. From these results, it is quite obvious that GA and SRI outperform Chu heuristic, GA also obtain better solutions than SRI. It is obvious that GA produces high quality solutions in the reasonable time.

Our GA approach is also tested on large-scale instances from [4], both homogeneous and heterogeneous, and results are presented in Table 4 and Table 5. In that case our GA algorithm is hybridized with local search used in [20]. Local search is not applied on each individual since it is very time consuming. The strategy of applying local search from [15] is reapplied for this problem.

For all large-scale instances the optimal solution is not known, so the best solutions reported in the literature are used instead of optimal ones.

6. Conclusions

We present new heuristic, based on a genetic search framework, for solving the Routing and Carrier Selection Problem. Arranging unserved customers in non-decreasing order by their distances from current customer directs GA to promising search regions. Computational experiments on existing RCSP instances demonstrate the robustness of the proposed algorithms with the respect to the solution quality and running times. Comparisons with results from the literature show the appropriateness of proposed algorithm.

Table 4. GA results on large homogeneous RCSP instances

$Inst$	$Best$	GA		
		Sol	$t[s]$	$t_{tot}[s]$
CE-01	1119.47	1158.98	48.778	49.666
CE-02	1814.52	1893.66	90.213	91.234
CE-03	1921.1	1987.75	104.078	111.434
CE-04	2515.5	2668.87	191.656	204.088
CE-05	3097.99	3279.64	308.912	342.056
CE-06	1207.47	1233.20	46.605	47.316
CE-07	2004.53	2086.17	91.102	91.969
CE-08	2052.05	2130.82	107.776	113.816
CE-09	2429.19	2558.70	204.949	224.706
CE-10	3393.41	3598.36	314.969	358.665
CE-11	2330.94	2383.34	125.611	137.418
CE-12	1952.86	2042.84	105.744	111.045
CE-13	2858.94	2929.02	125.086	163.665
CE-14	2214.14	2338.22	101.846	103.830
G-01	14160.77	14910.52	479.625	629.328
G-02	19208.52	20258.91	1699.072	4568.416
G-03	24592.18	25941.17	7691.849	13529.265
G-04	34607.12	36083.77	9697.439	22360.709
G-05	14249.82	14875.44	1002.598	1803.100
G-06	21498.03	22440.03	3231.161	4826.779
G-07	23513.06	24621.42	6638.286	11098.164
G-08	30073.56	31326.38	7311.450	12532.019
G-09	1323.57	1368.47	2032.363	3236.873
G-10	1590.82	1646.20	5633.333	7682.187
G-11	2166.66	2235.24	9246.969	17381.100
G-12	2490.01	2578.12	18287.615	32100.689
G-13	2268.32	2347.49	772.484	1113.649
G-14	2693.35	2796.74	1487.419	2454.822
G-15	3157.31	3283.07	3264.514	5083.658
G-16	3637.52	3804.04	6351.355	11131.333
G-17	1631.49	1898.36	329.982	372.932
G-18	2691.61	3079.03	613.516	851.768
G-19	3452	3940.71	878.972	1110.866
G-20	4272.98	4823.76	1085.466	1606.512

Table 5. GA results on large heterogeneous RCSP instances

$Inst$	$Best$	GA		
		Sol	$t[s]$	$t_{tot}[s]$
CE-H-01	1191.7	1203.27	45.697	49.168
CE-H-02	1790.67	1860.84	92.357	94.371
CE-H-03	1917.96	1988.73	101.455	109.335
CE-H-04	2475.16	2622.24	191.363	206.061
CE-H-05	3143.01	3314.16	308.520	340.371
CE-H-06	1204.48	1210.75	50.987	51.531
CE-H-07	2025.98	2108.23	89.202	90.408
CE-H-08	1984.36	2057.75	109.998	119.157
CE-H-09	2438.73	2601.96	212.156	233.717
CE-H-10	3267.85	3415.40	326.473	382.491
CE-H-11	2303.13	2381.52	121.292	139.536
CE-H-12	1908.74	1954.80	104.721	109.677
CE-H-13	2842.18	2883.67	124.298	143.939
CE-H-14	1907.74	1988.79	104.464	111.559
G-H-01	14174.27	14812.40	460.866	661.852
G-H-02	18537.7	19395.20	2216.556	4757.013
G-H-03	25177.92	26523.43	5994.606	14040.338
G-H-04	34589.11	36261.53	12965.626	23744.678
G-H-05	15411.82	16254.20	572.518	889.646
G-H-06	19859.3	20717.86	2509.686	5350.663
G-H-07	23481.28	24727.21	5475.465	10955.261
G-H-08	27334.84	28605.47	13105.465	23568.062
G-H-09	1329.27	1386.03	1904.543	3259.721
G-H-10	1554.96	1622.14	4642.931	8750.368
G-H-11	2191.23	2266.04	6885.852	14759.343
G-H-12	2482.92	2580.32	14375.062	32527.158
G-H-13	2231.88	2330.81	839.558	1256.318
G-H-14	2682.85	2809.86	1043.280	2687.650
G-H-15	3123.6	3285.70	2379.560	5963.082
G-H-16	3621.85	3780.43	5750.238	10786.368
G-H-17	1664.08	1932.18	344.216	379.060
G-H-18	2708.73	3062.08	571.672	729.860
G-H-19	3443.59	3892.96	717.370	1004.026
G-H-20	4306.53	4865.32	930.398	1450.434

Our future research will be directed to parallelization of the presented GA, incorporation it in exact methods and its applying in solving similar routing problems.

References

1. Agarwal, Y.K.: Vehicle routing with limited fleet and common carrier option. TIMS/ORSA Joint National Meeting, Boston (1985)
2. Ball, M.O., Golden, A., Assad, A., Bodin, L.D.: Planning for truck fleet size in the presence of a common-carrier option. Decision Sciences 14, 103–120 (1983)
3. Bolduc, M.C., Renaud, J., Boctor, F.: A heuristic for the routing and carrier selection problem. European Journal of Operational Research 183, 926–932 (2007)
4. Bolduc, M., Renaud, J., Boctor, F., Laporte, G.: A perturbation metaheuristic for the vehicle routing problem with private fleet and common carriers. Journal of the Operations Research Society 59, 776–787 (2008)
5. Chu, C.: A heuristic algorithm for the truckload and less-than-truckload problem. European Journal of Operational Research 165, 657–667 (2005)
6. Clarke, G., Wright, J.W.: Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 12, 568–581 (1964)
7. Diaby, M., Ramesh, R.: The distribution problem with carrier service: a dual based penalty approach. ORSA Journal on Computing 7, 24–35 (1995)
8. Djurić, B., Kratica, J., Tošić, D., V., F.: Solving the maximally balanced connected partition problem in graphs by using genetic algorithm. Computing and Informatics 27(3), 341–354 (2008)
9. Filipović, V.: Fine-grained tournament selection operator in genetic algorithms. Computing and Informatics 22, 143–161 (2003)
10. Goel, A., Gruhn, V.: A general vehicle routing problem. European Journal of Operational Research 191(3), 650–660 (2008)
11. Jozefowiez, N., Semet, F., Talbi, E.G.: Multi-objective vehicle routing problems. European Journal of Operational Research 189(2), 293–309 (2008)
12. Klincewicz, J., Luss, H., M.G., P.: Fleet size planning when outside carrier service are available. Transportation Science 24, 169–182 (1990)
13. Kratica, J.: Improving performances of the genetic algorithm by caching. Computers and Artificial Intelligence 18, 271–283 (1999)
14. Kratica, J., Kovačević-Vujčić, V., Čangalović, M.: Computing strong metric dimension of some special classes of graphs by genetic algorithms. Yugoslav Journal of Operations Research 18(2), 143–151 (2008)
15. Kratica, J., Milanović, M., Stanimirović, Z., Tošić, D.: An evolutionary based approach for solving a capacitated hub location problem. Applied Soft Computing 11(1), 1858–1866 (2011)
16. Li, J., Pan, Q., Xie, S.: A hybrid variable neighborhood search algorithm for solving multi-objective flexible job shop problems. ComSIS 7(4), 907–930 (2010)
17. Liu, H.: Generative 3d images in a visual evolutionary computing system. ComSIS 7(1), 111–125 (2010)
18. Mao, Q., Zhan, Y.: A novel hierarchical speech emotion recognition method based on improved ddagsvm. ComSIS 7(1), 211–221 (2010)
19. Mitchell, M.: An introduction to genetic algorithms. MIT Press, Cambridge, Massachusetts (1999)

20. Potvin, J., Naud, M.: Tabu search with ejection chains for the vehicle routing problem with private fleet and common carrier. Journal of the Operations Research Society 62, 326–336 (2011)

Jozef Kratica was born in 1966 in Belgrade, Serbia. He received his B.Sc. degrees in mathematics (1988) and computer science (1988), M.Sc. in mathematics (1994) and Ph.D. in computer science (2000) from University of Belgrade, Faculty of Mathematics. In 2002 he joined Mathematical Institute as a researcher. As a delegation leader participated on the International Olympiads in Informatics (IOI'90 Minsk - Belarus, IOI'93 Mendoza - Argentina). His research interests include genetic algorithms (evolutionary computation), parallel and distributed computing and location problems.

Tijana Kostić was born in 1984 in Belgrade. She received bronze medal on the 41st International Mathematical Olympiad in Seoul 2000. She received her B.Sc. (2006) and M.Sc. (2007) degree in numerical mathematics from University of Belgrade, Faculty of Mathematics. Now, she is a Ph.D. student and Teaching/Research Assistant at UCLA, Department of Mathematics, USA. Her research interests include genetic algorithms and combinatorial optimization.

Dušan Tošić was born in 1949 in Knjaževac, Serbia. He received his B.Sc. degree in mathematics (1972), M.Sc. in mathematics (1977) and Ph.D. in mathematics (1984) from the University of Belgrade, Faculty of Mathematics. Since 1985 he has been professor of computer science at the Faculty of Mathematics. His research interests include parallel algorithms, optimization and evolutionary computation, numerical solving of the differential equations and teaching computer science.

Djordje Dugošija was born in 1947 in Sremska Mitrovica, Serbia. He received his B.Sc. degree in mathematics (1970), M.Sc. in mathematics (1974) and Ph.D. in mathematics (1986) from the University of Belgrade, Faculty of Mathematics. Since 1986 he has been professor of mathematical programming and discrete mathematics at the Faculty of Mathematics. He was the leader of the national team on many International Mathematical Olympiads. His research interests include operations research, combinatorial optimization and evolutionary computation.

Vladimir Filipović was born in 1968 in Podgorica, Montenegro. He received his B.Sc. degree (1993), M.Sc. (1998) and Ph.D. (2006) in computer science from University of Belgrade, Faculty of Mathematics. In 2006, he becomes assistant professor of computer science at the Faculty of Mathematics. His research interests include genetic algorithms, parallel algorithms and operational research.

Received: April 25, 2010; Accepted: October 21, 2011.