Modified ant colony optimization algorithm for solving the vehicle routing problem with a given load capacity

S N Medvedev and O A Medvedeva
Department of Applied Mathematics, Informatics and Mechanics, Voronezh State University, 1, University Square, Voronezh, 394018, Russia
E-mail: s_n_medvedev@mail.ru

Abstract. The article presents a mathematical model of the Vehicle routing problem with loading constraints in the form of a mathematical programming problem. A modification of the Ant colony optimization algorithm is presented to solve the problem. The main idea of the modification is to use the probabilistic rule of returning the agent (ant) to the depot before full unloading. The source of such behavior of the agent can be the concept of «fullness» of the ant. The computational experiment for comparing the classical and modified Ant colony optimization algorithms is presented. Testing was carried out according to two schemes: intensive testing and cross-testing. The conclusions about the efficiency of the proposed algorithm are presented.

1. Introduction
The Vehicle Routing Problem (VRP) is one of the key combinatorial optimization problems in transport and logistics. The goal of the VRP is to find a set of routes for the delivery of a certain resource to each point by agents (vehicles) with their return to the common starting location – the depot (center, base). Having a map of cities with depots and roads connecting them at the input, it is necessary to get a set of routes with a minimum total length at the output [1, 2]. The computational complexity of the algorithm depends on the size of the input data exponentially, i.e. the problem is NP-hard [3].

Figure 1 shows an example of the map of cities, depots, and roads, for which it is necessary to find a solution to a classic VRP. The map of delivery points (cities) is represented by a graph on which the numbered circles are delivery points, the lines connecting the circles are roads, and the vertical oval numbered with a zero is a depot. In general, there is a direct road from the depot to each of the delivery points, and every two delivery points are connected to each other in pairs. All distances are known. In this example, it is necessary to deliver resources from the depot to 18 delivery points, using 6 agents.

An example of a possible solution to this problem is shown in Figure 2. It can be seen how the set of available paths formed clusters – routes with a total minimum length, which are the solution to the VRP.
2. Problem statement
Let be given:

\[ V = \{v_0, v_1, ..., v_N\} \] – is the set of delivery points (cities), \( v_0 \) is the depot;

\[ A = \{(v_i, v_j); i, j = 1, N, i \neq j\} \] – is the set of roads;

\[ C = \{c_{ij}; i, j = 1, N, i \neq j\} \] – is the cost of moving from city \( i \) to city \( j \);

\[ D = \{d_i; i = 1, N\} \] – is the demand of the city \( i \);

\( W \) – is the load capacity of each agent;

\( K \) – is the number of agents.

It is necessary to find a set of routes that start and end at the depot, so that their total length is minimal.

To construct a mathematical model in the form of a mathematical programming problem, enter the following variables:
\[ x_{ij}^k = \begin{cases} 1, & \text{if the agent } k \text{ passed through the arc } (i,j), \\ 0, & \text{otherwise,} \end{cases} \]

\[ i = \overline{1,N}, j = \overline{1,N}, k = \overline{1,K}. \]

Consider the constraints of the problem.

1. The total length of the routes must be minimal

\[ S = \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{k=1}^{K} c_{ij} x_{ij}^k \rightarrow \min. \quad (1) \]

2. Each route starts and ends at the depot

\[ \sum_{j=1}^{N} \sum_{k=1}^{K} x_{0j}^k = K, \quad (2) \]

\[ \sum_{i=1}^{N} \sum_{k=1}^{K} x_{i0}^k = K. \quad (3) \]

3. Each vertex enters only one route and only one time

\[ \sum_{j=1}^{N} \sum_{k=1}^{K} x_{ij}^k = 1, \quad i = \overline{1,N}, \quad (4) \]

\[ \sum_{i=1}^{N} \sum_{k=1}^{K} x_{ij}^k = 1, \quad j = \overline{1,N}. \quad (5) \]

4. The total weight of the delivered goods for each route does not exceed \( W \)

\[ \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij}^k \cdot d_j \leq W, \quad k = \overline{1,K}. \quad (6) \]

5. Special condition for the absence of sub-cycles [4].

Thus, a mathematical model (1)–(6) of the VRP with limited load capacity is obtained. This problem is successfully solved by meta-heuristic algorithms, one of which is Ant colony optimization algorithm (ACO). ACO is very flexible, it can be easily expanded and modified.

3. Classification of solution algorithms

It is possible to use different approaches to solve the VRP with limited load capacity. By convention, all the used algorithms are divided into two classes: exact and approximate.

Exact algorithms produce an optimal solution for a given data set. Since the problem is NP-hard, there are no exact algorithms whose running time depends on the size of the input data in a polynomial way. Therefore, such algorithms are used for small input data volumes, or when the time required for the solution is not an important criterion.

Examples of exact algorithms are [5]:

1) Dynamic Programming methods.

In such methods, the general problem is broken down into simpler recursively related sub-problems, and each sub-problem is solved once.

2) The Branch and Bound Method.

The basic idea of this method is partition feasible set into subsets, each of which is checked and eliminated if it does not contain optimal solutions.

Approximate algorithms allow finding a suboptimal solution in polynomial time. If the input data is large enough, or computing resources are limited, these algorithms are used. Such algorithms do not guarantee optimality but give results that are accurate enough to solve the problem. Among the approximate algorithms it can be distinguished:

1) Heuristic algorithms [6].

Such algorithms implement some practical method, the «correctness» of which is not proved for all possible conditions, but showing results good enough to continue using it (ant colony optimization algorithm, genetic algorithm, simulated annealing algorithm).

2) Greedy algorithms [7].

When choosing a solution, preference is given to locally optimal options, on the assumption that the global solution will also be optimal.
3) Penalty Function Methods [8].

Some additional function is added to the objective function. This additional function «penalizes» non-optimal solutions by some rule.

Among the algorithms presented above, a heuristic algorithm simulating the behavior of an ant colony, the so-called Ant colony optimization, is of particular interest [9, 10].

The ACO has interesting and intuitive biological prerequisites. Its structure provides wide opportunities for modification and easy integration of testing and analysis systems. The algorithm allows monitoring the construction of the solution at every moment.

4. Modification of the ACO

The purpose and main idea of this study are to estimate the convergence rate of the algorithm (based on the average number of iterations) in the case when each ant (agent) has the opportunity to return to the depot at its own choice before full unloading. This decision is made by the agent probabilistically, depending on its current load. To do this, a new rule is added to the basic rules of the classical ACO [9, 10] – the probability of return based on the «fullness» of the ant. Thus, the modification uses the rules presented below.

The probability of selecting a city $j$ from the current city $i$:

$$ p_{ij} = \frac{(\text{pheromone}_{ij})^\alpha \cdot (\text{attractiveness}_{ij})^\beta}{\sum_{j \in J} (\text{pheromone}_{ij})^\alpha \cdot (\text{attractiveness}_{ij})^\beta} $$

where

$j$ – is the city from the number of available cities $J$;

$\text{pheromone}_{ij}$ – is the amount of pheromone on the road $(i, j)$;

$\text{attractiveness}_{ij} = \frac{1}{c_{ij}}$ – is the attractiveness of the road $(i, j)$, as the inverse of its length. That is, the longer the road, the less attractive it is;

$\alpha, \beta$ – are the configurable ACO parameters.

Pheromone Update Rule:

$$ \text{pheromone}_{ij} = (1 - \rho) \cdot \text{pheromone}_{ij} + \sum_{k \in K_{ij}} \frac{\theta}{L_k} $$

where

$\rho$ – is the pheromone evaporation coefficient;

$K_{ij}$ – is the set of ants that passed along the road $(i, j)$ on the iteration;

$\theta$ – is the value that characterizes the order of the optimal solution;

$L_k$ – is the total route length of the $k$-th ant.

The probability of returning to the depot at the current fullness is:

$$ p(\text{return}) = 1 - \frac{1}{1 + e^{-(\text{capacity} - a)}} $$

Here

$\text{capacity}$ – is the current fullness of the ant. The parameter not only describes the idea of modification, but also corresponds to common sense – the ants return to the anthill, not reaching hungry death;

$a$ – is the parameter responsible for the steepness of the curve and the displacement.

Figure 3 shows the graph of the function (9) at $a = 5$. 
5. Modified ACO algorithm

Algorithm 1.

Step 1. If the current system does not meet the exit conditions (limit on the maximum number of iterations or limit on the maximum number of iterations without improving the current solution, etc.), go to Step 2. Otherwise, the end of the algorithm.

Step 2. Create $K$ ants. Put the number of the current ant $k = 1$.

Step 3. Select the $k$-th ant. Put it in the depot and make up for fullness $\text{capacity} = D$.

Step 4. If there are unvisited cities, go to Step 5. Otherwise, go to Step 8.

Step 5. If the ant is still full with $\text{capacity} > 0$, go to Step 6. Otherwise, go to Step 3.

Step 6. Make a probabilistic decision to return to the depot or continue driving according to the formula (9). Go to Step 3, if the decision to return is made, otherwise to Step 7.

Step 7. For each of the unvisited cities, calculate the transition probability using the formula (7). Make a probabilistic transition to one of the available cities.

Step 8. Lower the $\text{capacity}$ (fullness) by the value of $d$ (the current city query). Go to Step 4.

Step 9. If necessary, update the best solution.

Step 10. If the current ant is not the last ($k < K$), take the next ant ($k = k + 1$), and go to Step 3. Otherwise, go to Step 11.

Step 11. For each road, update the pheromone concentration according to rule (8). Go to Step 1.

Thus, the proposed modification idea is implemented in steps 5 and 6. The remaining steps of the algorithm correspond to the classical ACO algorithm.

6. Test results

To evaluate the efficiency of both versions of the algorithm, we used given immutable data sets of different dimensions with known solutions for them and two testing schemes [11].

1. Intensive testing: for each data set, the solution of the classical and modified versions of the ACO is found 100 times. After that, the average deviation from the optimal solution, the average number of iterations required for convergence, and the frequency of obtaining optimal solutions are calculated.

2. Cross-testing: a series of 14 consecutive tests is performed; two tests for each data set, in which the problem is solved by classical and modified ACO, respectively. The purpose of this test is to
evaluate the efficiency of the algorithm in conditions close to real, in other words, what results the algorithm will show the first time.

The name of each of the sets [11] contains a description of the system: $E$ – is the source, means «Eilon and Christofides»; $n$ – is the dimension of the system; $k$ – is the minimum number of agents required for the convergence of the algorithm to the optimal value.

The results of the classical ACO are presented in Table 1, and the modified ACO – in Table 2.

### Table 1. Results of intensive testing of the classical ACO.

| Data set | Dimension (number of vertices) | Number of agents (ants) | Average deviation from the optimal solution | Average number of iterations |
|----------|--------------------------------|-------------------------|---------------------------------------------|----------------------------|
| $E$-$n22$-$k4$ | 22                            | 4                       | 3.7991                                      | 1141                       |
| $E$-$n33$-$k4$ | 33                            | 4                       | 42.1704                                     | 1130                       |
| $E$-$n51$-$k5$ | 51                            | 5                       | 83.3090                                     | 1171                       |
| $E$-$n76$-$k8$ | 76                            | 8                       | 106.6526                                    | 1139                       |
| $E$-$n76$-$k10$ | 76                            | 10                      | 106.7372                                    | 1163                       |
| $E$-$n101$-$k8$ | 101                           | 8                       | 136.7752                                    | 1248                       |
| $E$-$n101$-$k14$ | 101                           | 14                      | 133.4561                                    | 1189                       |

### Table 2. Results of intensive testing of the modified ACO.

| Data set | Dimension (number of vertices) | Number of agents (ants) | Average deviation from the optimal solution | Average number of iterations |
|----------|--------------------------------|-------------------------|---------------------------------------------|----------------------------|
| $E$-$n22$-$k4$ | 22                            | 4                       | 2.8551                                      | 1124                       |
| $E$-$n33$-$k4$ | 33                            | 4                       | 35.2679                                     | 1119                       |
| $E$-$n51$-$k5$ | 51                            | 5                       | 107.0561                                    | 1122                       |
| $E$-$n76$-$k8$ | 76                            | 8                       | 143.2744                                    | 1186                       |
| $E$-$n76$-$k10$ | 76                            | 10                      | 163.7372                                    | 1190                       |
| $E$-$n101$-$k8$ | 101                           | 8                       | 193.9881                                    | 1224                       |
| $E$-$n101$-$k14$ | 101                           | 14                      | 174.2317                                    | 1121                       |

We can see that for small dimensions, the modified ACO shows better results, both in accuracy and in the number of iterations. When increasing the input data, the classical ACO shows greater accuracy with a superior number of iterations.

The results of cross-testing are presented in Table 3. The algorithm is considered more efficient if it shows better parameters for the accuracy of the solution and the number of iterations, or if, with a small loss in accuracy, it finishes the work in a much smaller number of iterations. So, for example, on the third, fourth, and sixth data sets, the modified ACO slightly (in relation to the average deviation of intensive testing) was inferior to the classical ACO in accuracy, but it required a significantly smaller number of iterations.

### Table 3. Results of cross-testing.

| Dimension (number of vertices) | Number of agents (ants) | A more efficient algorithm | Closer to the optimal solution by | How many fewer iterations were required |
|--------------------------------|-------------------------|----------------------------|-----------------------------------|----------------------------------------|
| 22                             | 4                       | Modification              | 5.2204                            | 527                                    |
| 33                             | 4                       | Modification              | 11.1327                           | 6                                      |
| 51                             | 5                       | Modification              | -19.8774                          | 1340                                   |
| 76                             | 8                       | Modification              | -44.5441                          | 836                                    |
| 76                             | 10                      | Classical                 | 12.4472                           | 114                                    |
| 101                            | 8                       | Modification              | -33.0381                          | 887                                    |
| 101                            | 14                      | Classical                 | 10.0313                           | 323                                    |
In general, it can be argued that the algorithms show approximately the same results. Despite the fact that the modified ACO showed better results with small input data and was more often effective in cross-testing, these results, however, are not enough to give the modified ACO a preference over the classical version.

If we answer the practical question, namely, how would the overall operation of the system for the formation of shortest paths be affected by the possibility for the driver of the vehicle to return to the depot at his own choice before the full delivery of the resource, then we can assume that such behavior will not damage the efficiency of an irregular nature.

References
[1] Dantzig G B and Ramser J H 1959 Management Science 6 (1) 80–91
[2] Christofides N, Mingozzi A, Toth P and Sandi C 1979 Combinatorial Optimization (Chichester: Wiley)
[3] Toth P and Vigo D 2002 The Vehicle Routing Problem SIAM monographs on discrete mathematics and applications (Philadelphia: Society for Industrial and Applied Mathematics)
[4] Korbut A A and Finkelstein Yu Yu 1969 Discrete programming (Moscow: Nauka) (in Russ.)
[5] Gary M and Johnson D 1982 Computing machines and hard-to-solve problems (Moscow: Mir) (in Russ.)
[6] Boroznov V O 2008 Researched in Russia 11 322–328 (in Russ.)
[7] Kormen T, Leiserson Ch, Rivest R and Stein K 2005 Algorithms: construction and analysis (Moscow: Williams) (in Russ.)
[8] Eremin I I and Kostina M A 1967 Computational Mathematics and Mathematical Physics 7 (6) 1358–66 (in Russ.)
[9] Kazharov A A and Kureychik V M 2010 Journal of Computer and Systems Sciences International 1 32–45 (in Russ.)
[10] Dorigo M 1992 Optimization Learning and Natural Algorithms Ph.D Thesis Dip. Electronico (Milan: Politecnico di Milano) (in Italian)
[11] Vehicle Routing Data Sets: Instances of Eilon and Christofides URL: http://www.dca.fee.unicamp.br/projects/infobiosys/vrp/