# Ant Colony Optimization for Competitive Facility Location Problem with Elastic Demand

Tatyana Levanova and Alexander Gnusarev *

Sobolev Institute of Mathematics of SB RAS, Omsk Branch
13 Pevtsova Str., 644043, Omsk, Russia
levanova@ofim.oscsbras.ru,
alexander.gnussarev@gmail.com

**Abstract.** This work is devoted to development of swarm intelligence for competitive facility location problem with elastic demand in the following formulation. In a competitive environment, Company plans to locate new facilities which differ in design. Clients of each point choose the facilities of Company or Competitor depending on their attractiveness and distance. The total share of demand of facilities varies flexibly depending on the behaviour of clients. The Company's goal is to maximize the fraction of demand it serves. The modelling of this demand leads to the nonlinearity of the mathematical model and to the difficulties of finding the optimal solution by commercial software in an acceptable time. There is a small number of publications devoted to the problems with elastic demand, and a set of methods for solving the problem is limited. In this paper, we develop ant colony optimization approach. A comprehensive computational experiment is carried out, and the results are discussed.

**Keywords:** Swarm intelligence · Ant colony optimization · Discrete optimization · Location problem · Elastic demand

## 1 Introduction

During last decade the main attention, among all the branches of artificial intelligence, has been paid to researching multiagent systems. Those systems consist of a set of intercommunicating agents. The agents are relatively simple, but via cooperation they model so called swarm intelligence [22]. There are several examples of such intelligence in nature, such as: an ant colony, a bee hive, a flock of birds, a fish shoal, etc. Ant colony algorithms were developed by analogy with them. This work is devoted to the development of those algorithms.

* Section 4 of this research was supported by the Russian Foundation for Basic Research, grant 16-01-0740.


In: S. Belim et al. (eds.): OPTA-SCL 2018, Omsk, Russia, published at http://ceur-ws.org

At the base of ant colony algorithms (AC) is the idea of live ant swarm intelligence. Biological researches showed that ants are able to find the shortest path from the anthill to the food source using special essence, pheromone [4]. While moving, ants spread the track of odorous substance. The pheromone cumulates till new amounts of ants follow the track, and dissolves if it does not attract any more attention. The higher the pheromone level is the more ants choose the path. The binary bridge experiment was carried out. During the experiment, the anthill and the source of food were separated by a bridge with two branches of different length. With the course of time almost all the ants chose the shortest branch, which can be explained by the factthat during the same period of time the ants could pass through the shortest path more times than through the longest one. The pheromone level on the shortest path gradually increased and the bigger amount of ants followed the track.

Ants behavior while choosing the shortest path can be considered as an optimal solution search prototype. Artificial ant (AA) is a agent; ant colony is a multiagent system. In most cases, AA is a relatively simple probability algorithm, which occurs to be a part of a more complicated ant colony algorithm. In the process of solution constructing agents accumulate information about the task. That information serves like pheromone and is AC algorithm parameter. It is processed in AC algorithm and allows artificial ants cooperate during further research. The algorithm is completed in case several conditions are fulfilled, for instance: the given number of iterations, computing time and others.

Ant colony algorithm was proposed by Dorigo M., Maniezzo V., Colorny A. in 1991 and was given the name of Ant System. For the first time it was used to solve travelling salesman problem. The problem allows drawing an analog between the travelling salesmans motion along the edges of a graph and the ants movement from one point to another. Travelling salesman problem can be formulated the following way: there is a complete directed graph $G = (N, E)$, where the vertex set $N$, $|N| = n$ is the amount of cities, $E$ is the set of edges which describe the connections among the cities. The edge length $d_{ij}$ is equal to the distance between the cities $i$ and $j$; $i, j \in N, i \neq j$. Hamiltonian cycle of the minimal length is to be searched.

Ant colony algorithm belongs to so called metaheuristic algorithms, which general schemes can be applied to a wide range of problems [9]. Let us write out the AC algorithm general scheme.

AC algorithm scheme

Set initial parameters.
*Algorithm step:*

– Solution constructing via the artificial ant algorithm;

– Applying the local search procedure to the constructed solutions;

– Updating the statistical information for the following artificial ant algorithm usage.

According to [5], at each iteration in the set above travelling salesman problem the given set of artificial ants $L$ are constructing the solution moving along the graph from one city to another according to some probabilistic rule. If ant chooses to go from vertex $i$ to vertex $j$, then edge $(i, j)$ is added to the solution. Artificial ant repeats the iterations until the Hamiltonian cycle is constructed. Parameter $\tau_{ij}$ shows the desirability of the edge $(i, j)$ to appear in the solution and serves like an analog to the pheromone. The attractiveness of $\eta_{ij} = 1/d_{ij}$, which was called the visibility of city $j$ from city $i$ in [5], also influences the ants choice. The probability of the $l$-ant from the $L$-colony of artificial ants to go from city $i$ to city $j$ at iteration $t$ of the AC algorithm is calculated the following way:

$$p_{ij}^l(t) = \begin{cases} \frac{(\tau_{ij}(t))^\alpha (\eta_{ij})^\beta}{\sum_{k \in N^l} (\tau_{ik}(t))^\alpha (\eta_{il})^\beta}, & \text{if } \ j \in N^l, \\ 0, & \text{else,} \end{cases}$$

Where $N^l$ is the vertex set which ant $l$ has not visited yet, the values $\alpha$ and $\beta$ are the control parameters of the algorithm.

The update of the pheromone takes place at the end of each iteration $t$ of the ant colony algorithm with the ant-cycle scheme [7], when all the ants complete constructing solutions, according to the following formula:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t),$$

where $\rho \in (0, 1]$ is so called pheromone evaporation coefficient, $\Delta\tau_{ij}(t)$ is the total change of the pheromone on the edge $(i, j)$, that is

$$\Delta\tau_{ij}(t) = \sum_{l=1}^{L} \Delta\tau_{ij}^l(t).$$

Where $\Delta\tau_{ij}^l(t)$ is the impact of ant $l$ into the pheromone level on the edge $(i, j)$, which is calculated through the following formula:

$$\Delta\tau_{ij}^l(t) = \begin{cases} Q/P^l(t), & \text{if } (i,j) \in T^l(t); \\ 0, & \text{else.} \end{cases}$$

In the given formula $T^l(t)$ is the Hamiltonian cycle constructed by ant $l$ at iteration $t$, the variable $P^l(t)$ is its length, $Q$ is a positive constant.

Thereafter the scheme was enhanced: different pheromone update rules were researched [2, 5, 7]; only the global best current solution was considered for the pheromone update [6]; a group of elite ants was separated out (the best ants according to the objective function value) [13]; the local search was used in order to increase the quality of the solution [17]; new ideas were performed in order to prevent stagnation [20]; when the concurrent computing had been developed, parallelization of the process was used [19] and other ideas.

## 2 Competitive Facility Location Problem with Elastic Demand

This work is devoted to the development of swarm intelligence for the competitive facility location problem with elastic demand in the following formulation: Company plans to locate some new different by design facilities in a competitive environment. Clients at each point choose the facilities of Company or Competitor depending on the attractiveness and distance. The total share of the facilities demand varies flexibly depending on the clients behaviour. The Company's goal is to serve the largest share of the total demand. Modelling of this demand leads to the mathematical model nonlinearity and to the difficulties in finding the optimal solution by commercial software in acceptable time.

This problem was firstly formulated and modelled by Aboolian R., Berman O. and Krass D. [14]. Let us set the following mathematical model. In the given problem $N = \{1, 2, \ldots, n\}$ with weight $w_i, i \in N$ is a set of customers. Let $R$ be the set of facility designs which differ from one another in size, range, etc., $r \in R$. It is assumed that Competitor has already placed his facilities in $C \subset N$ and is not going to change them. Company may open its markets in $S = N \setminus C$ taking into account the budget $B$ and the opening cost $c_{jr}$ of the facility $j \in S$ with the design $r \in R$. Clients decide to choose the Companies' or the Competitors' facilities depending on the attractiveness $a_{jr}$ and the distance $d_{ij},\ i, j \in N, r \in R$. The Company's goal is to maximize the fraction of the demand it serves.

Let us introduce the variables: $x_{jr} = 1$, if the facility $j$ is opened with the design variant $r$, otherwise $x_{jr} = 0$; $j \in S; r \in R$. Thus the mathematical model can be presented the following way:

$$\max \sum_{i \in N} w_i \cdot g(U_i) \cdot MS_i, \tag{1}$$

$$\sum_{j \in S} \sum_{r \in R} c_{jr} x_{jr} \leq B, \tag{2}$$

$$\sum_{r \in R} x_{jr} \leq 1, j \in S, \tag{3}$$

$$x_{jr} \in \{0, 1\}, \quad r \in R, j \in S. \tag{4}$$

The demand function of this model has an exponential form: $g(U_i) = 1 - \exp(-\lambda_i \cdot U_i)$, where $\lambda_i$ is the characteristic of the elastic demand in point $i$, $\lambda_i > 0$; $U_i$ is the total utility for a customer at $i \in N$ from all open facilities:

$$U_i = \sum_{j \in S} \sum_{r \in R} k_{ijr} x_{jr} + \sum_{j \in C} \sum_{r \in R} k_{ijr} x_{jr}.$$

Coefficients $k_{ijr} = a_{jr}(d_{ij} + 1)^{-\beta}$ depend on the customers sensitivity $\beta$ to the distance from the facility and the attractiveness $a_{jr}$. The Company's total share of the facility $i \in N$ is measured by:

$$MS_i = \frac{\sum_{j\in S}\sum_{r\in R} k_{ijr}x_{jr}}{\sum_{j\in S}\sum_{r\in R} k_{ijr}x_{jr} + \sum_{j\in C}\sum_{r\in R} k_{ijr}x_{jr}}.$$

Based on the above notation, the objective function (1) looks as follows:

$$\max \sum_{i\in N} w_i \cdot \left(1 - \exp\left(-\lambda_i\left(\sum_{j\in S}\sum_{r\in R} k_{ijr}x_{jr} + \sum_{j\in C}\sum_{r\in R} k_{ijr}x_{jr}\right)\right)\right)\cdot \qquad (5)$$

$$\cdot\left(\frac{\sum_{j\in S}\sum_{r\in R} k_{ijr}x_{jr}}{\sum_{j\in S}\sum_{r\ R} k_{ijr}x_{jr} + \sum_{j\in C}\sum_{r\ R} k_{ijr}x_{jr}}\right).$$

The objective function (5) reflects the Company's goal to maximize the customers demand share. Inequality (2) takes the available budget into account. Condition (3) shows that only one variant of the design can be selected for each facility.

## 3 Development of the Ant Colony algorithms

Successful supplement of the ant colony algorithm for the travelling salesman problem led to the usage of this approach to a wide range of combinatorial problems. For instance, there are investigations for the Set Covering Problem [1]; Quadratic Assignment Problem [8]; Scheduling theory [3]; graph theory [12]; routing problem [21]; clustering problem [11]; Data mining [10] and many others. The development of this approach for the Discrete Optimization Problems has been investigated for several years in Sobolev Institute of Mathematics SB RAS (e.g. [13, 17]). The ant colony algorithm varieties were developed for the classical $p$-median minimization problem in [18]. The final set of possible facilities locations and the list of customers are already given in the problem. The facilities are able to produce the unlimited quantity of the similar product. The service cost for each customer are known; the opening expenses for each facility are equal to zero. The task is to locate $p$ facilities and attach the customers to them, so that each clients demand is satisfied and the total service charges are minimal. While constructing the ant colony algorithm for the stated above problem, the artificial ant marks the points $i$ from the set of possible facilities locations $I$ with the pheromone, unlike in [5], where the edges are marked. That is why the information about the solution is kept not in a matrix but in a vector. Each location vector component $z_i$ corresponds to the pheromone parameter $\alpha_i$. In order to define the pheromone value only the $t$ best solutions out of $L$ solutions are used at each iteration. Let us set $f$ as a value of the objective function; $F$ is the best known value of the objective function; $\alpha_{min}$ is the minimal possible pheromone level. The ant colony algorithm for the $p$-median problem looks as follows:

Scheme of AC algorithm for the $p$-median problem (ACPM)

(1) Set $\alpha := (\alpha_i), F := \infty, k := 1$.

Repeat the following until the stopping condition is met:

*Iteation* $k, k \geq 1$. (1) Construct $L$ possible solutions for the AA algorithm.

(2) Choose $t$ best solutions among them, according to the objective function value; $f^*$ is the best value at the given iteration.

(3) Change the parameters $\alpha_i$ with regard to $t$.

(4) If $f^* < F$, then for the non-zero vector components of the vector $z^*$, of the corresponding $f^*$, set $\alpha := \alpha_{min}$, $F := f^*$.

*Go to the next iteration* $k := k + 1$.

The vector components $\alpha_i$ are changed according to the following rule:

$$\alpha_i = \frac{\alpha_{min} + q^{\gamma_i}(\alpha_i - \alpha_{min})}{\rho}, i \in I,$$

$\rho \in (0, 1)$ is the pheromone parameter change coefficient, $\gamma_i \in [0, 1]$ is the frequency of the point $i$ appearance among the best solutions $t$; $q \in (0, 1)$ is the algorithm parameter.

In the AC algorithm version [18] the artificial ant is represented as a probabilistic variation of the greedy descent algorithm, where with the probability of

$$p_i = \frac{\alpha_i(\Delta f_{max} - \Delta f_i + \varepsilon)}{\sum_{k \in F(\lambda)} \alpha_k(\Delta f_{max} - \Delta f_i + \varepsilon)}$$

one facility is chosen out of the set of facilities $F(\lambda)$ included in the solutions so that the objective functions differ from $F$ not more than $\lambda$ times; $\Delta f_i$ is the objective function change as a result of facility closing at point $i$, $\Delta f_{max}$ is the maximum change; $\varepsilon > 0$ is the algorithm parameter.

The algorithm for the competitive facility location and design problem with elastic demand has been developed in this work based on the ant colony algorithm (ACPM) for the $p$-median problem (5), (2)-(4). The value of the pheromone vector component is changed by the following rule:

$$\alpha_i = \frac{\alpha_{min} + q^{(1-\gamma_i)}(\alpha_i - \alpha_{min})}{\rho}, i \in I,$$

An artificial ant in the ant colony algorithm version, offered in this work, is a probabilistic modification of a greedy ascent algorithm. The ant starts from the zero solution $z = 0$. Running through the possible facility locations, the ant either includes the location into the search list or not with the probability:

$$p_i = \frac{\alpha_i(\Delta f_i + \varepsilon)}{\sum_{k \in S} \alpha_k(\Delta f_i + \varepsilon)}.$$

Therefore, the list of the facility numbers to be searched is formed. After that a unit is taken out of the budget and the facility, which improves the objective function the most, is chosen. The algorithm works until the budget is not over. As a result, the artificial ant finds some solution. The local ascend along the Lin-Kernighan neighborhood with the length 3 is applied to the solution, which the ant brought. The Lin-Kernighan neighborhood permutes the existing facilities into the closed, i.e. an open facility becomes a closed one, but the closed facility is opened according to the first design variant.

## 4 Computatinal Experements

It must be mentioned, that because of a set of several parameters in the algorithm, there is problem of their setting, i.e. the parameter values for the best algorithm behavior for most instances should be chosen. One of the traditional ant colony algorithm parameters is the pheromone evaporation coefficient $\rho$, this one value equal to 0.95. The following values are chosen for the rest of the parameters: the number of ants at each iteration is $s = 30$; the minimal pheromone level $\alpha_{min}$ and the initial pheromone level $\alpha_0$ are equal to 0.3; $q = 0.5$ respectively. The maximal number of iterations is equal to $T_{max} = 5$.

The computational experiment was held for two series of test instances: in the first series (Series 1) the distances among the points were set with uniform distribution of distances in the interval [0;30]; in the second series (Series 2) the distances satisfied the triangle inequality. A set of 16 instances of the dimension $|N| = 60, 80, 100, 150, 200, 300$ with three possible projects and budget limits of 3, 5, 7 and 9 units were formed for each series. The distance sensitivity parameter is $\beta = 2$, the elastic demand parameter is $\lambda = 1$. The results of the algorithm execution were compared with the upper bounds which was described in [16] earlier.

The average deviations of the ant colony algorithm for the series of the test cases are given in tables 1 and 2. It could be seen that the average deviation of the ant colony algorithm for the first series with the dimension 300 is 1.71%, the maximal deviation does not exceed 5.595%. The minimal deviation is less then 0,001%. This is a very good result for problems of such dimension. For the second series we see a different situation. Even the minimum deviation from the upper bound is 11.116%. Here the average algorithm deviation is 14.821%, the maximal deviation is 19.232%. This is not a failure and can be explained by different reasons. In many cases, for other tasks it is possible to compare the obtained solution with the optimal solution or with the best known solution. For our problem it is often impossible to find even a feasible solution. Therefore, we have to use the upper bounds. In the rule for constructing the upper bound we used, there is a configurable parameter $m$ (10). It is possible to apply another rule to calculate it. In addition, large deviations may indicate an inaccuracy of the upper bound for those cases. We have studies that have shown this fact for other values of $\lambda$ [15].

The computational experiment was carried out on a computer with CPU Intel Xeon X5675 @ 3.07 GHz, 32 GB RAM. Information on the CPU time is given in Table 3 and in Figure 1. The CPU time for both series was much the

**Table 1.** Deviations from upper bounds in case of uniform distribution distances (Series 1) %

| | 60 | 80 | 100 | 150 | 200 | 300 |
|---|---|---|---|---|---|---|
| min | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| aver | 0.586 | 0.300 | 0.766 | 1.040 | 0.902 | 1.710 |
| max | 1.640 | 2.078 | 3.701 | 4.530 | 4.403 | 5.595 |

**Table 2.** Deviations from upper bounds in case of uniform distribution distances (Series 2), %

| | 60 | 80 | 100 | 150 | 200 | 300 |
|---|---|---|---|---|---|---|
| min | 13.090 | 12.847 | 9.727 | 13.689 | 12.025 | 11.116 |
| aver | 21.317 | 18.624 | 15.344 | 18.163 | 14.425 | 14.821 |
| max | 37.678 | 29.679 | 24.250 | 22.896 | 18.891 | 19.232 |

same, which is why in Table 3 there is the execution time for the first series only. In Figure 1 it is possible to observe a significant advantage of ant colony algorithm on sistem GAMS (solver CoinBonmin). The results of the CoinBonmin are given in brackets. The proposed algorithm works faster than the known software up to 13 times. For instance, the average execution time for SA for the dimension 300 was 1021 sec. and 11831 for CoinBonmin. Also usually, the ant colony algorithm has the advantage over a standard multistart procedure. Often it is faster than other algorithms for various difficult problems (for example, [13, 18]). The results of the current experiments indicate the necessity of its further development. Perhaps we get such values of CPU time (Table 3) because the facility number list to be search, formed by the ant, was excessively large. Now we are researching the algorithm behaviour for the smaller dimension of the list and the bigger number of iterations.

All in all, the experiment and the relative simplicity of the proposed algorithm implementation indicate appropriateness of the stated above methods application for the considered class of the problems.

## 5 Conclusion

This paper is devoted to the development of swarm intelligence approach for one variant of the competitive facility location and design problem. It is known that the location problem considered in this paper is NP-hard. Sience the objective function of corresponding mathematical model is non-linear, it is impossible to use the linear programming methods to solve problem. The ant colony algorithm for the search of approximate solutions have been built, their parameter setting has been carried out with the help of a special computational experiment. The proposed algorithm was implemented, as a result of the computational experiment, interesting data are obtained. Due to the fact that not all test instances

**Table 3.** CPU time, sec

| | 60 | 80 | 100 | 150 | 200 | 300 |
|---|---|---|---|---|---|---|
| min | 4 | 13 | 1 | 67 | 136 | 450 |
| aver | 14 | 31 | 54 | 170 | 308 | 1021 |
| max | 33 | 48 | 116 | 343 | 625 | 2036 |

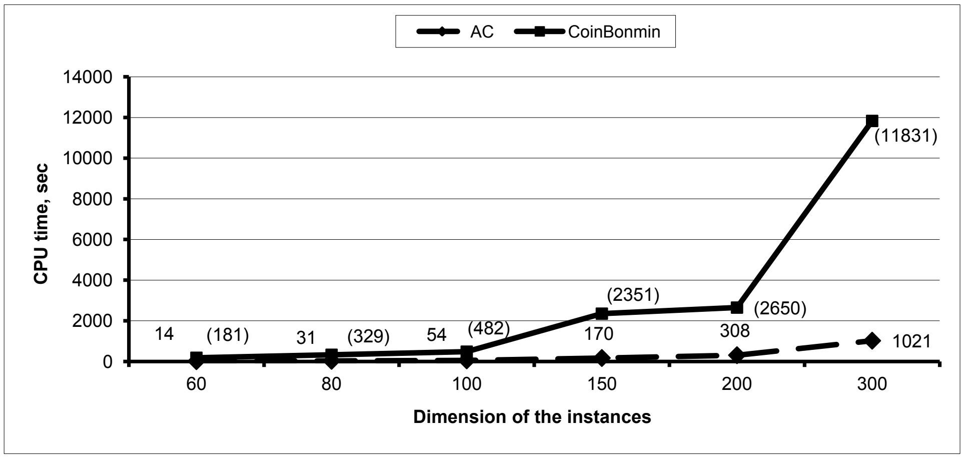


**Fig. 1.** Comparing the average CPU time of the ant colony algorithm and CoinBonmin.

know the values of the objective function, it was necessary to work in its upper bounds. Thus, for a series of test cases with uniform distribution distances the minimum deviation from the upper bounds did not exceed 0.0001%, and for the other series with Euclidean distances this value was not less than 11%. The proposed algorithm works faster than the known software up to 13 times, but it's not fast enough for a metaheuristic. Perhaps here it is worth using other upper bounds, adjust the rules of their construction, improve the development of the algorithm.

All in all, the results of computational experiment and the relative simplicity of the proposed algorithm implementation indicate appropriateness of the stated above methods application for the considered class of the problems.

## References

1. Alexandrov, D., Kochetov. Yu.: The behavior of ant colony algorithm for the set covering problem. In: Operations Research Proc. pp. 255–260. Springer (1999)
2. Colorny, A., Dorigo, M., Maniezzo, V.: Distributed optimization by ant colonies. In: First European Conference on Artificial Life. pp. 134–142. Elsevier (1992)

3. Dorigo, M., Birattari, M., Stützle, T.: Ant Colony Optimization. Artificial Ants as a Computational Intelligence Technique. IRIDIA – Technical report series: TR/IRIDIA/2006–023 (2006)
4. Dorigo, M., Di Caro, G., Gambardella, L.M.: Ant algorithms for discrete optimization. Artificial Life 5(2), 137–172 (1999)
5. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: An autoctalytic optimizing process. Report no TR-91-016. 16 (1991)
6. Dorigo, M., Gambardella, L.M.: Ant colony system: A cooperative learning approach to the Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation 1(1), 53–66 (1997)
7. Dorigo, M., Maniezzo, V., Colorny, A.: The ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B 26(1), 29–41 (1996)
8. Gambardella, L.M., Taillard, E.D., Dorigo, M.: Ant colonies for the QAP. J. Oper. Res. Soc. (JORS) 50(2), 167–176 (1999)
9. Gendreau, M., Potvin, J.-Y. (eds.): Handbook of Metaheuristics, 2nd. 648 p. Springer (2010)
10. Junior, I.C.: Data mining with ant colony algorithms. In: Huang, D.-S., Jo, K.-H., Zhou, Y.-Q., Han, K. (eds.) ICIC 2013. LNCS. vol. 7996, pp. 30–38. Springer, Heidelberg (2013)
11. Jafar, M., Sivakumar, R.: Ant-based clustering algorithms: a brief survey. Int. J. Comput. Theor. Eng. 2, 787–796 (2010)
12. Kaveh, A., Sharafi, P.: Ant colony optimization for finding medians of weighted graphs. Engineering Computations 25(2), 102–120 (2008)
13. Kochetov, Y., Alekseeva, E., Levanova, T., Loresh, M.: Large neighborhood local search for the $p$-median problem. Yugoslav J. of Oper. Res. 15(2), 53–64 (2005)
14. Aboolian, R., Berman, O., Krass, D.: Competitive facility location and design problem. Eur. J. Oper. Res. 182(1), 40–62 (2007)
15. Levanova, T.V., Belan S.E.: Application of upper bounds for analysis of approximate algorithms for a competitive location problem with elastic demand. Vestnik Omskogo universiteta – Herald of Omsk University 4(86), 4–10 (2017). https://doi.org/10.25513/1812-3996.2017.4.4-10 (in Russian)
16. Levanova, T., Gnusarev, A.: Variable neighborhood search approach for the location and design problem. In: Kochetov, Y. et al (eds.) DOOR-2016. LNCS. vol. 9869, pp. 570–577. Springer, Heidelberg (2016)
17. Levanova, T., Loresh, M.: Ant colony and similuted annealing algorithms for the $p$-median problem. In: 2nd Int. Workshop on Discrete Optimization Methods in Production and Logistics (DOM'2004). pp. 70–74 (2004)
18. Levanova, T.V., Loresh, M.A.: Algorithms of ant system and simulated annealing for the $p$-median problem. Automation and Remote Control 65 (3), 431–438 (2004)
19. Pedemonte, M., Nesmachnow, S., Cancela, H.: A survey on parallel ant colony optimization. Appl. Soft Comput. 11(8), 5181–5197 (2011)
20. Stützle, T., Hoos, H.H.: MAX-MIN ant system. Future Generation Computer Systems 16, 889–914 (2000)
21. Valeeva, A.F., Goncharova, Yu.A.: Practical application of population based ant colony optimization algorithm. Vestnik UGATU, vol. 17, 6(59), 75–78 (2013)
22. Yang, X.S.: Nature-Inspired Metaheuristic Algorithms. 148 p. Luniver Press (2008)