A METAHEURISTIC ALGORITHM FOR THE ANTI COVERING LOCATION PROBLEM

Branka Dimitrijević
University of Belgrade, Faculty of Transport and Traffic Engineering, brankad@sf.bg.ac.rs
Milica Šelmić
University of Belgrade, Faculty of Transport and Traffic Engineering, m.selmic@sf.bg.ac.rs
Dušan Teodorović
University of Belgrade, Faculty of Transport and Traffic Engineering, dusan@sf.bg.ac.rs
Vladimir Simić[*]
University of Belgrade, Faculty of Transport and Traffic Engineering, vsima@sf.bg.ac.rs

***Abstract**: In order to improve performances of logistics systems, one of tasks is to properly locate relevant logistics objects. Some of these tasks can be modeled as the Anti-Covering Location Problem (ACLP). The ACLP belongs to the class of discrete location problems and could be defined in the following way: for a given set of potential facility location sites, a maximally weighted set of facilities is located in such a way that no two placed facilities are inside a pre-specified distance of each other. It is NP hard problem. In this paper we presented the Bee Colony Optimization (BCO) tailored for the ACLP. The BCO algorithm is popular technique inspired by bees' behavior in nature. In the contrary to the previously published papers related to the BCO application on the ACLP, here we focus on weighted version of this location problem. The proposed approach is tested on numerous benchmark problems.*

Keywords: *Anti Covering Location Problem, Bee Colony Optimization, Logistics objects.*

[*] Corresponding author

1. INTRODUCTION

The Bee Colony Optimization (BCO) is a meta-heuristic which belongs to the class of stochastic swarm optimization methods. The BCO is inspired by the foraging habits of bees in the nature. The basic idea behind the BCO is to create the multi agent system (colony of artificial bees) capable to successfully solve difficult combinatorial optimization problems. The artificial bee colony behaves partially alike, and partially differently from bee colonies in nature. The BCO has been proposed by Lučić and Teodorović [1, 2, 3, 4], and belongs to the class of constructive methods. It was designed as a method which built solutions from the scratch with execution steps, contrary to the local search based meta-heuristics, which perform iterative improvements of the current best solutions. In this paper we present the BCO algorithm tailored to deal with the Anti-Covering Location Problem (ACLP).

The ACLP belongs to an important class of location models which maximizing the total value of facilities sited while also ensuring spatial separation between located facilities. It is discrete location problem and could be defined in the following way: for a given set of potential facility location sites, a maximally weighted set of facilities is located in such a way that no two placed facilities are inside a pre-specified distance or time standard of each other. In the case of ACLP the total number of facilities to be sited is not given in advance.

If we define a network where the set of facility sites represent the nodes and the arcs represent each pair of nodes which lie within the pre-specified distance or time standard, then the ACLP is equivalent to the Node/Vertex Packing Problem once the network has been defined using the restriction requirement. Node/Vertex Packing Problem and ACLP are closely related to the Maximal Clique and Vertex Cover Problems. Further, they are related to minimum separation and dispersion models [5].

A wide variety of particular applications of the ACLP and related problems can be found in literature. The applications include forest management, nature reserve design, telecommunications equipment sitting, military defense location, undesirable facility location, water conservation, social service provision, zoning policy development, franchise outlet location, cartographic design, and a range of other planning contexts [7, 8, 9, 10, 11, 12, 13].

The ACLP, as the NP-hard problem [14], was solved by various heuristic and meta-heuristic algorithms: Greedy Heuristic algorithms [9], Lagrangian relaxation [6] and Genetic Algorithm [14]. The BCO implementation on the "unweighted" version of the ACLP (all potential facility sites have the same weights that are equal to one) [13] showed that the BCO is very competitive with other state-of-art techniques. This implementation was illustrated on example of dangerous goods warehouse location problem as an important location problem in the field of hazardous materials logistics. One of directions for future research listed in [13] was to tailor the BCO on "weighted" version of the ACLP. Thus, the purpose of this paper is to present the BCO algorithm tailored to deal with more realistic, "weighted" version of the ACLP and to demonstrate that in this case it can also be successfully applied.

This paper is organized as follows. Section after Introduction provides mathematical formulation of the ACLP. Content of the following Section presents brief description of the BCO and actual implementation of the BCO for solving "weighted" version of the ACLP. Section named Computational examples contains experimental evaluations and the last Section brings conclusions related to the research.

2. THE ACLP FORMULATION

The ACLP was introduced by Moon and Chaudhry [8]. There are few mathematical formulations of the ACLP proposed in literature [6, 8].

Let us introduce binary variables x_i defined in the following way:

$$x_i = \begin{cases} 1 \text{ if node } i \text{ is chosen to be a facility location site} \\ 0 \text{ otherwise} \end{cases} \quad (1)$$

Consider the following notation:
i – index representing potential location sites,
n – total number of potential location sites,
w_i – node weight (benefit associated with the use of location i),
d_{ij} – the shortest distance between node i and node j,
R – pre-specified minimal distance,
$\pi_i = \{j | d_{ij} \le R \wedge i \ne j\}$ – nodes that are on distance less or equal to R, excluding particular node i,
M – a large positive number.
The following, original, mathematical formulation of the ACLP is proposed by Moon and Chaudhry [8]:

$$max \quad Z = \sum_i w_i x_i \quad (2)$$

$$Mx_i + \sum_{j \in \pi_i} x_j \le M, \quad \forall i \quad (3)$$

$$x_i \in \{0,1\}, \quad \forall i \quad (4)$$

The objective function (2) maximizes the total weighted selection of the facility location sites. Constraints (3) are referred as Neighborhood Adjacency Constraints (NAC). If node i is selected for facility placement (i.e. x_i = 1), then the term Mx_i equals the right hand side term, M, and forces $\sum_{j \in \pi_i} x_j = 0$. Thus, if a site i is used, then all sites j within the R distance neighborhood of site i , π_i, are restricted from use. Constraint (4) defines problem binary variables.

Other mathematical formulations of the ACLP [6] differ from the original ACLP formulation in the specification of the NAC, because of the impact that NAC structure has on problem solvability.

3. THE BEE COLONY OPTIMIZATION

The Bee Colony Optimization meta-heuristic was introduced by Lučić and Teodorović [1, 2, 3, 4] as a new direction in the field of Swarm Intelligence.

The BCO algorithm is inspired by the foraging behavior of honeybees. The basic plan behind BCO is to build a multi-agent system (a colony of artificial bees) that can efficiently solve hard combinatorial optimization problems. The artificial bee colony behaves similarly to bee colonies in nature in some ways but differently from them in other ways.

3.1 The BCO algorithm

During the evolution of the BCO algorithm authors developed two different approaches. The first approach is based on constructive steps in which bees build solutions step by step. The second approach of the BCO algorithm is based on the improvement of complete solutions in order to obtain the best possible final solution. In this paper we use constructive concept due to the problem nature.

The BCO is a population based algorithm. Population of the *artificial bees* searches for the optimal solution. Every artificial bee generates one solution to the problem. The algorithm consists of two alternating phases: *forward pass* and *backward pass*. During each forward pass, every bee is exploring the search space. It applies a predefined number of moves, which construct and/or improve the solution, yielding to a new solution.

Having obtained new partial solutions, the bees return to the nest and start the second phase, the so-called backward pass. During the backward pass, all bees share information about their solutions. In nature, bees would perform a dancing ritual, which would inform other bees about the amount of food they have found, and the proximity of the patch to the nest. In the search algorithm, the bees announce the quality of the solution, i.e. the value of objective function. During the backward pass, every bee decides with a certain probability whether it will advertise its solution or not. The bees with better solutions have more chances to advertise their solutions. The remaining bees have to decide whether to continue to explore their own solution in the next forward pass, or to start exploring the neighborhood of one of the solutions being advertised. Similarly, this decision is taken with a probability, so that better solutions have higher probability of being chosen for exploration.

The two phases of the search algorithm, forward and backward pass, are performed *iteratively*, until a stopping condition is met. The possible stopping conditions could be, for example, the maximum total number of forward/backward passes, the maximum total number of forward/backward passes without the improvement of the objective function, etc.

The BCO algorithm parameters whose values need to be set prior the algorithm execution are as follows:

B - the number of bees involved in the search and
NC - the number of constructive / improvement moves.

The pseudo-code of the BCO algorithm could be described in the following way:

```
Do
1. Initialization: a(n) (empty) solution is assigned to each bee.
2. For ($i = 0$; $i < NC$; $i$ ++ )
//forward pass
(a) For ($b = 0$; $b < B$; $b$++)
    i) Evaluate possible moves.
    ii) Choose one move using the roulette wheel.
//backward pass
(b) For ($b = 0$; $b < B$ ; $b$++)
    Evaluate the partial/complete solution for bee $b$;
(c) For ($b = 0$; $b < B$; $b$++)
    Loyalty decision using the roulette wheel for bee $b$;
(d) For ($b = 0$; $b < B$; $b$++)
    If ($b$ is uncommitted), choose a recruiter by the roulette wheel.
3. Evaluate all solutions and find the best one.
while stopping criteria is not satisfied.
```

Steps 1, 2(a) and 2(b) are problem dependent and should be resolved in each BCO implementation. On the other hand, there are formulae specifying steps 2(c), loyalty decision, and 2(d), recruiting process, and they are all described in the next section in details.

4. THE BCO APPROACH TO THE ACLP

In this section, we describe our implementation of the BCO algorithm to be applied to the weighted version of Anti Covering Location Problem. In order to make it more interesting to logistics engineering audience, we choose to illustrate it on example of dangerous goods warehouse (in following text warehouse) location problem. Those facilities (for example: radioactive waste warehouses, explosives warehouses, as well as noise, odor or heat emitters, etc.) generate different undesirable effects that can be felt within certain geographical area. Making decisions about their spatial positions are crucial when it comes to minimize the environmental risk. Such facilities should be located under condition of the minimal safety distance. For some dangerous goods (for example some explosives) the minimal safety distance may be determined as constant value which depends only on the dangerous goods' characteristics. Weights which represent the warehouse capacities are associated to potential locations.

The objective is to maximize quantity of dangerous goods stored respecting the minimal safety distance between warehouse facilities.

The main specificity of BCO application to the ACLP is in the fact that *NC* has to be equal to one, and number of located warehouses isn't known in advance.

Let us denote by V_i bee's utility when choosing the node i to be a warehouse site (within a single forward step, each bee has to select NC=1 node). However, in this paper, it is assumed that there are two bee utility criteria, i.e. V_{1i} and V_{2i}, in order to better describe the nature of the weighted ACLP. In this sense, the first utility criterion is:

$$V_{1i} = \frac{N^{max} - N_i}{N^{max} - N^{min}}, \quad i = 1,2,...,n \tag{5}$$

where:

$N^{max} = \max_i \left(w_i + \sum_{j \in \pi(i)} w_j \right)$ - the largest possible sum of nodes' weights which are jeopardized by any observed location in current forward pass;

$N^{min} = \min_i \left(w_i + \sum_{j \in \pi(i)} w_j \right)$ - the smallest possible sum of nodes' weights which are jeopardized by any observed location in current forward pass;

$N_i = w_i + \sum_{j \in \pi(i)} w_j$ - the sum of nodes' weights which are jeopardized by the i-th observed location in current forward pass.

The second utility criterion is:

$$V_{2i} = \frac{w_i}{N_i}, \quad i = 1,...,n \tag{6}$$

We set up p_i (the probability that specific bee chooses node i) to:

$$p_i = \frac{T_i}{\sum_{k=1}^{K} T_k}, \quad i = 1,...,n \tag{7}$$

where:

T_i - the relative closeness to the ideal solution for node i calculated with TOPSIS method [16] and with the following (empirically obtained) criteria weights: 0.75 for V_{1i} and 0.25 for V_{2i}.

K - the number of "free" nodes (not previously chosen).

Using relation (7) and a random number generator, we determine the nodes to be chosen by each bee.

After determining warehouse locations in current partial solution, it is necessary to evaluate all bees' solutions. It is obviously that particular partial solution is better if the sum of located nodes' weights is higher and vice versa.

Let us denote by C_i (i=1,2,...,b) sum of nodes' weights in the solution generated by the i-th bee. Let us normalize the value C_i. We denote by O_i normalized value of C_i, i.e.:

$$O_i = \frac{C_i - C_{max}}{C_{max} - C_{min}}, \; O_i \in [0,1], \; i = \overline{1,b} \tag{8}$$

where C_{min} and C_{max} are the minimal and the maximal sum of nodes' weights obtained by all bees, respectively.

After the completion of a forward pass, each bee decides whether it stays loyal to the previously discovered partial solution or not. This decision depends on the quality of its own solution related to all other existing solutions. The probability that b-th bee (at the beginning of the new forward pass) is loyal to its previously generated partial solution is expressed as follows:

$$p_b^{u+1} = e^{-\frac{O_{max} - O_b}{u}}, \quad b = 1,..., B \tag{9}$$

where:

O_b - denotes the normalized value for the objective function of partial solution created by the b-th bee;

O_{max} - represents the maximum over all normalized values of partial solutions to be compared;

u - counter of the forward passes (taking values 1, 2, .., NC).

For each uncommitted bee it is decided which recruiter it will follow, taking into account the quality of all advertised solutions. The probability that b's partial solution would be chosen by any uncommitted bee equals:

$$p_b = \frac{O_b}{\sum_{k=1}^{R} O_k}, \quad b = 1,..., R \tag{10}$$

where O_k represents the normalized value for the objective function of the k-th advertised solution and R denotes the number of recruiters. Using equation (10) and a random number generator, each uncommitted follower joins one recruiter through a roulette wheel.

5. COMPUTATIONAL EXAMPLES

Computational results for the BCO algorithm are presented in this section. All analyzed problem instances are generated by the authors of this paper and they are available upon request. The obtained results are presented in the Table 1. All the tests were performed using SciLab (version 4.0) on Intel Core i7 i7-4820K computer processor with 3.7 GHz and 4 GB of RAM.

The proposed BCO algorithm was capable to find the optimal solution for all analyzed problem instances. The CPU times for all problem instances varied from 0.01 to 0.101 second. Additionally, we solved all tested examples by LpSolve in order to compare the minimum BCO CPU times needed to obtain the solutions with ones which were reported by LpSolve. LpSolve was capable to find the optimal solution for all analyzed problem instances and its CPU times varied in [0.1, 0.5] seconds interval. Therefore, presented numerical results showed that the BCO generated successful performances. These performances will become more important for a larger size networks.

Table 1. Results for the generic data set (20 nodes)

R^a	D^b	O^c	BCO^d	I^e	CPU^f	$BCO_m{}^g$	$I_m{}^h$	$CPU_m{}^i$
50	0.076	6840	6840	1	0.001	6840	7.9	0.004
100	0.126	6840	6840	416	0.125	6840	2403.0	0.707
150	0.168	6789	6789	8	0.003	6789	63.7	0.021
200	0.210	5525	5525	40	0.010	5525	8006.2	1.917
250	0.260	5332	5332	1861	0.041	5332	17698.3	3.865
300	0.326	5068	5068	138	0.046	5068	7932.9	1.587
350	0.376	4855	4855	200	0.038	4855	1759.9	0.327
400	0.421	4855	4855	181	0.034	4855	3396.9	0.591
450	0.492	4298	4298	13	0.002	4298	5096.8	0.841
500	0.545	4298	4298	1.5	0.101	4298	10576.1	1.617
550	0.608	3543	3543	55	0.010	3543	19314.6	2.725
600	0.647	3084	3084	1	0.001	3084	31.1	0.006
650	0.690	2463	2463	1	0.002	2463	53.3	0.010
700	0.732	2316	2316	1	0.001	2316	26.8	0.007
750	0.774	1798	1798	8	0.002	1798	37.0	0.007
800	0.832	1798	1798	1	0.001	1798	20.1	0.005
850	0.868	1702	1702	4	0.001	1702	17.2	0.005
900	0.90	1702	1702	1	0.001	1702	14.6	0.005
950	0.953	1576	1576	1	0.001	1576	15.4	0.002
1000	1.0	859	859	1	0.001	859	7.6	0.002

[a] Pre-specified minimum distance.
[b] Network density.
[c] Optimal objective function value.
[d] The best objective function value obtained by BCO using two artificial bees .
[e] The minimum number of iterations (among optimal solutions discovered) during ten problem solving cycles.
[f] The minimum CPU time in seconds (among optimal solutions discovered) during ten problem solving cycles.
[g] The mean objective function value obtained by BCO using two artificial bees during ten problem solving cycles.
[h] The mean number of iterations during ten problem solving cycles.
[i] The mean CPU time in seconds during ten problem solving cycles.

6. CONCLUSION

The Bee Colony Optimization (BCO) meta-heuristic is used as a tool to solve weighted version of the Anti-Covering Location Problem (ACLP). This paper represents a natural extension of our previous study [13], in which ‘unweighted’ ACLP was solved.

For the first time in relevant literature we considered the multi-objective approach in order to determine utilities that bees have when locate a facility at a given node. We stated from the assumption that decisions related to the utility of facility/warehouse locations should to be made in the presence of trade-offs between two conflicting criteria. We merged these criteria using TOPSIS method.

The BCO algorithm was tested on a variety of numerical examples. The performed numerical experiments show that the proposed algorithm can generate high-quality solutions in a reasonable CPU times.

ACKNOWLEDGMENT

This work was partially supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia through projects TR 36006 and TR 36002 for the period 2011-2014.

REFERENCES

[1] Lučić, P. and Teodorović, D., 2001. *Bee System: Modeling Combinatorial Optimization Transportation Engineering Problems by Swarm Intelligence.* Preprints of the TRISTAN IV Triennial Symposium on Transportation Analysis, 441-445, Sao Miguel, Azores Islands, Portugal.

[2] Lučić, P. and Teodorović, D., 2002. *Transportation Modeling: An Artificial Life Approach.* Proceedings of the 14th IEEE Int. Conference on Tools with Artificial Intelligence, 216-223, Washington D.C., USA

[3] Lučić, P. and Teodorović, D., 2003. *Vehicle Routing Problem With Uncertain Demand at Nodes: The Bee System and Fuzzy Logic Approach.* Fuzzy Sets in Optimization, J.L. Verdegay, ed., Springer – Verlag, Berlin Heidelberg, 67-82.

[4] Lučić, P. and Teodorović, D. 2003. *Computing with Bees: Attacking Complex Transportation Engineering Problems.* International Journal on Artificial Intelligence Tools 12, 375-394.

[5] Erkut, E., ReVelle, C. and Ulkiisal, Y. 1996. *Integer-friendly formulations for the r-separation problem*, Eurpean Journal of Operational.Research 92, 342–351.

[6] Murray, A. T. and Church, R. L., 1997. *Solving the anti-covering location problem using lagrangian relaxation*. Computers & Operations Research 24, 127-140.

[7] Barahona, F., Weintraub, A. and Epstein, R., 1992. *Habitat dispersion in forest planning and the stable set problem*. Operations Research, 40, I S14-S21.

[8] Moon, I. D. and Chaudhry, S. S., 1984. *An analysis & network location problems with distance constraints.* Management Science, 30, 290-307.

[9] Chaudhry, S., McCormick, S. T. and Moon, I. D., 1986. *Locating independent facilities with maximum weight: greedy heuristics*. Omega, 14, 383-389.

[10] Goycoolea, M., Murray, A.T., Barahona, F., Epstein, R. and Weintraub, A., 2005. *Harvest scheduling subject to maximum area restrictions:*

exploring exact approaches, Operations Research 53, 490–500.

[11] Grubesic, T.H. and Murray, A.T., 2008. *Sex offender residency and spatial equity.* Applied Spatial Analysis and Policy 1, 175-192.

[12] Downs, J.A., Gates, R.J. and Murray, A.T., 2008. *Estimating carrying capacity for sandhill cranes using habitat suitability and spatial optimization models.* Ecological Modelling 214, 284–292.

[13] Dimitrijevic, B., Teodorovic, D., Simic, V. and Selmic, M., 2012. *Bee Colony Optimization Approach to Solving the Anticovering Location Problem.* Journal of Computing in Civil Engineering 26(6), 759–768.

[14] Garey, M. and Johnson, D., 1979. *Computers and Intractability: A guide to the Theory of NP Completeness*, Freeman and Company, New York.

[15] Chaudhry, S. S., 2006. *A genetic algorithm approach to solving the anti-covering location problem*. Expert Systems, 23, 251-257.

[16] Hwang, C.L. and Yoon, K., 1981. *Multiple attribute decision making – methods and applications: A state of the art survey*, Springer-Verlag, New York.