Bird colony algorithm use in the logistics module of the production management system

I A Belousov and K I Firsov
Moscow Aviation Institute (National Research University), 4, Volokolamskoe shosse, Moscow, 125993, Russia
E-mail: kifirigor@gmail.com

Abstract. The report discusses the application of the bird colony algorithm together with the solution of the clustering problem in the logistics module of the ERP-system. This polynomial algorithm belongs to the category of stochastic biotic algorithms. The main rule of the implementation of the bird colony algorithm is that the so-called swarm intelligence is used to solve the optimization problem. The article describes the rule of algorithm operation, the study of the properties of this algorithm based on the results of modeling on a real base of target points and makes a comparative analysis of the algorithm with previously used algorithms in the system. The application of the algorithm in matrix form using the entropy matrix is described in the report. The obtained results allow us to say with confidence that the use of the bird colony algorithm can significantly increase the algorithm speed of logistic module.

1. Introduction
The logistics module is designed to plan the use of more than one hundred and twenty vehicles operating simultaneously from one base, despite the fact that the number of target points per day usually exceeds the number of vehicles in several times. The article discusses the module algorithms to solve the problem of a priori partitioning points into clusters and constructing a route for each, if possible, optimal.

The task should be solved considering the following limitations:

- vehicle capacity;
- the duration of the route based on the power reserve and the limitations of the person driving the vehicle;
- deadlines for completing the task;
- allocated time for building/rebuilding the route;
- desirability of obtaining a global optimum;
- the vehicle must return to the point from which it started moving.

The system has the ability to receive and save real vehicle trajectories from on-board navigators. As a result, this system has accumulated a large database of target points over 8 years, planned and developed routes, that allows to check the correctness of the forecast and modeling of uncontrollable factors a posterior for a comparative analysis of different algorithms on real material.
To solve this problem, we propose to consider the results of using the bird colony algorithm [1] in this module together with the K-middle clustering algorithm in cartesian and polar coordinate systems. The system was tested on five different algorithms [2–4].

2. Clustering in the polar coordinate system

Preliminary clustering is performed by the k-nearest neighbor method. It’s proposed to calculate up to the cluster centers in the polar coordinate system. This method of partitioning into clusters works perfectly in the case of vehicles starting from the same starting point.

To translate the coordinates of points from the cartesian coordinate system to the polar coordinate system, the formulas are used:

\[ r = \sqrt{x^2 + y^2}, \quad \varphi = \arctan(y/x). \]

As all routes start from one point (the base), we will consider this point as the beginning of coordinates. Thus, the equations will take the following form:

\[ r = \left[ (x - x_b)^2 + (y - y_b)^2 \right]^{1/2}, \]

where \( x_b \) and \( y_b \) are the coordinates of the center of the polar coordinate system in the cartesian coordinate system (base).

We get the points whose coordinates are now in the polar coordinate system \( p(r_i, \varphi_i) \), \( i = 1, \ldots, n \) where \( n \) is number of points.

In the cartesian coordinate system, the cluster centers were chosen in such a way that their coordinates should be as far apart as possible, in other words, the Euclidean distance between the cluster centers is maximized. In the polar coordinate system, the cluster centers are selected from the points, so that their angle values should differ as much as possible from each other

\[ p_{c,0} = \max(\varphi_i - \varphi_0), \quad p_{c,k} = \max(\varphi_{c,k} - \varphi_i), \quad i = 1, \ldots, n, \quad k = 1, \ldots, m, \]

where \( p_{c,0} \)—first cluster center, \( p_{c,k} \)—k-th cluster center, \( m \)—total number of clusters.

For each cluster, the center is obtained in this way, it’s necessary to take the cartesian coordinates of the remaining points and translate them into the polar coordinate system. The point belongs to the cluster for which the value is minimal.

3. Bird colony algorithm

The Bird Colony algorithm (BCA) is a new meta-heuristic algorithm that has been published in [1]. The principle of implementing the bird colony algorithm is that the so-called swarm intelligence is used to solve the optimization problem. Its essence lies in the fact that the movement of a bird colony can be described mathematically and thus obtain a solution that will find the global optimum when solving the problem of route optimization. In other words, the bird colony algorithm models the behavior of a flock of birds and can be applied to find solutions to difficult or unsolvable combinatorial problems [5, 6]. The advantage of this algorithm is that in comparison with other algorithms from this class of problems, the bird colony algorithm has high convergence and accuracy.

To apply the algorithm to the task, it is necessary to upgrade it. We add the so-called entropy matrix [7], which is a matrix of size \( N \times N \), where \( N \)—number of cities, each element of which \( I_{ij} \) is a number by which you can select the point \( j \) that follows the point \( i \). Each bird of the colony is looking for its own solution to the optimization problem (builds a route around the points), which is presented as a list containing a sequence of flying points. Because of the difference function (discussed below), it is necessary to represent this solution in the form
of a Boolean matrix $B$ (containing ones and zeros). In BCA, the difference function is used to calculate the difference between two Boolean matrices. The result of the difference function is used to calculate the new bird entropy matrix. This calculation occurs $M$ times in order to increase the accuracy of the solution of the problem [8].

The algorithm simulates the behavior of a bird in a colony and is used to solve the problem of optimizing the flyby of points. The essence of the algorithm is that the bird can “switch” between different states: foraging (flight for food), observation (search for food), as well as the state when it can fly with the flock to another point.

During foraging, each bird looks for food based on personal experience and the experience of the entire flock. This can be described mathematically:

$$x_{i,j}^t = x_{i,j}^{t-1} + C(p_{i,j} - x_{i,j}^{t-1}) \text{rand}(0,1) + S(g_{\text{best},j} - x_{i,j}^{t-1}) \text{rand}(0,1),$$

where $x_{i,j}^{t-1}$ is the value of the $j$-th element of the $i$-th solution to generate $t$, $\text{rand}(0,1)$ is a uniform distribution function, $p_{i,j}$ is the best previous position of the $j$-th element of the $i$-th bird, and $g_{\text{best},j}$ means the $j$-th element of the global optimal solution. $C$ and $S$ are two positive numbers, which are called cognitive and social coefficients, respectively. When the bird is in a state of observation (searching for food), each bird will try to move to the center of the flock and will inevitably compete with the others. This behavior can be represented as follows:

$$x_{i,j}^t = x_{i,j}^{t-1} + A_1(\text{mean}_j - x_{i,j}^{t-1}) \text{rand}(0,1) + A_2(p_{i,j} - x_{i,j}^{t-1}) \text{rand}(0,1),$$

where $A_1$ and $A_2$ are two positive constants within [0, 2]; $p_{\text{Fit}_i}$ denotes the best value of the weight function of the $i$-th bird; $\varepsilon$ is used to avoid division by zero and is the smallest constant on the computing machine; $\text{mean}_j$ denotes the $j$-th element of the average position of the entire pack.

The birds will periodically change their location. During the flight, from time to time, there is a change of roles between the breadwinners and the dependents. Birds whose weight function is the maximum to be miners, while birds with the smallest weighting functions are dependent. The rest of the birds will be randomly assigned roles. The behavior during the flight can be mathematically represented as follows:

$$x_{i,j}^t = x_{i,j}^{t-1}(1 + \text{randn}(0,1)),$$

$$x_{i,j}^t = x_{i,j}^{t-1} + FL(x_{k,j}^{t-1} - x_{i,j}^{t-1}) \text{rand}(0,1),$$

where randn(0,1) is a Gaussian distribution with mathematical expectation equal to zero and SKO equal to 1, $k \in [0, N], k \neq i$, $FL \in (0, 2)$—the probability that the dependents will pursue the foragers in order to find food. Considering the individual characteristics, the $FL$ value of each dependent will be randomly selected from 0 to 2. The birds switch between roles every $FQ$ steps.

For the current problem, the element of the entropy matrix can be calculated as follows:

$$I(x_{i,j}) = \log_2(1/p_{i,j}).$$

The greater value $I(x_{i,j})$ takes, the greater probability of the choose of the path from point $i$ to point $j$ as the next point is. $P_{ij}$ is the probability of the path from point $i$ to point $j$ and can be calculated as follows:

$$P_{ij} = \frac{\text{dist}(i,j)}{\sum_{k=1}^{n} \text{dist}(i,k)} \quad (i \neq j, i \neq k),$$
where \( \text{dist}(i, j) \) — distance between point \( i \) and point \( j \). Thus, for four points, we get the following matrix:

\[
X_I = \begin{bmatrix}
I(x_{1,1}) & I(x_{1,2}) & I(x_{1,3}) & I(x_{1,4}) \\
I(x_{2,1}) & I(x_{2,2}) & I(x_{2,3}) & I(x_{2,4}) \\
I(x_{3,1}) & I(x_{3,2}) & I(x_{3,3}) & I(x_{3,4}) \\
I(x_{4,1}) & I(x_{4,2}) & I(x_{4,3}) & I(x_{4,4})
\end{bmatrix}.
\]

However, this algorithm can be improved by methods described in [9, 10].

### 4. Improved bird colony algorithm

During testing, it was revealed that the performance of the algorithm directly depends on the initial population of “birds”. The bird colony algorithm cannot guarantee the diversity of population solutions, because it generates the initial solution in random order and this affects the performance of the algorithm. Therefore, we can apply an approach, the essence of which is that adaptive coefficients are introduced to initialize the population. In this paper, we apply this method to evenly distribute individuals in the search space and improve the optimization capabilities of the bird colony algorithm.

We introduce such a concept as adaptive inertial weight, which we will add to equation (1). Balancing between local and global search is achieved by varying the value of the inertial weight \( W \):

\[
W = W_{\text{max}} \left( 1 - \ln \left( 1 + \frac{e^{t - 1}}{t_{\text{max}}} \right) \right) + W_{\text{min}} \ln \left( 1 + \frac{e^{t - 1}}{t_{\text{max}}} \right),
\]

where \( W_{\text{max}} = 0.9, W_{\text{min}} = 0.4, t \) — the current iteration number, \( t_{\text{max}} \) — the maximum possible iteration. This method can speed up the convergence and improve the performance of the algorithm.

Nonlinear adjustment of learning coefficients: \( C \) and \( S \) in the bird algorithm are the cognitive and social coefficients, respectively. This approach adapts the cognitive and social coefficients. In the initial iterations of the algorithm, \( C \) and \( S \) are equal. This means that “individual” and “group” have the same effect on the particle search process. In later iterations, \( C \) takes a small value, \( S \) takes a large value, which enhances the local search capabilities. New formulas for calculating the cognitive and social coefficients:

\[
C = \sqrt{2\pi} \exp \left[ -0.5(t/t_{\text{max}})^2 \right], \quad S = \sqrt{2\pi} - 1/\sqrt{2\pi} \exp \left[ -0.5(t/t_{\text{max}})^2 \right] - 1.
\]

The improved foraging formula will look like this:

\[
x_{i,j}^t = W_{i,j}^t + C(p_{i,j} - x_{i,j}^{t-1}) \text{rand}(0, 1) + S(g_{\text{best},j} - x_{i,j}^{t-1}) \text{rand}(0, 1).
\]

Since the dependents seek for food by chasing the foragers, there is a problem when the solution of the forager falls into the local optimum, and then the dependents also fall into the local optimum. In order to solve this problem, perturbation elements are added to the position formula of the miner, which may increase its ability to obtain a global optimal solution to avoid local optimization. Improved getter formula:

\[
x_{i,j}^t = x_{i,j}^t + x_{i,j}^{t-1}(1 + \text{randn}(0, 1)) + \text{randn}(0, 1) \left( \frac{t_{\text{max}} - t + 1}{t_{\text{max}}} \right)^2.
\]

The position of the forager adds perturbation: formula (2) is a positional update formula for foraging, indicating that the dependent receives food from the forager.

Thus, we see that this algorithm satisfies all the constraints introduced and searches for a global optimum. The main advantage of this algorithm is that it searches for the global optimal solution in a relatively small number of iterations.
5. Simulation results

The algorithms under consideration were tested on the following model data. The elements of the distance matrix are taken from a database that stores information about actual built and planned routes. The number of points \( n \) gradually increases from 4 to 24, for each \( n \) 200 experiments are conducted, so 200 variants of distance matrices \( D \) are constructed. For each instance of the matrix, the routes are calculated by the following five algorithms:

1) bird colonies algorithm;
2) genetic algorithm;
3) branch and bound algorithm;
4) ant colony algorithm;
5) brute force algorithm.

All algorithms are implemented in C# in the JetBrains Rider interactive development environment. The calculations were performed on a computer with a 4-core Intel Core i5 processor running at a frequency of 1.4 GHz.

The average length of the constructed route and the average time of route calculation obtained from the same matrix instances for all five algorithms are analyzed. Only the results of calculations that would take too much time to get are missing (the brute force algorithm works too long if the number of points in the route is more than 10).

Figure 1 shows how the total distance constructed by the algorithms differs depending on the number of points in the route. The graph shows that when the number of points is close to 10, the bird colony and genetic algorithms give almost the same result, which is the same as the reference algorithm (the brute force method is used as the reference algorithm, but only for the number of points no more than ten).

As the number of points in the route increases, there are discrepancies in the results of the algorithms. The best result in this category is shown by the genetic algorithm because it has the smallest length of the route.

However, the main drawback of the genetic algorithm is its running time. Figure 2 shows the average running time of the algorithms. As you can easily see, the running time of the genetic algorithm is significantly longer than the bird and ant algorithms. The ant colony algorithm works out the fastest.
When the number of points becomes about 20, the branch and boundary method start working longer than the bird algorithm.

Despite the high speed of the ant algorithm, the bird algorithm produces a solution closer to the reference one and in all subsequent experiments with an increase in the dimension of the distance matrix $D$, it produces a route whose total length is less than that of the ant.

![Figure 2](image_url)

**Figure 2.** Dependence of the average running time of the algorithm on the number of points in the route.

Let’s compare the results of all the algorithms with the reference one. As a reference, we will use the brute force method, since it is guaranteed to find the best solution. To perform this, from the collected statistics, we subtract from the average indicators of the route length and time spent, the characteristics of the brute force method. The larger the difference, the worse the algorithm performance.

![Figure 3](image_url)

**Figure 3.** Advantage of using clustering in the polar coordinate system: (a) cartesian coordinate system; (b) polar coordinate system.

In figure 3, you can see that the split in the polar coordinate system shows the best result. When clustering at the beginning by angle, and then by distance, the upper points of the blue cluster will turn to red, while the length of the red route will practically not change, and the blue one will be significantly reduced.

### 6. Conclusion

The claimed convergence is investigated on real data. The bird colony algorithm under consideration, indeed, has a high speed of operation and fast convergence on small samples (comparable to the speed of operation of the “ant colony” and “branches and borders”). From figure 4 it can be seen that the bird colony algorithm builds a solution that is as close as possible to the reference one. When you increase the sample size for this algorithm, the rate of increase...
in the running time is greater than the rate of increase in the number of points. In figure 4 it is not difficult to notice that the speed of the algorithm is comparable to the “ant” and “branches and boundaries”, but at the same time, it has much greater accuracy since it is looking for a solution corresponding to the global optimum. The introduction of adaptive parameters into the equation describing the behavior of the forager allows you to expand the search intervals for a solution, which doesn’t allow you to focus on the local extremum. For transport operations from a single base, more accurate clustering and routes are obtained using polar coordinates.

References
1] Meng X B, Gao X Z, Lu L, Liu Y and Zhang H 2016 A new bio-inspired optimisation algorithm: Bird Swarm Algorithm Journal of Experimental & Theoretical Artificial Intelligence 28 673–87
2] Parashar M, Rajput S, Dubey H M, and Pandit M 2017 Optimization of benchmark functions using a nature inspired bird swarm algorithm Proceedings of the 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT) pp 1–7
3] Escario J B, Jimenez J F and Giron-Sierra J M 2015 Ant colony extended: Experiments on the travelling salesman problem Expert Systems with Applications 42 390–410
4] Chen S and Chien C 2011 Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques Expert Systems with Application 38 14439–50
5] Parashar M, Rajput S, Dubey H M and Pandit M 2017 Optimization of benchmark functions using a nature inspired bird swarm algorithm Proceedings of the 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT) pp 1–7
6] Wang X, Deng Y and Duan H 2018 Edge-based target detection for unmanned aerial vehicles using competitive Bird Swarm Algorithm Aerospace Science and Technology 78 708–20
7] Min L, Yiwen Z, Juan L and Xiaoyu L 2018 Discrete Bird Swarm Algorithm Based on Information Entropy Matrix for Traveling Salesman Problem Mathematical Problems in Engineering 2018 1–15
8] Shannon C E 1948 A mathematical theory of communication Bell Labs Technical Journal 27 379–423
9] Zhang J and Feng J 2011 Particle Swarm Optimization based on dynamic chaos perturbation and its Application Computer Engineering 1517–20
10] Zhong Y, Lin J, Wang L and Zhang H 2017 Discrete comprehensive learning particle swarm optimization algorithm with Metropolis acceptance criterion for traveling salesman problem Swarm and Evolutionary Computation 42 77–88