Review of Information Engineering and Applications
2020 Vol. 7, No. 1, pp. 6-17.
ISSN(e): 2409-6539
ISSN(p): 2412-3676
DOI: 10.18488/journal.79.2020.71.6.17
© 2020 Conscientia Beam. All Rights Reserved.

APPLICATION OF ANT ALGORITHM FOR SOFTWARE OPTIMIZATION

Shafagat Mahmudova

SE Department, Institute of Information Technology of ANAS, Baku, Azerbaijan.
Email: shafagat@gmail.com

Article History
Received: 16 January 2020
Revised: 20 February 2020
Accepted: 23 March 2020
Published: 13 April 2020

Keywords
Software
Optimization
Ant algorithm
Soft computing
Technology
Method.

ABSTRACT

Different technologies, methods and algorithms are used when developing high-quality software systems. There are various ways to create optimal software. One of them is an ant algorithm. The ant algorithm can be attributed to the field of biomimetic. Ant algorithm is one of the most effective polynomial algorithms for finding approximate solutions, and it also solves similar route search problems in graphs. Various methods, including the ant algorithm, are used to improve software efficiency and optimize it. The essence of the concept is to analyze, use, and meta-heuristically optimize the behavior of ants in search of paths to the food source of the colony. This article analyzes the studies in this area, and explains the idea of the algorithm used, compares different ant systems, shows program code operators as ants, and applies the ant algorithm to them. As a result of applying the algorithm, the shortest path to some operators (cycle, condition) contained in the program code is found. The experiments perform good results.

Contribution/Originality: The ant algorithm, various methods including the ant algorithm was studied. Analyzes the studies in this area, and explains the idea of the algorithm used, compares different ant systems, shows program code operators as ants, and applies the ant algorithm to them.

1. INTRODUCTION

Soft Computing technology focuses on solving management problems, and a set of tools utilize fuzzy systems technology: neural networks, genetic algorithms, and evolution modeling. Different Soft Computing methods complement each other and are often used together.

Software systems development technologies are based on modern automated techniques and technical tools applied during their lifecycle. These expensive tools also increase the cost of software products.

Different technologies and techniques are used in developing high-quality software systems. Optimal software is developed in various ways.

One of them is the use of soft computing [1].

Software efficiency (ISO/IEC standard 25010: 2011 (R ISO/MEK 25010-2015 state standard)) defines a product quality model, hence, it contains eight top-level characteristics [2] Figure 1. Optimizing the software code increases its.

The mathematical methods, that recently have incorporated the principles of natural decision-making mechanisms, have been intensively developing in the field of "Biomimetic". The relevance of the issue is based on it.

Various methods, including the ant algorithm, are used to improve software efficiency and optimize it. The ant algorithm can be referred to the field of biomimetic.

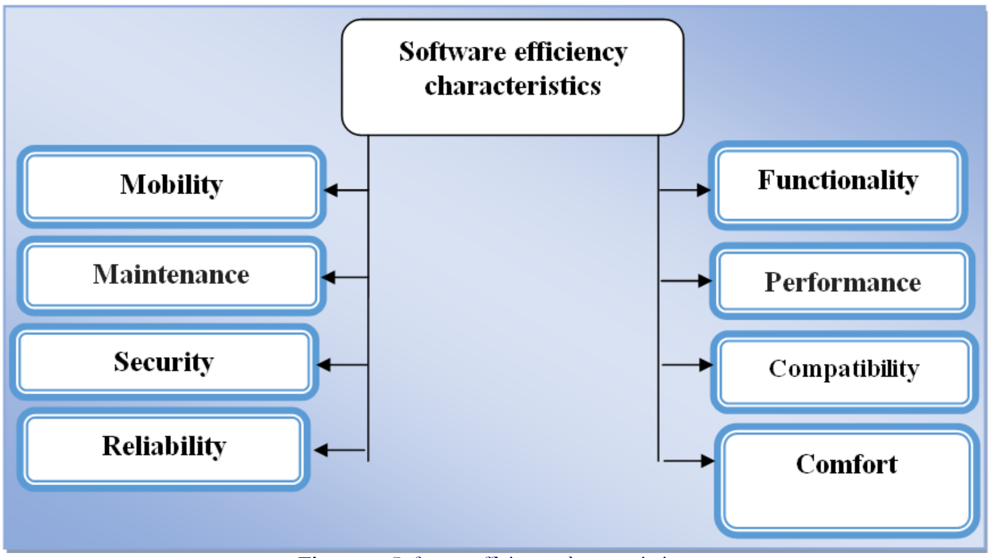

Figure-1. Software efficiency characteristics.

Note that biomimetic is an imitation model of the systems and elements in the nature to solve complex human problems. Living organisms have well-adapted structures and materials for natural selection and have developed over many years. The study of biomimetic technologies and their application in various fields can play an important role in the successful development of the economy.

Ant algorithm (Ant Colony Optimization, ACO) is one of the most effective polynomial algorithms for finding approximate solutions, and it also solves similar route search problems in graphs. The essence of the concept is to analyze, use, and meta-heuristically optimize the behavior of ants in search of paths to the food source of the colony. In 1992, Director of the Belgian Foundation for Scientific Research, Italian scientist Dr. Marco Dorigo proposed the first version of the algorithm [3, 4] to find the optimal path in the graph.

The imitation of the self-organization of ant colony is the basis of the ant optimization algorithm. Ant colony can be viewed as a multi-agent system, since each agent (ant) operates autonomously with very simple rules. Unlike the initial behavior of agents, the behavior of the entire system is surprisingly reasonable.

The self-organization of ants is the consequence of the interaction of the following five components [5]:

- Randomness.
- Multiplicity.
- Positive feedback.
- Negative feedback.
- Target function.

The ant was estimated to have its advantages and disadvantages.

1.1. Advantages

- Applying to some issues is relatively effective.
- The task can be solved by full search path for a small number of ants.
- Performs better than genetic algorithm.
- Quickly adapts to the changes in dynamic programs.
- Algorithm is simple and so on.

1.2. Disadvantages

- Theoretically difficult to analyze.

- Due to the result of the sequence of random solutions, the distribution of the probability changes during the iteration.
- Research is more experimental than the theory.
- Correspondence is guaranteed, however correspondence time is not specified.
- Application of additional methods, such as local search, is required.
- Strongly depends on the regulated settings, thus, it differs from the experiments.

2. STUDY OF RELATED WORKS

The ant algorithm is used to optimize various issues. In 1959, Pierre-Paul Grasse used the theory of Stigmergia (Stigmergia - Greek word meaning sign, etiquette and movement, work in) to explain the behavior of ant colony.

In 1983, Denebourg and his colleagues analyzed the collective behavior of ants [6].

In 1988, Mason and Mandersky published an article on self-organization among ants [7].

In 1989, the idea of an ant colony algorithm was proposed by Arona, Gossa, Denerborg and Pastelesas on the study "Collective Behavior of the Argentine Ants" [8].

In 1989, an ant foraging model was implemented by Jeblingcom and his colleagues [9].

In 1992, M. Dorigo proposed the concept of "ant system" in his doctoral thesis.

Another work [10] focus on the studies devoted to the increased interest in wind energy as a source of electric energy with minimal environmental impact, including the wind turbines, and the development of aerodynamic structures. The main goal of this work is to develop and improve the management strategies to achieve maximum strength using these methods. The Ant Colony Optimization (ACO) algorithm is used to determine the optimal control parameters for fast control. Wu, et al. [11] employs an ant algorithm to enhance the safety and efficiency of aircraft control.

Currently, the organizations specialized in telecommunication software development are attempting to provide the customers with high-quality software in the expected timeframe to remain competitive in the market. State Transition Testing Approach for wireless Networks using Ant Colony Optimization is applied in Mahlous, et al. [12]. Expected result was achieved in terms of situation and transition.

In work [13] authors adopted a swarm intelligent algorithm, Ant Colony Optimization (ACO), to solve the scheduling optimization of processes with two business objectives.

The great functionalities of object-orientation not only improve the developed software but also introduce a wide variety of problems in testing. In this paper [14] authors presenting a novel nature-inspired algorithm, AntLion Optimization (ALO) to generate optimal test paths for object-oriented software.

In this paper [15] an intelligent traffic engineering technique for a video surveillance system over SDN is proposed.

Ants belong to a group of social insects. They are able to handle the complex dynamic tasks of collaborative work, which cannot be implemented separately in separate environments without collective system, supervision and coordination.

In the real world, the ants travel randomly and return to their colony by extracting pheromones (odorants) on their paths to find food. If other ants find those paths, they will likely follow the same path. If the ant eventually finds a food source, it strengthens the path on the way back to follow this chain. Over time, pheromone begins to evaporate, reducing its effect along the way. Therefore, it is required to reach the target on time, and when the ant returns, the pheromone will evaporate more. For comparison, the transition to the shortest path will be faster, and as a result, the density of pheromones will remain high. Evaporation of pheromone will also affect the optimal decision. If the pheromone was not evaporated, then the first path would be the best. In this case, research on spatial decision-making would be limited. Thus, when an ant finds its way from the colony to the food source, the other ants will probably follow that path.

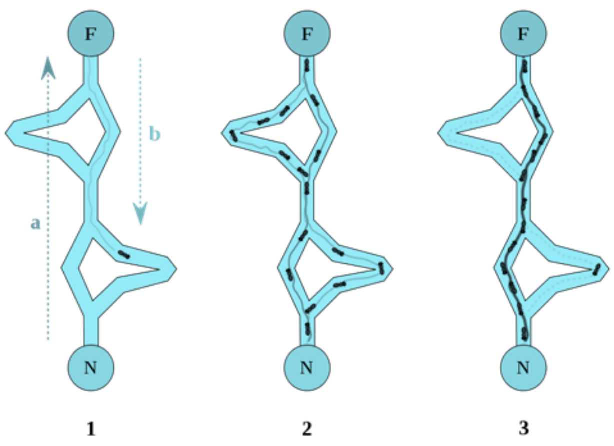

Figure-2. Behavior of ants searching for food.

1. The first ant finds a food source (F) in any way, and then returns to the nest (N), leaving behind a path from the pheromones.
2. Then the ants choose one of four possible paths, then strengthen it and make it attractive.
3. Ants choose the shortest route, since pheromones from longer paths evaporate faster.

The idea is that the ant reaches the food source from the colony through the shortest path.

1. The first ant finds the food source (F) through any paths (a) and leaves a route of pheromone (b) and then returns to the nest (N).
2. Ants then choose one of four possible ways and then make it sustainable.
3. Ants choose the shortest path as the pheromone evaporates faster on long paths.

Biologists emphasize that ants generally use the shortest path out of the two un similar paths from the colony to the food source [16].

The model of ant behavior can be explained as follows:

- Ants (called "Blitz") unintentionally go through a colony.
- If it finds a food source, then it goes back to the nest leaving pheromone on the path.
- The pheromone attracts other ants that are nearby, and they are most probably to follow this path.
- As a result, a long path will disappear due to the evaporation of the pheromone.
- If it finds a food source, then it goes back to the nest leaving pheromone on the path.
- On the way back to the nest, they will strengthen the pheromone path.
- If there are 2 routes, the ants will have to choose shorter one during that time period, and more ants will have chance to go through shorter path.
- Shorter route will be more effective.
- Eventually, the pheromone will disappear on longer paths.

Elite ant system. "Elite ants" differ for their total number. According to the results of each period of the algorithm, elite ants choose the best routes which leads to an increase of pheromone levels.

Max-Min ant system (MMAS). Boundary conditions are added to the amount of pheromone (T_{max}, T_{min}). The pheromone is delayed only in the best ways in the global repeat. All ribs, are initialized by the value T_{max}.

Ranked ant system (Asrank). All solutions are ranked according to the degree of validity of the ants. The amount of delayed pheromone for each solution produces more pheromones for more suitable solutions.

Long-term orthogonal ant colony (COAC). The mechanism of COAC pheromone cancellation enables the ants to jointly search for solutions. Using an orthogonal method, the ants can perform rapid and efficient exploration of their chosen areas with enhanced global search capabilities and accuracy [17].

Ant algorithm has already performed good accuracy for many complex combinations: mobile vendor issues, truck route optimization, graphic coloring, network graphics optimization, and so forth.

The idea of the ant algorithm is to model the behavior of ants.

Ant algorithm is particularly effective for on-line process optimization in distributed non-stationary systems, such as the traffic in telecommunication networks and so on. The application of ant algorithm in software optimization is viewed below.

The main steps of the algorithm include:

- Ants organizing.
- Solution search.
- Pheromone renewal.
- Additional works (optional).

3. PROBLEM STATEMENT

Each operator in software is viewed as an ant. Suitable operators form an ant colony. For example, cycle, conditional, transition, and so on. Two ants out of the total ants have to reach the food beyond the barrier Figure 3. As it moves, each ant produces little pheromone as a marker.

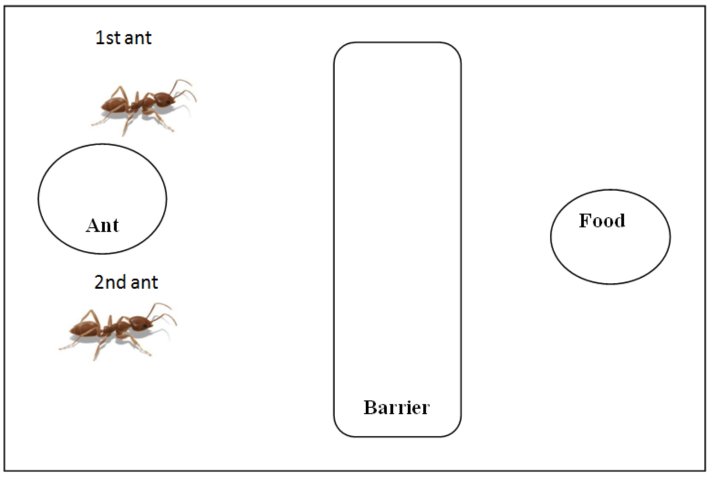

Figure-3. Short movement of ants to reach the food.

Each ant here chooses its own path. The first ant chooses the path above, and the second one chooses the path below. Since the path below is two times shorter than another, the second ant will reach the target in time. At that moment, the first ant will pass half the way.

When an ant reaches the food, it takes some of the foods and goes back to the ant nest along the same path. As each ant moves, little pheromone remains on the path. For the first ant, during this period of time, the path is covered with pheromones only once. During this period of time, the first ant goes back, whereas the second ant manages to eat once and go back. In this case, the pheromone concentration on the path below will be twice higher than the one above. Therefore, for the second time, the first ant will choose the path below, because the concentration of pheromone is higher there.

According to successfully selected heuristics, the repeated use of the algorithm often performs optimal results.

Biologists estimate that, as a rule, ants often choose the shortest path between the two paths from a colony to a food source for their own choice.

Note that transport logistics is one of the most obvious and popular examples for the application of ant algorithms. The tasks of transport logistics also include the task of commas voyageur and the routing of road transport [18].

4. APPLICATION OF ANT ALGORITHM TO SOFTWARE

The ant algorithm is used to optimize various tasks. In this work, it helps to improve and optimize software efficiency. The Dorigo method was used to solve this problem [4]. As it is mentioned above, the software code contains operators. Switching from one operator to another for a specific purpose can be through a long way, resulting in slower performance of the software. An appropriate group of operators (period, conditional, etc.) is viewed as an ant colony as follows.

i = 1, where n shows the i-th ant colony.

The work schedule begins with the placement of the ants, and then the movement of the ants starts - the probability of transition to the direction is calculated by a certain formula:

$$P_i = \frac{l_i^q f_i^p}{\sum_{k=0}^{N} l_k^q f_k^p} \tag{1}$$

P_i probable transition to the path *i*.

l_i - length of the i-th path.

f_i – amount of pheromone in the i-th path.

q – measure that determines the "greediness" of the algorithm.

p –measure that determines the "colony" of the algorithm.

"Greedy algorithm" makes optimal decisions at each stage assuming that the final solution is optimal.

The application of the ant algorithm to the software is viewed below. The algorithm consists of several steps.

Step1. Operators are viewed as ants. An ant is placed on each node, i.e. the number of operators is equal to the number of ants. This process involves the creation of "memory" blocks where different parameters must be stored. These include the nodes of the graph, which the ant (operator) visits, and the length of the path.

Step2. The initial pheromone is first freely applied to each section of the path; the parameters of the algorithm are defined: α (degree of 'greediness'), β (effect of pheromone on the search for optimal solution), P (pheromone evaporation rate), Q (pheromone distribution on the path).

Step3. Iteration process begins for each ant. All possible options for the path are selected for the current ant using formula (1). In this case, several probabilities are obtained for each operator.

Step4. The visited operator is placed in the "memory" of the ant as the calculated distance.

Step5. Once the iteration is completed for each ant, the amount of the pheromone in each section of the paths is recalculated. It returns to step 2 at the end of the iteration.

Step6. The saved values follow the search for minimum distance and the sequence rules.

Obviously, the software has a code. Here the operators are written in a certain order and they are implemented in accordance with the rule. The task of software optimization is often ignored which can lead to slow performance of the program. An ant algorithm is used to achieve results through the shortest way and to speed up performance. The application of the ant algorithm to the software includes several steps. This is illustrated in the form of a graph Figure 4.

Step1. Let's define the probability of transiting to one of the peaks. (q = 1 and p = 1) are the parameters and will not change in the algorithm throughout the implementation. Let's describe each software operator as an ant and use a key operator. The value of pheromone in all the nodes of the graph is taken 0.6. This value may also vary for each node. Table 1 show the path length and pheromone values. Here, 8 nodes are used and transitions to each node are shown.

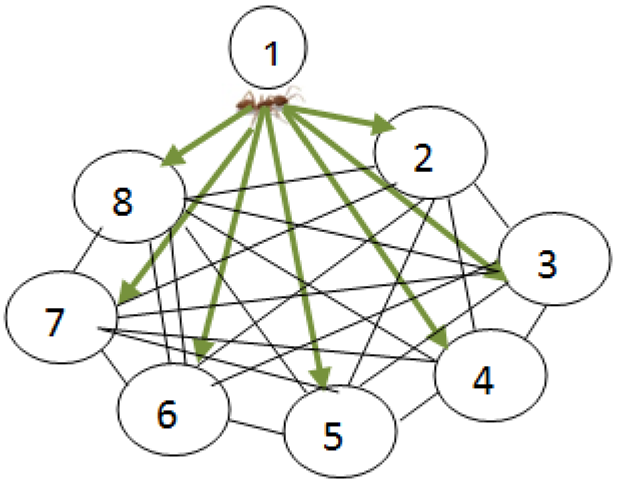

Figure-4. Locating the ants (operators) on the nodes.

Table-1. Road length and price of pheromone.

Path	Path length	Pheromones
1→ 2	1	0.6
1→ 3	2	0.6
1→4	3	0.6
1→5	4	0.6
1→6	5	0.6
1→7	3	0.6
1→8	1	0.6
2→3	4	0.6
2→4	3	0.6
2→5	5	0.6
2→6	4	0.6
2→7	5	0.6
2→8	2	0.6
3→4	1	0.6
3→5	2	0.6
3→8	6	0.6
3→7	4	0.6
3→6	5	0.6
4→5	1	0.6
4→8	3	0.6
4→7	5	0.6
4→6	2	0.6
5→6	1	0.6
5→8	4	0.6
5→7	3	0.6
6→7	4	0.6
6→8	2	0.6
7→8	3	0.6

In accordance with these values, the appropriate values are calculated being placed in formula (1).

$$P_{11} = \frac{\frac{1}{1} * 0,6}{\frac{1}{1} * 0,6 + \frac{1}{2} * 0,6 + \frac{1}{3} * 0,6 + \frac{1}{4} * 0,6 + \frac{1}{5} * 0,6 + \frac{1}{3} * 0,6 + \frac{1}{1} * 0,6} = 0,28$$

$$P_{12} = \frac{\frac{1}{2} * 0,6}{\frac{1}{1} * 0,6 + \frac{1}{2} * 0,6 + \frac{1}{3} * 0,6 + \frac{1}{4} * 0,6 + \frac{1}{5} * 0,6 + \frac{1}{3} * 0,6 + \frac{1}{1} * 0,6} = 0,14$$

$$P_{13} = \frac{\frac{1}{3}*0,6}{\frac{1}{1}*0,6+\frac{1}{2}*0,6+\frac{1}{3}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{3}*0,6+\frac{1}{1}*0,6} = 0,09$$

$$P_{14} = \frac{\frac{1}{4}*0,6}{\frac{1}{1}*0,6+\frac{1}{2}*0,6+\frac{1}{3}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{3}*0,6+\frac{1}{1}*0,6} = 0,07$$

$$P_{15} = \frac{\frac{1}{5}*0,6}{\frac{1}{1}*0,6+\frac{1}{2}*0,6+\frac{1}{3}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{3}*0,6+\frac{1}{1}*0,6} = 0,06$$

$$P_{16} = \frac{\frac{1}{3}*0,6}{\frac{1}{1}*0,6+\frac{1}{2}*0,6+\frac{1}{3}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{3}*0,6+\frac{1}{1}*0,6} = 0,09$$

$$P_{17} = \frac{\frac{1}{1}*0,6}{\frac{1}{1}*0,6+\frac{1}{2}*0,6+\frac{1}{3}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{3}*0,6+\frac{1}{1}*0,6} = 0,28$$

In this step, the right transition from one operator to others has to be found. As seen from the values, the probability value varies within 0, 1.

P_{11} – [0, 0.3]

P_{12} – [0, 0.2]

P_{13} – [0, 0.1]

P_{14} – [0, 0.1]

P_{15} – [0, 0.1]

P_{16} – [0, 0.1]

P_{17} – [0, 0.3]

Step 2. All transitions of the 2nd node are assessed.

$$P_{23} = \frac{\frac{1}{4}*0,6}{\frac{1}{4}*0,6+\frac{1}{3}*0,6+\frac{1}{5}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{2}*0,6} = 0,14$$

$$P_{24} = \frac{\frac{1}{3}*0,6}{\frac{1}{4}*0,6+\frac{1}{3}*0,6+\frac{1}{5}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{2}*0,6} = 0,19$$

$$P_{25} = \frac{\frac{1}{5}*0,6}{\frac{1}{4}*0,6+\frac{1}{3}*0,6+\frac{1}{5}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{2}*0,6} = 0,12$$

$$P_{26} = \frac{\frac{1}{4}*0,6}{\frac{1}{4}*0,6+\frac{1}{3}*0,6+\frac{1}{5}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{2}*0,6} = 0,14$$

$$P_{27} = \frac{\frac{1}{5}*0,6}{\frac{1}{4}*0,6+\frac{1}{3}*0,6+\frac{1}{5}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{2}*0,6} = 0,12$$

$$P_{28} = \frac{\frac{1}{2}*0,6}{\frac{1}{4}*0,6+\frac{1}{3}*0,6+\frac{1}{5}*0,6+\frac{1}{4}*0,6+\frac{1}{5}*0,6+\frac{1}{2}*0,6} = 0,28$$

P_{23} – [0, 0.2]

P_{24} – [0, 0.3]

P_{25} – [0, 0.2]

P_{26} – [0, 0.2]

P_{27} – [0, 0.2]

P_{28} – [0, 0.3]

As seen from these values, their probability value also varies within 0.1.

Step 3. All transitions of the 3rd node is assessed.

$$P_{34} = \frac{\frac{1}{1} * 0,6}{1/1 * 0,6 + 1/2 * 0,6 + 1/6 * 0,6 + 1/4 * 0,6 + 1/5 * 0,6} = 0,47$$

$$P_{35} = \frac{\frac{1}{2} * 0,6}{1/1 * 0,6 + 1/2 * 0,6 + 1/6 * 0,6 + 1/4 * 0,6 + 1/5 * 0,6} = 0,24$$

$$P_{38} = \frac{\frac{1}{6} * 0,6}{1/1 * 0,6 + 1/2 * 0,6 + 1/6 * 0,6 + 1/4 * 0,6 + 1/5 * 0,6} = 0,08$$

$$P_{37} = \frac{\frac{1}{4} * 0,6}{1/1 * 0,6 + 1/2 * 0,6 + 1/6 * 0,6 + 1/4 * 0,6 + 1/5 * 0,6} = 0,12$$

$$P_{36} = \frac{\frac{1}{5} * 0,6}{1/1 * 0,6 + 1/2 * 0,6 + 1/6 * 0,6 + 1/4 * 0,6 + 1/5 * 0,6} = 0,09$$

P_{34} – [0, 0.5]

P_{35} – [0, 0.2]

P_{38} – [0, 0.1]

P_{37} – [0, 0.2]

P_{36} – [0, 0.1]

Step 4. All transitions of the 4th node are assessed.

$$P_{45} = \frac{\frac{1}{1} * 0,6}{\frac{1}{1} * 0,6 + \frac{1}{3} * 0,6 + \frac{1}{5} * 0,6 + \frac{1}{2} * 0,6} = 0,48$$

$$P_{48} = \frac{\frac{1}{3} * 0,6}{\frac{1}{1} * 0,6 + \frac{1}{3} * 0,6 + \frac{1}{5} * 0,6 + \frac{1}{2} * 0,6} = 0,24$$

$$P_{47} = \frac{\frac{1}{5} * 0,6}{\frac{1}{1} * 0,6 + \frac{1}{3} * 0,6 + \frac{1}{5} * 0,6 + \frac{1}{2} * 0,6} = 0,16$$

$$P_{46} = \frac{\frac{1}{2} * 0,6}{\frac{1}{1} * 0,6 + \frac{1}{3} * 0,6 + \frac{1}{5} * 0,6 + \frac{1}{2} * 0,6} = 0,12$$

P_{45} – [0, 0,5]

P_{48} – [0, 0.3]

P_{47} – [0, 0.2]

P_{46} – [0, 0.2]

Step 5. All transitions of the 5th node is assessed.

$$P_{56} == \frac{\frac{1}{1}*0,6}{\frac{1}{1}*0,6+\frac{1}{4}*0,6+\frac{1}{3}*0,6} = 0,63$$

$$P_{58} == \frac{\frac{1}{4}*0,6}{\frac{1}{1}*0,6+\frac{1}{4}*0,6+\frac{1}{3}*0,6} = 0,16$$

$$P_{57} == \frac{\frac{1}{3}*0,6}{\frac{1}{1}*0,6+\frac{1}{4}*0,6+\frac{1}{3}*0,6} = 0,21$$

P_{55} – [0, 0,7]

P_{58} – [0, 0.2]

P_{57} – [0, 0.3]

Step 6. All transitions of the 6nd node are assessed.

$$P_{67} = \frac{\frac{1}{4}*0,6}{\frac{1}{4}*0,6+\frac{1}{2}*0,6} = 0,33$$

$$P_{68} = \frac{\frac{1}{2}*0,6}{\frac{1}{4}*0,6+\frac{1}{2}*0,6} = 0,33$$

P_{67} – [0, 0,4]

P_{68} – [0, 0.7]

Step 7. The 7th node will have one value.

$$P_{78} == \frac{\frac{1}{3}*0,6}{\frac{1}{3}*0,6} = 1,0$$

P_{78}–[0, 1.0]

After all transitions, the pheromone is renewed.

Update of pheromones $t(i+1) = (1-p)*t(i) + Q/L_0$

Here p - regulates the evaporation rate of the pheromone.

Q - regulates the concentration of pheromone.

L_0– indicates the length of the path in a given section of the path.

$$\Delta T = \frac{Q}{D}$$

When Q = 1 and D = 1

The pheromone is renewed with the following formula:

$$\Delta T = \frac{1}{1} = 1 \qquad (2)$$

Here, 1-P denotes an evaporation coefficient, τ - previous value of pheromone, and $\Delta\tau$ – updated value of pheromone.

P indicates the evaporation rate of pheromone, if P> 1, evaporation increases, whereas, if

P <1, evaporation decreases.

If P = 0.6, then the pheromone to be evaporated after the iteration is recalculated.

Based on formula (2), the newly calculated values of the pheromone are shown in Table 2 and Figure 5.

Table-2. Calculated new values of pheromone.

Path	Path length	New values of pheromone
1→ 2	1	(1-0.6)*(0.6+1/1) =0.64
1→ 3	2	(1-0.6)*(0.6+1/2) =0.44
1→4	3	(1-0.6)*(0.6+1/3) =0.37
1→5	4	(1-0.6)*(0.6+1/4) =0.34
1→6	5	(1-0.6)*(0.6+1/5) =0.32
1→7	3	(1-0.6)*(0.6+1/3) =0.37
1→8	1	(1-0.6)*(0.6+1/1) =0.64
2→3	4	(1-0.6)*(0.6+1/4) =0.34
2→4	3	(1-0.6)*(0.6+1/3) =0.37
2→5	5	(1-0.6)*(0.6+1/5) =0.32
2→6	4	(1-0.6)*(0.6+1/4) =0.34
2→7	5	(1-0.6)*(0.6+1/5) =0.32
2→8	2	(1-0.6)*(0.6+1/2) =0.44
3→4	1	(1-0.6)*(0.6+1/1) =0.64
3→5	2	(1-0.6)*(0.6+1/2) =0.44
3→8	6	(1-0.6)*(0.6+1/6) =0.31
3→7	4	(1-0.6)*(0.6+1/4) =0.34
3→6	5	(1-0.6)*(0.6+1/5) =0.32
4→5	1	(1-0.6)*(0.6+1/1) =0.64
4→8	3	(1-0.6)*(0.6+1/3) =0,37
4→7	5	(1-0.6)*(0.6+1/5) =0.32
4→6	2	(1-0.6)*(0.6+1/2) =0.44
5→6	1	(1-0.6)*(0.6+1/1) =0.64
5→8	4	(1-0.6)*(0.6+1/4) =0.34
5→7	3	(1-0.6)*(0.6+1/3) =0.37
6→7	4	(1-0.6)*(0.6+1/4) =0,34
6→8	2	(1-0.6)*(0.6+1/2) =0.44
7→8	3	(1-0.6)*(0.6+1/3) =0.37

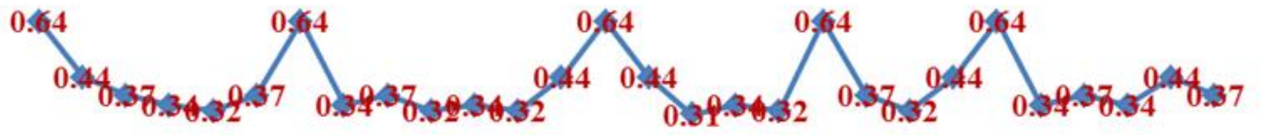

Figure-5. Calculated new values of pheromone.

5. CONCLUSION

The article used the ant algorithm proposed by Marco Dorigo to optimize software. Note that the number of ants in a colony may range from 30 to several million. Ant colony can be viewed as a multi-agent system, and each agent operates autonomously according to simple rules. In this article, each operator of a random program is viewed as an ant. Applying the ant algorithm, the shortest transition from one operator to others was found.

This algorithm can be improved to perform better results.

Funding: This study received no specific financial support.
Competing Interests: The author declares that there are no conflicts of interests regarding the publication of this paper.

REFERENCES

[1] X. S. Yang, Z. Cui, R. Xiao, A. Gandomi, and M. Karamanoglu, *Swarm intelligence and bio-inspired computation: Theory and applications*. Amsterdam: Elsevier, 2013.

[2] Systems and Software Engineering, "System and software quality models." Retrieved from https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en, 2011.

[3] A. Colorni, M. Dorigo, V. Maniezzo, and D. M. Politecnico, "Distributed optimization by ant colonies," in *Proc. Appeared in Proceedings of Ecal91 - European Conference on Artificial life, Paris*, 1991, pp. 134-142.

[4] M. Dorigo, "Optimization, learning and natural algorithms," PhD Thesis, Politecnico di Milano, 1992.

[5] Ant Colony Algorithms, "Retrieved from http://smart-blog.net/post/2359," 2018.

[6] J.-L. Deneubourg, J. M. Pasteels, and J.-C. Verhaeghe, "Probabilistic behaviour in ants: A strategy of errors?," *Journal of Theoretical Biology*, vol. 105, pp. 259-271, 1983. Available at: https://doi.org/10.1016/s0022-5193(83)80007-1.

[7] F. Moyson and B. Manderick, "The collective behaviour of Ants: An example of self-organization in massive parallelism," in *Proc. Actes de AAAI Spring Symposium on Parallel Models of Intelligence, Stanford, Californie*, 1988.

[8] S. Goss, S. Aron, J.-L. Deneubourg, and J. M. Pasteels, "Self-organized shortcuts in the argentine ant," *Naturwissenschaften*, vol. 76, pp. 579-581, 1989. Available at: https://doi.org/10.1007/bf00462870.

[9] M. Ebling, M. Di Loreto, Presley, M. F. Wieland, and D. Jefferson, "An ant foraging model implemented on the time warp operating system," in *Proc. of the SCS Multiconference on Distributed Simulation*, 1989.

[10] Y. Mokhtari and D. Rekioua, "High performance of maximum power point tracking using ant colony algorithm in wind turbine," *Renewable Energy*, vol. 126, pp. 1055-1063, 2018. Available at: https://doi.org/10.1016/j.renene.2018.03.049.

[11] J. Wu, M. Dong, K. Ota, J. Li, and Z. Guan, "Big data analysis-based secure cluster management for optimized control plane in software-defined networks," *IEEE Transactions on Network and Service Management*, vol. 15, pp. 27-38, 2018. Available at: https://doi.org/10.1109/tnsm.2018.2799000.

[12] A. R. Mahlous, A. Zarrad, and T. Alotaibi, "State transition testing approach for Ad hoc networks using ant colony optimization," *International Journal of Advanced Computer Science and Applications*, vol. 9, pp. 146-155, 2018. Available at: https://doi.org/10.14569/ijacsa.2018.090621.

[13] L. Tran, H. Huynh, and H. Akhtar, "Ant colony optimization algorithm for maintenance, repair and overhaul scheduling optimization in the context of industrie 4.0," *Aplied Sciences-Basel*, vol. 9, pp. 1-13, 2019. Available at: https://doi.org/10.3390/app9224815.

[14] R. Sharma and A. Saha, "Ant Lion optimizer for state based object oriented testing," *Journal of Information and Optimization Sciences*, vol. 40, pp. 219-232, 2019. Available at: https://doi.org/10.1080/02522667.2019.1578085.

[15] R. Mohammadi, R. Javidan, and M. Keshtgari, "An intelligent traffic engineering method for video surveillance systems over software defined networks using ant colony optimisation," *International Journal of Bio-Inspired Computation*, vol. 12, pp. 173-185, 2018. Available at: https://doi.org/10.1504/ijbic.2018.094625.

[16] J.-L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels, "The self-organizing exploratory pattern of the argentine ant," *Journal of Insect Behavior*, vol. 3, pp. 159-168, 1990. Available at: https://doi.org/10.1007/bf01417909.

[17] X.-M. Hu, J. Zhang, and Y. Li, "Orthogonal methods based ant colony search for solving continuous optimization problems," *Journal of Computer Science and Technology*, vol. 23, pp. 2-18, 2008. Available at: https://doi.org/10.1007/s11390-008-9111-5.

[18] A. A. I. Kazharov and V. M. Kureychik, "Ant algorithms for solving transport problems," *Proceedings of the Russian Academy of Sciences Theory and Control Systems*, vol. 1, pp. 32-45, 2010.