TECHNICAL TRANSACTIONS | **CZASOPISMO TECHNICZNE**
ELECTRICAL ENGINEERING | ELEKTROTECHNIKA

2-E/2015

KRZYSZTOF SCHIFF*

AN IMPROVEMENT OF THE ANT ALGORITHM FOR THE MAXIMUM CLIQUE PROBLEM

UDOSKONALENIE ALGORYTMU MRÓWKOWEGO DO WYZNACZANIA KLIKI MAKSYMALNEJ

A b s t r a c t

The maximum clique problem is a very well-known NP-complete problem and for such a problem, meta-heuristic algorithms have been developed which ant algorithms belongs to. There are many algorithms including ant algorithms that have been elaborated for this problem. In this paper, a new dynamic function of selecting with a new improvement procedure in order to get a larger size of clique for the ant algorithm is presented and this search for the maximum clique in graph is compared to the best ant algorithms that are already known.

Keywords: *maximum clique*, *ant algorithm*, *desirability function*

S t r e s z c z e n i e

Problem kliki maksymalnej przynależy do klasy problemów NP-zupełnych i dla takich problemów opracowuje się obecnie algorytmy metaheurystyczne, do których zaliczają się algorytmy mrówkowe. W niniejszym artykule prezentowany jest algorytm mrówkowy z dynamiczną funkcją wyboru wierzchołków włączanych do tworzonej kliki przez każdą mrówkę wraz z procedurą poprawy wymiaru otrzymanej kliki poprzez wymianę wierzchołków, a otrzymany algorytm został porównany z innymi już dotychczas opublikowanymi.

Słowa kluczowe: *klika maksymalna*, *algorytmy mrówkowe*, *funkcja pożądania*

DOI: 10.4467/2353737XCT.15.098.3930

* Ph.D. Eng. Krzysztof Schiff, Department of Automatic Control and Information Technology, Faculty of Electrical and Computer Engineering, Cracow University of Technology.

1. Introduction

The maximum clique problem has been proven to be a NP-complete problem, this is why there are many meta-heuristic algorithms for this problem to which ant algorithms belong. Ant algorithms are very well suited to solving combinatorial optimization problems [1]. Ant algorithms have already been used to find a maximum clique in graphs [1–6]. The first ant algorithm for the maximum clique problem was presented in [3], which was later improved and discussed in [4]. A distributed version of an ant algorithm was formulated in [2]. This paper presents a new ant algorithm in which there is a new dynamic heuristic and a new procedure for getting a better solution by each ant – this means obtaining a larger maximal clique size.

2. The maximum clique problem

Let $G = (V, E)$ be a graph with a set of vertices V and a set of edges E. A clique C is a subset of set V in which each two vertices (v_i, v_j) are linked together by an edge e_{ij}. A maximal clique is a clique which is not included in another clique. The size of clique C is equal to a number of vertices in the subset C. A maximum clique is the maximal clique with the greatest number of vertices. The vertex degree d_i is the number of edges adjacent to this vertex i. A graph with an almost equal degree of all vertices is a graph in which almost all vertices have the equal vertex degree $d_{v1} \cong d_{vi} \cong \ldots \cong d_{vn}$, $v_i \in V$, such a graph is shown in Fig. 1.

The maximum clique problem is a NP-complete problem – this is why there are many elaborated meta-algorithms for this problem. A vertex degree is the main information which is used by all meta-heuristic algorithms for the maximum clique problem. During the creation of a maximum clique, a vertex is selected depending upon all vertex degrees. The way in which these selected vertices are included into the maximum clique C is a very important factor on which the size of maximal clique depends. When there are many vertices with the equal or almost equal vertex degree, there is no useful information for vertex selection and the way in which vertices are included into the maximum clique is not right. The selection

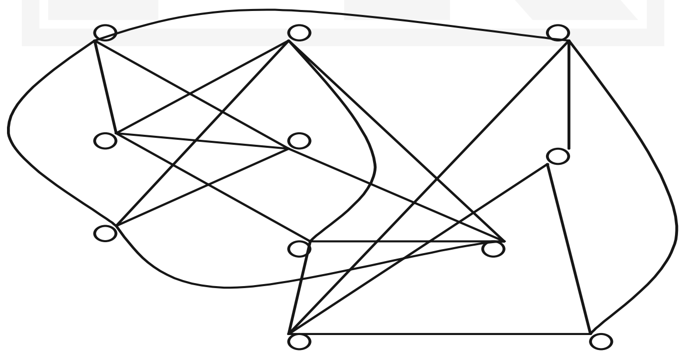

Fig. 1. A graph with almost equal degrees of all vertices

probability formula in ant algorithms includes a vertex degree as useful information, but it is very difficult for ants to make a selection in a correct manner and this is why ants have some difficulty in finding the maximum clique in a graph with almost the same vertex degree of all vertices. Since ants make their vertex selection depending upon the probability formula, they cannot ensure that this is the right way to obtain the maximum clique. If this probability formula allows an improved selection of vertices, then an obtained maximal clique would be closer to the maximum clique. It is therefore very important that the probability formula allows better distinguishing between vertices and better selection of one vertex from others – this is most appropriate in order to ensure an appropriate selection method for the creation of the maximum clique. The probability formula allows ants to better distinguish vertices when the information included in this formula is more precise; therefore, it is very important to formulate such a pattern. In this paper, apart from the more precise information included in a selection formula, an improvement procedure which has been elaborated in order to get a bigger size of received clique is included in the ant algorithm.

3. The improvement of the clique size

When a clique of some size has been found, it is unclear whether this clique is the maximal clique of the graph. In such a situation, we can try to enlarge the size of this clique by using a simple procedure, the pseudo code of which is shown as algorithm 1. We include a vertex in a clique which is not included in this clique and remove from the clique those vertices which are not connected to this new vertex by an edge; afterwards, into this clique additional vertices can be included which can create a new clique of a larger size. As an example: there is a clique of size 6 which consist of vertices 1, 2, 3, 4, 5, 6 in Fig. 2. When we include into this clique vertex 7, we can remove vertices 5 and 6, and afterwards, we can include vertices 8 and 9 – in this way, we can receive a clique with a size of 7 since the clique consist of vertices 1, 2, 3, 4, 7, 8, 9 and this size is larger than the size of the precedent clique, which has a size equal to 6. This improvement procedure is shown as algorithm 1.

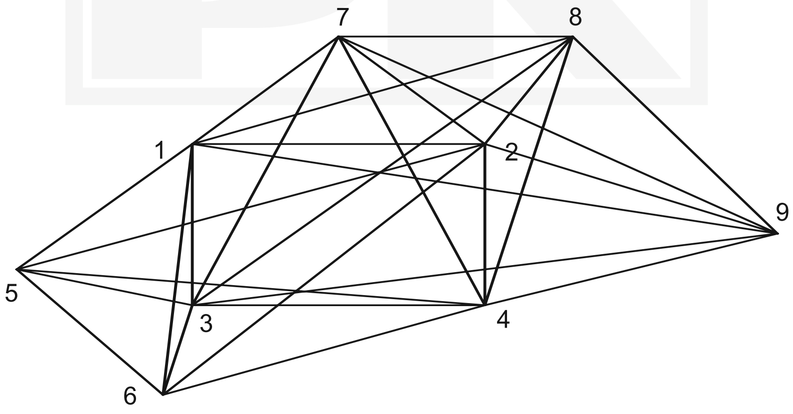

Fig. 2. Example showing how the size of a clique can be enlarged

Algorithm 1

The simple procedure for improvement of clique size

```
While (if a size of clique has been enlarged
    Begin
        For all vertices which are not included into the clique C
            Include a vertex v into the clique S
            Remove from the set S those vertices which are not adjacent to vertex v
            Check if the size of clique S is bigger than the size of the precedent clique C
            If not, return to the precedent clique C
            If yes, repeat a loop while
            If all vertices have been checked and a larger size of clique has been not found
                finish a while loop
End.
```

4. The ant method

Ants search for the best solution to encountered problems. In order to find a such solution, ants communicate with each other by means of a pheromone t. At the beginning of the general ant algorithm, which is presented as algorithm 2, on all elements $i \in M$ a [???], the maximal quantity of pheromone is deposited $t(i) = t_{max}$. The set M is the set of elements i, which can constitute a solution S to certain optimization problem. In the case of the maximum clique problem, the set M is the set of all vertices which are not included into the solution set S and which are connected by edges with all vertices which are already included into the solution set S. The general ant algorithm consists of two main loops: the first loop is connected with the number of cycles and the second loop is connected with the number of ants. With each repetition of the first loop, all repetitions of the second loop have to be performed. The best solution S_b, which was found by all ants in one cycle, is compared to the best solution S_{best}, which was found by ants in a previous cycle. In each cycle, an evaporation mechanism is also used: some of the pheromone is evaporated with the rate r from all elements $i \in M$. In each cycle, an additional quantity of pheromone is also deposited dt on these elements i which constitute a solution S_b. When all loops have been performed, the best solution is returned. At the beginning of each inner loop, a start point is prepared for each ant. From such a start point, each ant begins to create a solution to the optimization problem and then in the *while* loop, each ant selects a next element j with the probability $p(j)$ and puts it to the solution set S. The probability $p(j)$ can be expressed by the formula:

$$p(j) = \frac{t_j n_j}{\Sigma_j (t_j n_j)} \tag{1}$$

where:

- t_j – a quantity of pheromone deposited on element j, $(1 \le j \le \max)$,
- max – the maximum number of available elements from which the selection can be made,
- n_j – a heuristic, this is the desirability of including an element j into the solution set S.

Algorithm 2

The general ant algorithm

for all $i \in M$: $t(i) = t_{max}$
for all cycles
  for all ants
    make a start point
    while (a solution S is not completed) **do**
      check which elements are available to be selected, put them into the set A
      select a next element from the set A with probability $p(j)$
      add a selected element to the S
    save in the S_b a best solution, which has been found by all ants in a cycle
  if S_b is better than S_{best} then save the S_b as the S_{best} : $S_{best} = S_b$
  for all i: $t(i) = t(i) + r* t$
  $dt = f(S_b)$
  if $i \in S_b$ then $t[i] = t(i) + dt$
return S_{best}

Such a selection as mentioned above can be made only from set A, so from these elements i which are available and which can constitute at this moment of algorithm action a solution to the optimization problem. When any element is added to the solution set S, not all elements from the set A still satisfy constraints and from the previous set A, the new set A is created by only including those elements from the previous set A which satisfy constraints into this set A. In the case of the maximal clique problem, when a vertex i is included into the set S, because this vertex i is not connected together with some vertices from set A, set A should be updated in a such way that all vertices which are not connected together with a vertex i should be removed from set A. Only these vertices which can in future be included into the solution set S should be inside set A and only these vertices are available for the selection.

5. The improvement procedure in the ant algorithm

The structure of the proposed algorithm is the same as the structure of the ant algorithm which was discussed in [3], but there is a little difference between these two algorithms since in the ant algorithm which is proposed in this paper, there is an improvement procedure, which was described in section 2 and a desirability function n – both of these are new elements to the ant algorithm which do not occur in [3, 4, 6]. The procedure of improvement is used after a clique has been constructed by each ant and a new formula for the selection probability used by each ant during clique construction is expressed as follow:

$$p(v_i) = \frac{[t_{v_i} n_{v_i}]^{\alpha}}{\Sigma_{v_i \varepsilon \mathrm{Candidates}} [t_{v_i} n_{v_i}]^{\alpha}} \tag{2}$$

where:

$n(v_i)$ – a heuristic information, desirability of vertex v_i,

$$n(v_i) = \frac{D_{v_i}}{m}, \tag{3}$$

m – the number of vertices,

D_{vi} – the vertex degree in Candidates,

Candidates – the set A of available vertices or it could be a graph structure which is built of vertices from set A and from edges which connect together vertices from the set A, degree D_{vi} is computed in accordance with formula:

$$D_{v_i} = \Sigma_{k\varepsilon \text{Candidates}} d_{ik} \tag{4}$$

$d_{ij} = 1$ when $e_{ij} \in E$ else $d_{ij} = 0$, D_{vi} is not constant and is varying during algorithm action in accordance with contents of the set A.

6. Experiments

The first conducted experiment concerns average maximum cliques sizes obtained using the algorithm presented in [4] and the algorithm in which only a new selection formula with desirability function n is used, as was presented in section 4. The first algorithm is called the ACO algorithm, the second algorithm is called the ACOwith$n$ algorithm. Both algorithms were tested for different values of graph density $q\{0.97, 0.974, 0.978, 0.982, 0.986, 0.99, 0.994, 0.998\}$.Tests were conducted for graphs with a number of vertices equal to $n = 200$, a number of cycles equal to $lc = 200$, a number of ants equal to $lm = 30$ and an evaporation rate equal to 0.997. An average maximum clique size from 100 measurements and differences in these sizes received for both algorithms have been shown in Table 1 and in Fig. 3. As can be seen, a new selection formula with desirability function n allows receiving a larger clique size in comparison to the case when this desirability function n is not used in a selection formula.

The second conducted experiment concerns average maximum cliques sizes obtained using the first algorithm in which only a new selection formula with desirability function n is used as is presented in section 4 and the second algorithm, in which apart from a new selection formula with a desirability function n also an improvement procedure, which is described in section 2, is used. The first algorithm is called the ACOwith$n$ algorithm, the second is called the IACOwith$n$ algorithm. Both algorithms were tested for different values of graph density $q\{0.97, 0.974, 0.978, 0.982, 0.986, 0.99, 0.994, 0.998\}$. Tests were conducted for graphs with a number of vertices equal to $n = 200$, a number of cycles equal to $lc = 200$, a number of ants equal to $lm = 30$ and an evaporation rate equal to 0.997. An average maximum clique size from 100 measurements and differences in these sizes received for both algorithms have been shown in Table 2 and in Fig. 4. As can be seen, a new selection formula with desirability function n and improvement procedure allows receiving a larger clique size in comparison to the case when only a new selection formula with desirability function n without the improvement procedure is used.

Table 1

Average size, difference in size for different *q* and 200 vertices

Q	0.998	0.994	0.99	0.986	0.982	0.978	0.974	0.97
ACO	171.81	139.79	121.2	106.12	95.85	87.65	80.24	74.96
ACOwith*n*	172.20	140.33	121.8	107.00	96.60	88.43	81.87	75.63
ACOwith*n*-ACO	0.39	0.54	0.6	0.88	0.75	0.78	1.63	0.67

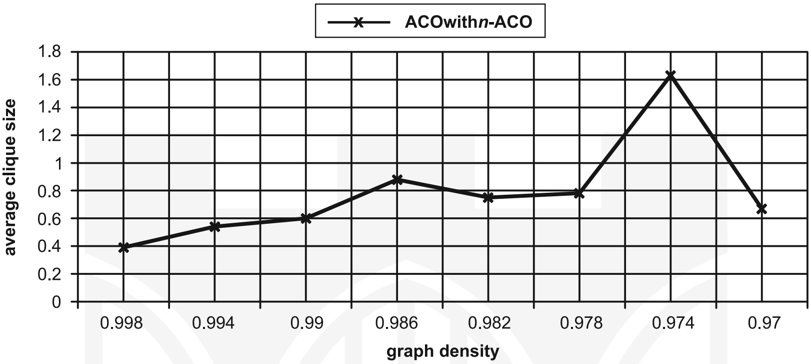

Fig. 3. Difference in average size for different *q* and 200 vertices

Table 2

Average size, difference in size for different *q* and 200 vertices

Q	0.998	0.994	0.99	0.986	0.982	0.978	0.974	0.97
ACOwith*n*	172.20	140.33	121.80	107.00	96.60	88.43	81.87	75.63
IACOwith*n*	172.29	140.94	123.41	109.98	100.23	92.58	86.12	80.91
IACOwith*n*-ACOwith*n*	0.29	0.61	1.61	2.98	3.63	4.15	4.25	5.28

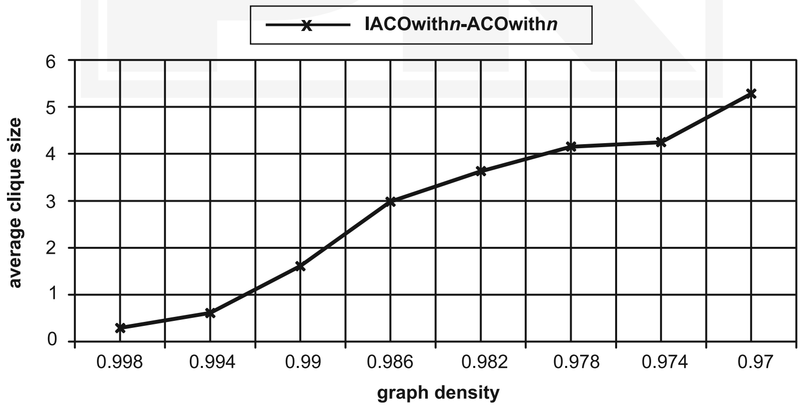

Fig. 4. Difference in average size for different *q* and 200 vertices

The last experiment concerns average maximum cliques sizes obtained using the first algorithm, which is described in [6]. This uses an improvement procedure, which is described in section 2, and the second algorithm in which a new selection formula with a desirability function *n*, which is presented in section 4, and also an improvement procedure, which is described in section 2, are used. The first algorithm is called the NACO algorithm, the second is called the IACOwith*n* algorithm. Both algorithms were tested for different values of graph density *q* {0.97, 0.974, 0.978, 0.982, 0.986, 0.99, 0.994, 0.998}. Tests were conducted for graphs with a number of vertices equal to $n = 200$, a number of cycles equal to $lc = 200$, a number of ants equal to $lm = 30$ and an evaporation rate equal to 0.997. An average maximum clique size for different graph density from 100 measurements and differences in these average maximum cliques size for both of these algorithms have been shown in Table 3 and in Fig. 5. As can be seen, a new selection formula with desirability function *n*, which has been described in section 4 and with an improvement procedure, which was described in section 2, allows receiving a bigger clique size in comparison to the case when an ant algorithm with desirability function described in [6] and with an improvement procedure is used.

This section presents only the main result of scientific investigation since the other investigation results also show that a new selection formula with desirability function *n*, which was described in section 4 and with an improvement procedure, which was described in section 2, improve the size of received clique. All used ant algorithm parameters were set experimentally.

Table 3

Average size, difference in size for different *q* and 200 vertices

Q	0.998	0.994	0.99	0.986	0.982	0.978	0.974	0.97
NACO	171.81	140.61	122.90	109.36	99.36	91.65	85.17	79.80
IACOwith*n*	172.29	140.94	123.41	109.98	100.23	92.58	86.12	80.91
IACOwith*n*-NACO	0.48	0.33	0.51	0.62	0.87	0.93	0.95	1.11

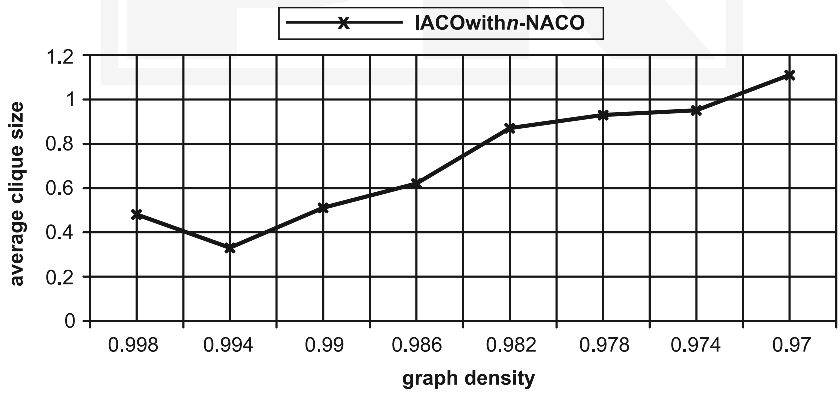

Fig. 5. Difference in average size for different *q* and 200 vertices

7. Conclusion

Experiments have shown that a selection formula with a desirability function *n* and an improvement procedure make an ant algorithm more powerful than over ant algorithms for the maximum clique problem when we take into consideration the size of the received clique. An improvement procedure allows getting larger cliques in comparison to the situation when this procedure is not used. Additionally, a new desirability function *n* is more helpful for ants to make a better selection of vertices in order to construct a maximum clique than other selection probability patterns, which have thus far been presented in scientific papers.

References

[1] Dorigo M., *Ant Algorithms for Discrete Optimization*, Artificial Life, 1999, Vol. 5, No. 2, pp. 137–172.
[2] Tang N., *Finding Maximum Cliques with Distributed Ants*, GECCO 2004, LNCS 3102, pp. 24–35.
[3] Fenet S., Solnon C., *Searching for Maximum Cliques with Ant Colony Optimization*, EvoWorkshops 2003, LNCS 2611, pp. 236–245.
[4] Xu X., Ma J., Lee J., *An Improved Ant Colony Optimization for the Maximum Clique Problem*, IEEE Third International Conference on Natural Computation, 2007.
[5] Rizzo J., *An Ant System Algorithm for Maximum Clique*, Master Thesis, The Pennsylvania State University, 2003.
[6] Schiff K., *Ant colony optimization algorithms for clustering problems*, Technical Transaction – Automatic Control, 2013, Y-110, 4-AC, pp. 77–87.