INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2024, VOL. 70, NO. 2, PP. 381-387
Manuscript received January 10, 2024; revised June, 2024. doi: 10.24425/ijet.2024.149556

Ant algorithms for finding weighted and unweighted maximum cliques in d-division graphs

Krzysztof Schiff

***Abstract*—This article deals with the problem of finding the maximum number of maximum cliques in a weighted graph with all edges between vertices from different d-division of a graph with the minimum total weight of all these cliques, and the problem of finding the maximum number of maximum cliques in a non-weighted graph with not all edges between vertices from different d-division of the graph. This article presents new ant algorithms with new desire functions for these problems. These algorithms were tested for their purpose with different changing input parameters, the test results were tabulated and discussed, the best algorithms were indicated.**

***Keywords*—d-division graph; ant algorithm; maximum clique; machine control; object tracking**

I. INTRODUCTON

DETERMINATION of weighted and unweighted all maximum cliques in d-division graphs serves to solve practical technical problems such as tracking multiple objects in vision systems[1]-[4] or controlling thread spools in textile machines [5], [6]. A graph is the set of vertices V and edges E and is denoted as $G(V, E)$. A graph clique occurs when all vertices are connected by edges, exactly every pair of clique vertices. A graph is d-divisible if the vertices of this graph can be divided into d sets of vertices and in each set there are vertices that are not connected to each other by edges, but edges exist between vertices from different divisions. The maximum clique in a d-division graph consists of vertices, each of which comes from a different division of the graph, and they are all connected by edges. The vertices of the d-division graph can therefore be subdivided into divisions $V = \{V_1, \dots, V_d\}$, $V_i = \{v_{ij},\ 1 \le j \le n\}$. Then the maximum clique can be expressed as follows $C = \{v_{l1}, \dots \dots, v_{ld}\}$, $v_{l1} \in V_1, \dots \dots, v_{ld} \in V_d$ and for each pair of $\{v_{li}, v_{lj}\}$ exists an edge $e(v_{li}, v_{lj})$. When the edges are assigned a weight, we are talking about a weighted graph, while when there are no weights, we are dealing with an unweighted graph. Due to the fact that determining the maximum cliques in d-division graphs is an NP-difficult problem [7]-[9], this problem is solved by artificial intelligence methods such as neural network [10], [11], particle swarm optimization [12], genetic algorithm [13]. A very good overview on this subject are works [14]-[16]. In this article, ant algorithms with a new dynamic desire functions are presented to solve these problems. For the first time ant algorithms were applied to these problems with a static function of desire, precisely in practical application in vision systems [17], [18]. Ant algorithms with a dynamic desire function have been shown to be better at achieving action goals, as demonstrated in the paper [19]. Comparative studies of the effectiveness of certain dynamic functions of desire for the problem of finding maximum cliques with their minimum total weight are presented in the paper [20] and for the problem of determining the maximum number of maximum cliques at work [21]. This paper presents results of research on the effectiveness of dynamic functions of desire, including desire functions containing, in addition to heuristic information on the current choice, additional heuristic information regarding the possibility of the next choice after the current choice. This work is organized as follows: section 2 describes the problem, section 3 describe the ant algorithm and section 4 described dynamic desire functions. The experimental results and their discussion are presented in Section 5, conclusions are given in Section 6.

II. PROBLEMS OF MAXIMUM CLIQUES

In d-divisible graphs, the vertices in these divisions are not connected to each other by any edge, but between these divisions the vertices are connected by edges, and if there are all such edges connecting all vertices from these divisions, the graph is called full, and the density of the graph q is 1. Of course, it may be that not all edges occur between vertices from different graph divisions and then the density of the graph q is less than 1. In each case, it is always about determining the maximum number of maximum cliques in d-divisible graphs.

In the case of a full graph, $q = 1$, there will be as many maximum cliques as there are vertices in each department of the graph, the number of vertices in each department of the graph is the same. In the case of an incomplete graph, $q \le 1$, the number of maximum cliques is not known.

Figure 1 shows a 4-divisons graph. For example, red, blue, and green cliques are the maximum number of cliques specified in such a graph, and the sum of the weights of those clicks is the minimum. The graph is full, but Figure 1 does not show the remaining edges between the vertices, which were excluded from the solution of finding the maximum number of maximum clicks with the minimum total weight.

Krzysztof Schiff is with Cracow University of Technology, Poland (e-mail: krzysztof.schiff@pk.edu.pl).

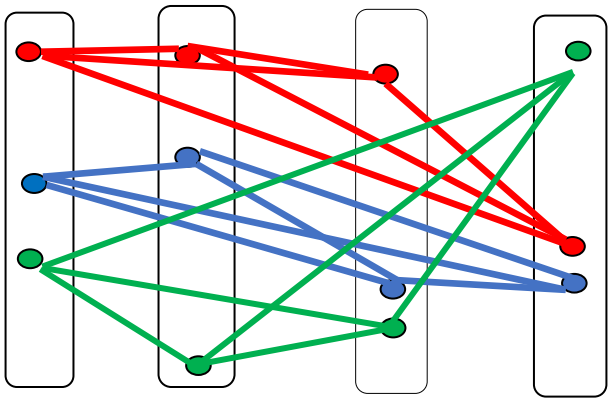

Fig. 1. Maximum number of maximum cliques in d–divisible graph $q = 1$.

Figure 2 shows the 4-divisons graph. For example, red, blue, and green cliques are the maximum number of cliques specified in such a graph, and the maximum number of cliques is 3, but the green clique can be indicated in a different way, as indicated by the green dashed line, in which case the maximum number of clicks will be 2. Since a graph is not a complete graph, it may be that the blue clique cannot be determined in any other way.

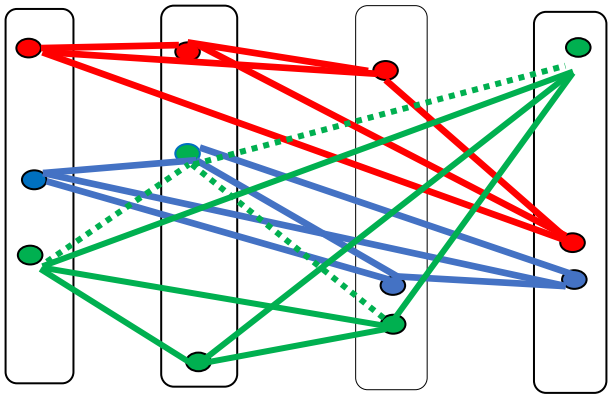

Fig. 2. Maximum number of maximum cliques in d–divisible graph $q < 1$.

The problem of finding the maximum number of maximum cliques in a weighted graph with all edges between vertices from different d-division of a graph with the minimum total weight of all these cliques can be stated as (1):

$$\min \sum_{i=1}^{k_n} w_{c_i} \tag{1}$$

where:
w_{c_i} – the weight of the ith clique
k_n – the maximum number of maximum cliques

This problem consists in determining all maximum cliques with the smallest possible total weight in a full graph, $q = 1$. Each maximum click consists of d vertices and each of the vertices comes from different d-sections of the graph. There are as many maximum cliques as there are vertices in each d-section of the graph.

The problem of finding the maximum number of maximum cliques in a non-weighted graph with not all edges between vertices from different d-division of the graph can be stated as (2):

$$\max\ i \quad of\ c_i,\ i \leq n \tag{2}$$

where:
c_i– the ith click in the d-division graph
n – the maximum number of maximum cliques while $q < 1$

The problem is to determine as many maximum cliques as possible in an incomplete graph, $q < 1$. Each maximum clique consists of d vertices and each of the vertices comes from different d-sections of the graph.

III. ANT ALGORITHM

The verbal description of the steps of the ant algorithm is shown in Listing 1 at the end of section IV. The algorithm consists of two main loops, of which the first inner loop cyclically repeats the action of each ant in the swarm, so it is performed as many times as there are ants in the swarm, and the second outer loop of them repeats the action of the whole swarm, so it is performed as many times as the whole swarm is supposed to work.

Each ant creates as many maximum cliques as can be created with a given input data set. Each ant creates a maximum clique, starting with the smallest, and systematically adds more vertices to the clique being created. However, when it finds a maximum clique, it starts creating another maximum clique until it has created all the maximum cliques. If the ant is at the vertex i and then it has the ability to include vertex j from the set of vertices available, from the set A, with probability specified (3).

$$p_{ij} = \frac{\tau_{ij}\,\eta_{ij}}{\sum_{k \in A}(\tau_{ik}\,\eta_{ik})} \tag{3}$$

where:
A - set of vertices available
p_{ij} - the probability of choosing vertex j when the ant is at the vertex i
τ_{ik} - the amount of pheromone on the edge connecting the vertex i with the vertex k
η_{ik} - a function of the desire to select vertex k by an ant at vertex i

The set of vertices available A is formed by those vertices that are not yet in the solution, i.e. in the clique created or in the cliques already created by the ant. In the case of a clique that is being created, vertices from already visited divisions of the graph cannot be included in set A when creating this clique. Thus, the available vertices are those vertices from the unvisited divisions of the graph, which are connected by edges to all vertices of the clique being created. In the case of an incomplete graph with a density of less than 1, there is also a need to verify and remove from the set of vertices available A those that are not connected by edges to all vertices of the clique being created.

Pheromone is a chemical agent with which ants communicate with each other. A larger amount of pheromone at the edges of the graph encourages ants to select these edges and include them in the created maximum clique when including the selected vertex in the created clique. Therefore, an additional amount of pheromone is stored on the edges that make up the solution to the problem, on the edges that form all the maximum cliques with the minimum sum of their weights. This additional quantity of pheromone depends on the smallest sum of the weights of all the maximum clicks determined by all ants in the current cycle

denoted by *best solution*$_c$ and the smallest sum of weights achieved so far in the previous cycles denoted by *best solution*$_g$ and this amount is expressed (4).

$$d\tau_s = \frac{1}{\left(1-\frac{best_{solution_g} - best_{solution_c}}{best_{solution_g}}\right)} \quad (4)$$

Of course, from all edges, part of the pheromone evaporates, and the intensity of evaporation depends on the evaporation coefficient ρ. The amount of pheromone at the edges of the graph in one cycle of the algorithm is therefore updated according to (5).

$$\tau_{ij} = (1-\rho)\tau_{ij} + d\tau_s \quad (5)$$

The vertex selection desire function η_{ij} is the heuristic information assigned to vertices j when the ant is at vertex i. This value is always calculated when the ant turns on another vertex i to the clique being created, so the value of the desire function is constantly changing.

The amount of pheromone at the edges of the graph indicates the extent to which the available vertices are suitable due to the purpose of the algorithm to be included in the clique created by the ant.

IV. DIFFERENT FUNCTIONS OF DESIRE

Depending on the purpose of the ant algorithm, different information turns out to be important when selecting the most suitable next vertex to be included in the clique being created. Tables I and II group the different functions of desire according to the purpose of the formic algorithms.

Table I contains the functions of desire due to the purpose of the ant algorithms, which is to determine the minimum sum of the maximum weights of all cliques, various functions containing information about:

1) the weight of the edge between vertex i and j, which is denoted as the weight of edge $e(i,j)$,
2) the sum of the edge weights by which vertex j is connected with the vertices of the clique formed so far by the ant, which is denoted as the *weight of clique* c_j,
3) possible next choice after selecting vertex j, which is marked as n_{j1}, see (6), vertices k are connected to vertex j.

The part of the desire function denoted by n_{j1} and expressed by the formula (6) is shown in Figure 2.

$$n_{j1} = \frac{1}{\min\{weight\ of\ c_k\}}, \qquad k \in A \quad (6)$$

An ant at the vertex i can incorporate different vertices j from the set of available vertices A. An ant may be guided by heuristic information about vertex j, which can be the weight of the edge between vertex i and vertex j which is denoted as *weight of edge* $e(i,j)$ or the sum of the edge weights that vertex j is connected to the vertices of the clique formed so far by the ant, which is denoted as *weight of clique* c_j, which edges for the example vertex j are marked with dashed red lines in Figure 3. The ant is also guided by additional heuristic information regarding the possible next vertex selection, i.e. heuristic information about the various vertices k and contained in the value of the quantity n_{j1}. The vertices k are vertices connected by edges to vertex j. These vertices have all the edges connecting them to the clique formed by the ant, so they also have the sum of their weights calculated, which is denoted as the *weight of clique* c_k, and from them the sum with the smallest value is selected and it is an additional heuristic information when selecting the vertex j defined by n_{j1}. Blue dot lines from vertex j to the other vertices k symbolize the edges and possible subsequent choices of the ant after selecting vertex j. Not all green dot lines in Figure 2 are plotted. Green solid lines are drawn for only one of the vertices k. The next possible choices after selecting vertex j are as many as there are vertices k connected by edges to vertex j.

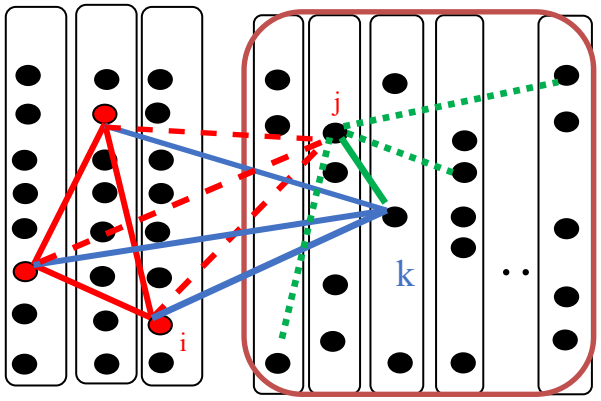

Fig. 3. Illustration for explanation of an additional heuristic information n_{j1}.

Taking into account the various heuristic information and their combinations, a number of different functions of desire and different ant algorithms were created. Six different ant algorithms with different desire functions to determine the minimum sum of the maximum cliques weights in a full graph with a density q of 1 are denoted from ALG_1 to ALG_6 and shown in Table I.

TABLE I
DESIRE FUNCTIONS - MINIMUM SUM OF WEIGHTS

	Formulas of desire functions
ALG_1	$\frac{1}{(weight\ of\ edge\ e(i,j))^3}$
ALG_2	$\frac{1}{(weight\ of\ clique\ c_j)^3}$
ALG_3	$\frac{1}{(weight\ of\ edge\ e(i,j))^2 * (weight\ of\ clique\ c_j - weight\ of\ edge\ e(i,j))^2}$
ALG_4	$\frac{1}{(weight\ of\ edge\ e(i,j))^2 * (weight\ of\ clique\ c_j)^2}$
ALG_5	$\frac{n_{j1}}{(weight\ of\ edge\ e(i,j))^2 * (weight\ of\ clique\ c_j - weight\ of\ edge\ e(i,j))^2}$
ALG_6	$\frac{n_{j1}}{(weight\ of\ edge\ e(i,j))^2 * (weight\ of\ clique\ c_j)^2}$

Table II contains the functions of desire, due to the purpose of the formic algorithm, which is to determine the maximum number of maximum cliques, various functions containing information about:

1) the *degree of vertex* j, i.e. the number of edges connecting other available vertices to that vertex j, which is the degree of vertex j, and the maximum degree among these vertices j, which is denoted as the *maximum degree of vertices*
2) the number of 3 vertex cliques (each vertex j and any vertex k with other available vertices can form some number of 3-vertices cliques) and the maximum number of these 3-vertices cliques, which is designated *number of 3 vertex cliques* and *maximum number of 3 vertex cliques* respectively,
3) possible next choice after selecting vertex j, which is marked as n_{j2}, see (7), vertices k are connected to vertex j.

The part of the desire function denoted n_{j2} and expressed (7) is shown in Figure 3.

$$n_{j2} = \frac{number\ of\ 3\ vertex\ cliques\ of\ vertex\ k\ +1}{maximum\ number\ of\ 3\ vertex\ cliques\ from\ vertices\ k} \quad (7)$$

An ant at vertex i can incorporate into the clique it creates different vertices j from the set of available vertices A. An ant can be guided by heuristic information about vertex j, which can be the degree of vertex j, which is denoted as *degrees of vertex* j, or the number of 3-vertex cliques that this vertex j forms with the other available vertices, which is denoted as *number of 3 vertex cliques*.

An example of a 3-vertex cliques is shown in blue in Figure 4. Each vertex j forms a certain number of such 3-vertex cliques, which is denoted as *number of 3 vertex cliques*. The maximum number of 3-vertex cliques formed by any of the vertices j is defined by *maximum number of 3 vertex cliques*.

The ant is also guided by additional heuristic information regarding its possible next selection, i.e. heuristic information regarding various vertices of k and this information is contained in the value of the quantity as n_{j2}. The vertices of k are vertices connected by an edge to the vertex j. Figure 2 shows only one vertex k with one 3-vertex clique. The green dotted line indicates the existing edge between the vertex j and k. Not all green dotted lines are specified in Figure 3. Of course, there may be more green 3-vertex cliques formed by vertex k. The vertices k are all vertices connected by an edge to the vertex j, and each vertex thus forms a different number of 3-vertices cliques, including their maximum number from all vertices k connected by an edge to any vertex j. Of all the 3-vertex cliques of k connected to vertex j, the maximum is selected and is additional information about vertex j and is denoted as the *number of 3 vertex cliques of vertex* k. Of the 3-vertex cliques found for all vertices k connected by edges to all vertices j, the largest is chosen and denoted as the *maximum number of 3 vertex cliques from vertices* k. The ratio of these two quantities is the heuristic information of vertex j and is denoted by n_{j2}.

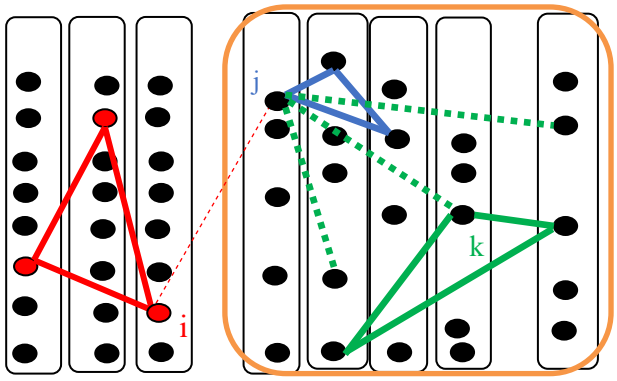

Fig 4. Illustration for explanation of an additional heuristic information n_{j2}.

Taking into account the various heuristic information and their combinations, a number of different functions of desire and different ant algorithms were created. Four different ant algorithms with different desire functions to determine the maximum number of maximum cliques in an incomplete graph, i.e. density q less than 1, are denoted from ALG_N1 to ALG_N4 and presented in Table II.

TABLE II
DESIRE FUNCTIONS - MAXIMUM NUMBER OF MAXIMUM CLIQUES

	Formulas of desire functions
ALG_N1	without a desire function
ALG_N2	$\frac{number\ of\ 3\ vertex\ cliques+1}{maximum\ number\ of\ 3\ vertex\ cliques}$
ALG_N3	$\frac{degres\ of\ vertex\ j+1}{maximum\ degree\ of\ vertices}$
ALG_N4	$\frac{(number\ of\ 3\ vertex\ cliques+1)\ n_{j2}}{maximum\ number\ of\ 3\ vertex\ cliques}$

The verbal description of the steps of the ant algorithm taking into account both goals of its operation is presented in Figure 1.

LISTING 1
THE VERBAL DESCRIPTION OF THE STEPS OF THE ANT ALGORITHM

BEGIN
1. The initial pheromone value is given to all edges of the d-division graph
2. Cycle loops - a whole swarm of ants works in one cycle
3. Each ant from the swarm designates the solution, this is the maximum number of maximum cliques:
3.1 From the available vertices, the ant chooses one by the roulette method with the probability p_{ij} and includes in the solution using the amount of pheromone and the function of desire
3.2 The set of available vertices A and their functions of desire are redetermined
3.2a The edge weights of the resulting solution are added together
3.2b The number of designated maximum cliques of the resulting solution is added together
3.2a The smallest sum of cliques weights from a given cycle is remembered

3.2b The largest number of maximum cliques from a given cycle is remembered
4. After the end of the cycle
4.1 The best solution of all cycles is stored
4.2 The pheromone value is updated: the amount of pheromone added to the edges of the solution is calculated and the evaporation coefficient is used
END.

V. EXPERIMENTAL RESULTS

The developed algorithms have been tested for different purposes of using the algorithm, and so Tables III and IV present the results for obtaining the minimum sum of weights of all edges of all maximum cliques in full graphs with a density q of 1, while Tables V, VI and VII present the results for obtaining the maximum number of maximum cliques in incomplete graphs with a density q less than 1. All tables show values averaged over the 10 results obtained in each case examined. All graphs have been generated randomly, weights were generated in range of 1 to 100.

Table III shows the obtained minimum sums of weights of all maximum cliques depending on the number of divisions of the d-division graph, and so the number of divisions was $Ldz\in\{10, 15, 20, 25, 30\}$ with a constant number of vertices in each of these divisions equal to 25 and a constant graph density equal to 1 and with constant parameters of the ant algorithm such as the number of cycles, which was 30, the number of ants which was 10 and the evaporation coefficient of 0.005. The most effective in their operation turn out to be ant algorithms with the function of desire defined as ALG_5 and ALG_6. It is clear that with the increase in the number of graph divisions, i.e. with the increase in the dimension of the determined maximum cliques, the minimum sum of the weights of the maximum cliques increases.

TABLE III
MINIMUM SUM OF CLIQUES WEIGHTS – NUMBER OF GRAPH DIVISIONS

	Ldz=10	Ldz=15	Ldz=20	Ldz=25	Ldz=30
ALG1	45 624	114 786	215 560	347 597	510 958
ALG2	43 648	111 766	211 569	341 426	523 796
ALG3	42 223	109 223	206 848	335 839	496 417
ALG4	42 658	109 018	207 055	336 624	497 680
ALG5	41 557	106 959	204 187	332 269	492 191
ALG6	41 810	107 424	204 351	332 874	493 106

Table IV shows the obtained minimum sums of weights of all maximum cliques depending on the number of vertices in each division of the d-division graph, and so the number of vertices in the graph divisions was $Lo\in\{20, 25, 30, 35, 40\}$ with a constant number of graph divisions equal to 20 and a graph density constant of 1 and constant parameters of the ant algorithm such as the number of cycles, which was 30, the number of ants which was 10 and the evaporation coefficient of 0.005. Here, too, the most effective in their operation turn out to be ant algorithms with the function of desire defined as ALG_5 and ALG_6. It is also clearly visible here that with the increase in the number of vertices in the graph divisions, i.e. with the increase in the number of maximum cliques determined, the minimum sum of weights increases.

Table V shows the obtained maximum number of maximum cliques depending on the number of divisions of the d-division graph, and so the number of divisions was $Ldz\in\{5, 7, 9, 11, 13\}$ with a constant number of vertices in each of these divisions equal to 50 and a constant density of the graph equal to 0.75 and with constant parameters of the ant algorithm such as the number of cycles, which was 30, the number of ants, which was 10 and a evaporation coefficient of 0.005. The most effective in its operation turns out to be an ant algorithm with a desire function referred to as ALG_N4. It is clear here that with the increase in the number of graph divisions, i.e. with the increase in the dimension of maximum cliques, the number of maximum cliques is decreasing.

TABLE IV
MINIMUM SUM OF CLIQUES WEIGHTS – NUMBER OF VERTICES IN DIVISIONS

	Lo=20	Lo=25	Lo=30	Lo=35	Lo=40
ALG1	172 411	215 560	258 237	301 306	344 782
ALG2	169 599	211 569	253 109	295 301	337 464
ALG3	165 896	206 848	247 344	287 822	328 570
ALG4	165 938	207 055	247 610	288 318	328 961
ALG5	164 374	204 187	244 045	284 057	324 276
ALG6	164 436	204 351	244 352	284 317	324 291

TABLE V
MAXIMUM NUMBER OF MAXIMUM CLIQUES – NUMBER OF GRAPH DIVISIONS

	ALG_N1	ALG_N2	ALG_N3	ALG_N4
Ldz=5	49.4	49.4	49.4	49.6
Ldz=7	46.7	46.8	46.8	46.9
Ldz=9	40.9	41.3	40.9	41.8
Ldz=11	30.7	31.0	30.8	32.7
Ldz=13	17.4	17.9	17.2	19.3

Table VI shows the obtained maximum number of maximum cliques depending on the number of vertices in each divisions of the d-division graph, and so the number of vertices in the graph divisions was $Lo\in\{20, 30, 40, 50, 60\}$ with a constant number of graph divisions equal to 11 and a graph density constant of 0.75 and constant parameters of the ant algorithm such as the number of cycles, which was 30, the number of ants, which was 10 and a evaporation coefficient of 0.005. Here, too, the most effective in its operation turns out to be an ant algorithm with a desire function defined as ALG_N4. It is also clear here that with the increase in the number of vertices in the graph divisions, the maximum number of maximum cliques increases.

Table VII shows the obtained maximum number of maximum cliques depending on the density number of the d-division graph, and so the graph density was $q\in\{0.60, 0.65, 0.70, 0.75, 0.80, 0.85\}$ with a constant number of graph divisions equal to 11 and a constant number of vertices in graph divisions equal to 60 and constant parameters of the ant algorithm such as the

number of cycles, which was 30, the number of ants, which was 10 and a evaporation coefficient of 0.005. Here, too, the most effective in its operation turns out to be an ant algorithm with a desire function defined as ALG_N4. It is also clear here that with the increase in graph density, the maximum number of maximum cliques increases.

Ant algorithms, regardless of the purpose of operation, were tested with changes in the values of ant algorithm parameters such as the number of cycles, the number of ants or the evaporation rate. The operation of algorithms and the results are similar and the best algorithms turn out to be, depending on the purpose of the action, and so the algorithms ALG_5 and ALG_6 for the purpose of determining the minimum sum of the weights of the maximum cliques and the algorithm ALG_N4 for the goal of determining the maximum number of maximum cliques. Ant algorithms determining the maximum number of maximum cliques were tested at a constant number of divisions of the graph Ldz=9 and a constant number of vertices in divisions Lo=50 and at graph density q=0.65, and ant algorithms determining the minimum sum of weights of maximum cliques were tested with a constant number of divisions of the graph Ldz=15 and a constant number of vertices in divisions Lo=25 and with a graph density q of 1.0.

TABLE VI
MAXIMUM NUMBER OF MAXIMUM CLIQUES – VERTICES IN GRAPH DIVISIONS

	ALG_N1	ALG_N2	ALG_N3	ALG_N4
Lo=20	6.5	6.9	6.9	7.0
Lo=30	13.5	14.2	13.8	15.1
Lo=40	21.6	22.8	21.9	23.6
Lo=50	30.7	31.0	30.8	32.7
Lo=60	40.2	41.4	40.6	42.0

TABLE VII
MAXIMUM NUMBER OF MAXIMUM CLIQUES – GRAPH DENSITY Q

	ALG_N1	ALG_N2	ALG_N3	ALG_N4
Q=0.60	4.0	4.6	3.9	3.9
Q=0.65	10.6	12.2	11.1	12.7
Q=0.70	24.3	27.1	25.6	27.2
Q=0.75	40.2	41.4	40.6	42.0
Q=0.80	50.4	50.9	50.6	51.6
Q=0.85	55.7	56.1	55.7	56.3

TABLE VIII
MAXIMUM NUMBER OF MAXIMUM CLIQUES – NUMBER OF CYCLES

	ALG_N1	ALG_N2	ALG_N3	ALG_N4
Lc=10	21.6	24.2	22.1	24.4
Lc=20	21.3	24.2	22.7	24.3
Lc=30	22.6	23.7	22.8	25.1
Lc=40	22.3	23.9	22.4	26.2
Lc=50	22.7	24.4	23.3	26.3

Tables VIII and IX show respectively the maximum number of maximum cliques and the minimum sum of the clique weights depending on the number of cycles $Lc\in\{10, 20, 30, 40. 50\}$ with constants of other parameters of the ant algorithm such as the number of ants, which was 10, and the evaporation coefficient of 0.005.

Tables X and XI show respectively the maximum number of maximum cliques and the minimum sum of the clique weights depending on the number of ants $Lm\in\{10, 20, 30, 40, 50\}$ with constants of other parameters of the ant algorithm such as the number of cycles, which was 30, and the evaporation coefficient of 0.005.

TABLE IX
MINIMUM SUM OF CLIQUES WEIGHTS – NUMBER OF CYCLES

	Lc=10	Lc=20	Lc=30	Lc=40	Lc=50
ALG1	115 395	114 848	114 786	114 658	114 480
ALG2	112 407	111 922	111 766	111 317	111 324
ALG3	109 500	109 509	109 223	108 909	108 663
ALG4	109 546	109 504	109 018	109 009	108 935
ALG5	108 847	108 356	106 959	108 143	108 027
ALG6	108 718	108 823	107 424	108 168	108 184

TABLE X
MAXIMUM NUMBER OF MAXIMUM CLIQUES – NUMBER OF ANTS

	ALG_N1	ALG_N2	ALG_N3	ALG_N4
Lm=10	22.6	23.7	22.9	25.1
Lm=20	23.2	24.9	23.6	25.8
Lm=30	23.1	24.8	24.1	26.6
Lm=40	23.4	25.3	23.9	26.2
Lm=50	23.1	25.2	23.9	26.5

TABLE XI
MINIMUM SUM OF MAXIMUM CLIQUES – NUMBER OF ANTS

	Lm=10	Lm=20	Lm=30	Lm=40	Lm=50
ALG1	114 786	114 326	114 312	114 327	114 154
ALG2	111 766	111 306	110 833	111 129	110 781
ALG3	109 223	108 783	108 303	108 367	107 957
ALG4	109 018	108 826	108 923	108 763	108 572
ALG5	106 959	108 115	107 807	107 852	107 247
ALG6	107 424	108 121	107 985	107 816	107 683

TABLE XII
MAXIMUM NUMBER OF MAXIMUM CLIQUES – EVAPORATION RATE

	ALG_N1	ALG_N2	ALG_N3	ALG_N4
ρ=0.001	22.2	23.4	22.8	24.6
ρ=0.003	22.4	24.3	22.8	25.3
ρ=0.005	22.6	23.7	22.9	25.1
ρ=0.007	22.2	23.1	23.0	25.1
ρ=0.009	22.4	24.0	23.1	25.5

Tables XII and XIII present respectively the obtained maximum number of maximum cliques and the obtained minimum sum of weights of maximum cliques depending on the evaporation coefficient $\rho \in \{0.001, 0.003, 0.005, 0.007, 0.009\}$ with constants of other parameters of the ant algorithm such as the number of cycles, which was 30, and the number of ants, which was 10.

TABLE XIII
MINIMUM SUM OF CLIQUES WEIGHTS – EVAPORATION RATE

	ρ=0.001	ρ=0.003	ρ=0.005	ρ=0.007	ρ=0.009
ALG1	114 871	114 888	114 786	114 172	114 779
ALG2	111 733	111 484	111 766	111 891	111 569
ALG3	108 967	108 899	109 223	108 962	108 781
ALG4	108 980	109 056	109 018	109 199	109 079
ALG5	108 172	108 360	106 959	108 252	108 310
ALG6	108 306	108 617	107 424	108 391	108 704

VI. CONCLUSIONS

The conducted comparative tests of ant algorithms with various functions of desire show that the functions of desire taking into account not only the direct choice of the ant but also its subsequent choices turn out to be more effective in achieving the goals in which these algorithms operate. For both ant algorithms, the desired functions developed with regard to future ant choices allowed to achieve better results with different input parameters such as the number of divisions of the d-division graph, the number of vertices in the graph divisions, and the density of the graph q.

REFERENCES

[1] A. Dehgan, S.M. Assari, M. Shah, “GMMCP Tracker: Globally Optimal Generalized Maximum Multi Clique Problem for Multiple Object Tracking”, Conference on Computer Vision and Pattern Recognition, pp. 4091-4099, 2015.

[2] L. Wen, Z. Lei, S. Lyu, S.Z. Li, M. Yang, “Exploiting hierarchical dense structures on hyper-graphs for multi-object tracking”, IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 38, no. 10, pp. 1983–1996, 2016

[3] A.R. Zamir, A. Dehghan, M. Shah, “GMCP-tracker: global multi-object tracking using generalized minimum clique graphs”, in European Conference on Computer Vision, 2012.

[4] W. Liu, O. Camps, M.Sznaier, “Multi-camera multi-object tracking” in Computer Vision and Pattern Recognition, arXiv:1709.07065, pp. 1-7, 2017.

[5] T. Grunert, S. Irnich, H.J. Zimmermann, M. Schneider, B. Wulfhorst, “Finding all k-cliques in k-partite graph, an application in textile engineering”, Computer and Operation Research, vol. 29, no. 2, pp. 13-31, 2002.

[6] M. Mirghorbani, P. Krokhmal, “On finding k-cliques in k-partite graphs”, Optimization Letters, vol 7, pp. 1155 -1165, 2013

[7] D. Deb, M. Yeddanapudi, K. PattiPati, T.A. Bar-Shalom, “A generalized S-D assignment algorithm for multi-sensor-multitarget state estimation”, IEEE Transactions on Aerospace and Electronic Systems, vol. 33, pp. 523-538, 1999.

[8] C. Feremans, M. Labbe, G. Laportee, “Generalized network design problem”, European Journal of Operational Research, vol. 148, pp. 1-13, 2003.

[9] A.M.C.A. Koster, S.P.M. Hoesel, A.W.J. Kolen, “The partial constraint satisfaction problem: Facets and lifting theorems”, Operation Research Letters, vol. 23, pp. 89-97, 1998.

[10] K. Yoon, D.Y. Kim, Y.C. Yoon, M. Jeon, “Data association for Multi-Object Tracking via Deep Neural Networks”, Sensors, vol. 19, no. 3, pp. 1-17, 2019.

[11] B. Lee, E. Erdenee, S. Jin, P.K. Rhee, “Efficient object detection using convolutional neural network-based hierarchical feature modeling”, Image Video Proc., vol. 10, no. 8, pp. 1503–1510, 2016.

[12] J. Kennedy, R. Eberhart, “Particle swarm optimization” in Proc. IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948, 1995.

[13] G. Chen, L. Hong, “A genetic based multi-dimensional data association algorithm for multi sensor multi target tracking”, Mathematical Computing Models, vol. 26, pp. 57-69, 1997.

[14] I.A. Temitope, S. Patrick Sebastian, I.I. Lila, I. Oladimeji, S.A. Lukman, A.B. Abdulrahman, B. Abubakar, A.S. Yau, “Multi-camera multi-object tracking: A review of current trends and future advances”, Neurocomputing, no. 552, pp. 1-32, 2023.

[15] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, T.K. Kim, “Multiple object tracking: A literature review”, Artificial Intelligence, no. 293, pp. 1-23, 2021.

[16] Ch. Qiu, Z. Zhang, H. Lu, H. Luo, “A survey of motion-based multi-target tracking methods”, Progress in Electro-magnetic Research, no. 62, pp. 195-223. 2015.

[17] A.O. Bozdogan, M. Efe, “Improved assignment with ant colony optimization for multi-target tracking”, Expert Systems with Applications, no. 38, pp. 9172–9178, 2011.

[18] A.O. Bozdogan, M. Efe, “Ant Colony Optimization Heuristic for the multidimensional assignment problem in target tracking”, IEEE National Radar Conference, 2008.

[19] K. Schiff, “Ant colony optimization for object identification in multi-cameras video tracking systems” in New advances in dependability of networks and systems, Seventeenth International Conference on Dependability of Computer Systems: DepCoS-RELCOMEX, June 27-July 1, Wrocław, Poland, 2022

[20] K.Schiff, “Partitioning of an M-Part Weighted Graph with N Vertices in Each Part into N Cliques with M Vertices and the Total Minimum Sum of Their Edges Weights Using Ant Algorithms”, in: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds) Dependable Computer Systems and Networks. DepCoS-RELCOMEX 2023. Lecture Notes in Networks and Systems, vol 737. Springer, Cham. https://doi.org/10.1007/978-3-031-37720-4_24

[21] K.Schiff, “Ant Colony Optimization Algorithm for Finding the Maximum Number of d-Size Cliques in a Graph with Not All m Edges between Its d Parts”, in: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds) Dependable Computer Systems and Networks. DepCoS-RELCOMEX 2023. Lecture Notes in Networks and Systems, vol 737. Springer, Cham. https://doi.org/10.1007/978-3-031-37720-4_23