Informatics and Control Problems 41 Issue 1 (2021)
journal homepage: www.icp.az

Application of greedy algorithms in combinatorial optimization

F.A. Sharifov

V.M. Glushkov Institute of Cybernetics of National Academy of Sciences of Ukraine, Kyiv, Ukraine

ARTICLE INFO

Article history:
Received 16.03.2021
Received in revised form 25.03.2021
Accepted 30.03.2021
Available online 20.05.2021

Keywords:
Maximum cut
Stable set
Vertex cover
Clique
Hamiltonian cycles

ABSTRACT

The paper considers well-known combinatorial optimization problems such as the maximum cut problem, the stable set problem (finding the maximum independent set of graph vertices), the clique problem, the minimum vertex cover problem, and the Hamiltonian cycle problem. The aim is to construct a mathematical model of these problems in terms of the theory of polymatroids, to study the functional relationship between their feasible solutions, as well as to describe the solution algorithms.

1. Introduction

The paper deals with well-known combinatorial optimization problems, such as the maximum cut problem, the stable set problem (finding the maximum independent set of graph vertices), the clique problem, the minimum vertex cover problem, and the Hamiltonian cycle problem. The aim is to construct a mathematical model of these problems in terms of the theory of polymatroids, to study the functional relationship between their feasible solutions, as well as to describe the solution algorithms. All problems are considered on an undirected simple graph $G = (V, E)$. Similar studies were carried out earlier in [1, 2].

The feasible solutions of the listed problems are bipartite spanning graphs in the maximum cut problem, subgraphs with many isolated vertices in the stable set problem, complete subgraphs in the clique problem, simple spanning cycles in the Hamiltonian cycle problem, subgraphs whose vertices cover all edges of the graph G in the vertex cover problem. One of the possible ways to describe the set of feasible solutions of the problems under consideration is to define the convex hull of $|E|$-dimensional characteristic vectors of admissible subgraphs. Then each of the above problems can be defined as the pair (Q, G), where G is a given graph and Q is the convex hull of the characteristic vectors of admissible subgraphs. It is known that all these problems are NP-hard for the general case of the graph G [3-5]. If there is a polynomially bounded algorithm for solving one of the NP-hard combinatorial optimization problems, then all other problems are also polynomially solvable. Such an amazing statement about the *algorithmic equivalence* of NP-hard problems on the arbitrary graph G is not true if G is not an arbitrary graph. For instance, the maximum cut problem is polynomially solvable on planar graphs (see [6]), but the stable set problem remains NP-hard [7-9] in this case. This example confirms that an NP-hard problem that is polynomially solvable on a

E-mail address: fasharifov@gmail.com (F.A. Sharifov).

www.icp.az/2021/1-07.pdf

special graph G can be transformed into another problem on a graph that is not isomorphic to G. To transform one problem into another on the same graph, it is useful to construct their models that have much in common. For instance, in terms of such a broad theory as mathematical programming, only the linear nature of the constraints and the objective function is common, inherent in the known models (see [10]) of the problems under consideration.

In the process of studying the non-algorithmic equivalence of these problems for special cases of the graph G, it is a natural fact that there is a functional connection between their admissible subgraphs from the set Q. To study such a connection, the models of the problems under consideration are presented in the paper in terms of such a narrow theory as polymatroids. It is shown that the problems under consideration can be formulated with similar constraints and different types of the objective function. The common feature of the presented models is that the optimal solution to each problem should be searched for among the bases (extreme points) of the polymatroid (constraint polyhedron) associated with the graphs G. Despite the different types of objective functions, it is also shown that the admissible subgraphs of one problem are a subset of the set Q of the other one. Unfortunately, within the framework of the theory of polymatroids, it is also not trivial to identify algorithmically equivalent problems from different classes for special cases of the graph G. Nevertheless, applying this narrower theory to the study of previously unknown properties of the above problems can be considered as a systematization of combinatorial algorithms, revealing their common features and patterns. A connected graph with unit weights of edges and vertices is called unweighted. An unweighted and undirected graph that does not contain parallel edges, isolated vertices, and loops is called a simple graph. In the paper, only simple graphs are considered for simplicity of presentation.

2. Definitions and auxiliary results

Suppose a simple graph $G = (V, E)$ is given with sets of vertices V and edges V. The set of edges with one endpoint in $S \subseteq V$ and another endpoint in $\overline{S} = V \backslash S$ is called the cut determined by the subset S and is denoted by $\overline{S} = V \backslash S$. For an arbitrary $S \subseteq V$, sets of edges with final vertices in S, with at least one final vertex in S, are denoted by $\gamma(S)$ and $\kappa(S)$, respectively. In what follows, we also use the notation $n = |V|$, $m = |E|$ and $a(F)$ for $\sum (a_j; j \in F)$, where F is an arbitrary subset of V and $a = (a_v; v \in V)$ is a vector whose components are indexed by the vertices of the graph G. From the definition of $\delta(S)$, $\kappa(S)$ and $\gamma(S)$, it follows that $\gamma(S) \backslash \kappa(S) = \delta(S)$ for an arbitrary $S \subseteq V$.

Consider the functions

$$f(S) = |\kappa(S)| \text{ and } g(S) = |\gamma(S)|,$$

determined on subsets S of the set V. It is know that $f(S)$ is a submodular, $g(S)$ – supermodular monotone functions. The polyhedron

$$P = \{x \in R^V; x(S) \leq f(S), S \subseteq V, x \geq 0\}$$

called a polymatroid associated with the graph G. The vector $x \in P$ is called the base of the polymatroid P, if $x(V) = f(V)$. It follows from the monotonicity of the function $f(S)$ that with respect to the linear ordering L of the vertices of the graph G greedy algorithm will determine the base $x \in P$ corresponding to L. It is not difficult to show that the vector $y = d - x$ is also a base of P, where $d = (d_v = |\delta(v)|; v \in V)$ and d_v is the degree of the vertex v.

The polyhedron

$$EP = \{z \in R^V; z(S) \leq f(S) - g(S), S \subseteq V\}$$

is called an extended polymatroid. It is easy to understand from the definition of the functions $f(S)$ and $g(S)$ that

$$f(S) + g(S) = d(S) \text{ and } f(S) - g(S) = \delta(S)$$

for all $S \subseteq V$. It follows from these equalities that the vectors $2x - d$, $d - 2x$ are the bases of EP, i.e., $2x - d, d - 2x \in EP$ and

$$2x(V) - d(V) = d(V) - 2x(V) = 0.$$

The proof of all the above facts can be found in [11]. In what follows, in order not to redefine the functions f and g, as well as the polyhedrons P and EP each time, we assume that they are similarly definite for the considered graphs.

3. Perfect matchings in graphs

In this section, we briefly outline the criterion for the existence of a perfect matching in bipartite and arbitrary graphs in terms of polymatroids [1, 2]. Suppose a bipartite graph $G = (UV, E)$ is given with sets of vertices UV and edges E, where $UV = U \cup V$ and U, V is the set of vertices of each part. Without loss of generality, we can assume that $|U| = |V|$. Consider a vector $b = (b_v; v \in UV)$, where $b_u = d_u - 1$ for $u \in U$ and $b_v = 1 - d_v$ for $v \in V$.

Theorem 1. *[1]. The bipartite graph $G = (UV, E)$ contains a perfect matching if and only if the vector $b = (b_v; v \in UV)$ is the base of the extended polymatride EP.*

Based on this theorem, a model is proposed for the problem of finding the maximum subgraph containing a perfect matching on a bipartite graph with nonnegative edge weights. It is also shown that the matrix of constraints of the last problem is unimodular, and $O(n^3)$ – the algorithm for solving it – is proposed.

To formulate a similar theorem for an arbitrary graph, consider the graph $G = (V, E)$ with sets V and E of vertices and edges. Since f is a monotone function, the greedy algorithm determines the base of the polymatroid P with respect to an arbitrary linear ordering L of the vertices. Suppose the base $x \in P$, as well as the vectors x^1 and x^1 are determined using x and $y = d - x$ as follows:

1) if $x_v = 0$ for the vertices $v \in V$, then $x_v^1 = x_v$ and $y_v^1 = y_v - 1 \geq 0$;
2) if $y_v = 0$ for the vertices $v \in V$, then $x_v^1 = x_v - 1 \geq 0$ and $y_v^1 = y_v$;
3) for the rest of the vertices v, or $x_v^1 = x_v - 1$ and $y_v^1 = y_v$, or $x_v^1 = x_v$ and $y_v^1 = y_v - 1$.

Theorem 2. *[2]. The graph $G = (V, E)$ contains a perfect matching if and only if for the base $x \in P$, vertices generated by the arbitrary linear ordering L, and the vector $y = d - x$, using rules 1–3, we can determined the vectors x^1 and y^1, such that $z = x^1 - y^1$ is the base of the extended polymatroid EP.*

In the case of the bipartite graph $G = (UV, E)$, the base $x = (x_u = d_u; u \in U, x_v = 0, v \in V)$ is determined by the greedy algorithm with respect to the linear ordering $L = (U, V)$ of the vertices. Therefore, $y = (y_u = 0; u \in U, y_v = d_v, v \in V)$ and $z = b$. In the case of the non-bipartite graph $G = (V, E)$, it is difficult to specify in advance the linear ordering of the vertices, relative to which the values x_v^1 and y_v^1 can be uniquely determined using rule 3.

Based on Theorem 2, the problem of finding a perfect matching of the minimum weight is reduced to the problem of finding the minimum cost flow on an almost bipartite (weakly bipartite) graph.

4. Maximum cut problem

In terms of polymatroids, the problem of finding the maximum cut in a given simple graph $G = (V, E)$ is formulated as follows [10]: find

$$MaxCut = \max\{Cut(x) = \frac{1}{2}\sum_{v \in V} |2x_v - d_v|\} \quad (1)$$

with the constraints

$$x \in B, \tag{2}$$

where B is the convex hull of the bases of the polymatroid P. Since $Cut(x)$ is a convex function, there exists the base x^* (extreme point) in B, which is a solution to problem (1), (2). In other words, there is a linear ordering L_* of the vertices, with respect to which the base x^* can be determined efficiently using the greedy algorithm. The kinear ordering L_* will be called the *optimal* linear ordering of the vertices. It is clear that when determining the order of the vertices in L_* it is necessary to take into account the topology of the graph G. For example, with respect to bipartite graphs, it is sufficient that in L_* so that the vertices of one part precede the vertices of the other. Therefore it follows from the formula of calculation of x_v^* by the greedy algorithm that either $x_v^* = d_v$, or $x_v^* = 0$. In the case when $G = C_k$ is a simple cycle (chordless cycle), if $|E(C)| = k$ is an even number, then the optimal ordering L_* is determined in the same way as for a bipartite graph. In the case when k is an odd number, L_* is determined in the same as for the simple path obtained after removing an arbitrary edge from the odd cycle C_k. Therefore L_* is determined in the same way as for even cycles. In this case, either $x_v^* = d_v = 2$, or $x_v^* = 0$ except for one vertex v of the cycle C_k, for which $0 < x_v^* = 1 < 2 = d_v$. Obviously there exist $\frac{k}{2}!$ and $\frac{k-1}{2}!$ options of the optimal ordering L_* of the vertices of simple even and odd cycles C_k, respectively. Thus, in the case when G is an arbitrary graph, it is necessary to find in L_* the order of the vertices v of simple even and odd cycles to maximize the absolute value $2x_v - d_v$ in objective function (1), for each $v \in V$. For this purpose, the use of well-known algorithms (see [12]) for determining simple cycles is fraught with great difficulties, since there is an exponential number of such cycles in G. Despite these difficulties, we will show in the following paragraphs that the refinement of the ordering of the vertices of odd and even cycles in the predetermined linear ordering L to find a new base x with the best value of the function $Cut(x)$ can be implemented by applying the shortest path algorithm.

5. Calculating the upper bound for $MaxCut$

In what follows, all algorithms that determine the partition of the set V into two disjoint subsets using the linear ordering of the vertices will be called greedy-type algorithms. In [10], the following *greedy type* algorithm $O(mn)$ was proposed. Given the abbreviation for Ggreedy Algorithm Maximum Cut, let us call it the GAMC algorithm.

The input of the GAMC algorithm is the graph G.

1. for $k = 1, \dots, n$ determine the base x^k from B and the cut $\delta(V^k)$ as follows.

2. Assume $V^+(k) := k$ and $V^-(k) := \emptyset$ and execute the following steps.

3. As long as there exists vertix t such that $V^+(k) + t$ is an independent set of the graph, include t in $V^+(k)$ and assume $x_t^k = d_t$. If there are several such vertices, among them select the vertex t with the maximum d_t.

4. Find the vertex $t \notin V^+(k)$ for which $d_t - n(t)$ is maximum, here $n(t)$ is the number of edges with one final vertex t and the other in $V^+(k)$. Assume $x_t^k = d_t - n(t)$. If $x_t^k > 0$, include t in $V^+(k)$, otherwise include t in $V^-(k)$.

5. Determine a record solution, i.e. the base $x \in B$, for which

$$Cut(x) = \max\{Cut(x^k); k = 1, \dots, n\},$$

and assume $V^+ = \{v; x_v - y_v > 0, v \in V\}$, $V^- = \{v; x_v - y_v \le 0, v \in V\}$, where $y = d - x$.

The output of the GAMC algorithm is the line

$$x, y, Cut(x), \delta(V^+), V^+, V^-.$$

Obviously, $V^+ = V \backslash V^-$ and $\delta(V^+) = \delta(V^-)$ by definition of the subsets V^+ and V^-. A simple analysis shows that on some random graphs the base x is an optimal solution to problem (1), (2). Now we will show that this algorithm also simultaneously determines the upper bound for $MaxCut$ over $O(mn)$ time, considering which it is possible to estimate the relative error of the solution of problem (1), (2). According to the abbreviation for Upper Value, the upper bound is denoted $UpVal$.

Theorem 3. *The GAMC algorithm calculates the upper bound for problem (1), (2) over* $O(mn)$ *time.*

To prove the theorem, we first prove the following lemma.

Lemma 1. *For the base* x *determined by the GAMC algorithm, and* $y = d - x$ *at least one of the inequalities:*

$$x(V^+) \geq MaxCut, y(V^-) \geq MaxCut$$

holds.

Proof. Suppose the base x is determined with respect to the linear ordering $L = (V^+, V^-)$. Since $v \prec u$ for the arbitrary vertices $v \in V^+$ and $u \in V^-$ в $L = (V^+, V^-)$, then

$$x(V^+) = f(V^+) = d(V^+) - g(V^+).$$

It follows frrom the determination of the function f and the equality

$$y(V^-) = d(V^-) - x(V^-)$$

that $y(V^-) = f(V^-)$. Suppose that both inequalities are not satisfied, i.e., their sum $f(V^+) + f(V^-) < 2MaxCut$. Since

$$f(V^+) + f(V^-) = m + 2|\delta(V^+)|,$$

it follows from the determination of the function f from the last inequality that

$$|\delta(V^+)| < MaxCut - m/2.$$

Clearly, $x_v \geq \frac{d_v}{2}$ for all $v \in V^+$ and $y_u \geq \frac{d_u}{2}$ for all $u \in V^-$ by the determination of the base x, i.e., $|\delta(V^+)| > m/2$. Therefore, it follows from the previous inequality that $MaxCut > m$. This inequality contradicts the fact that $m \geq MaxCut$ for an arbitrary graph.

Thus, by the statement of the lemma,

$$UpVal = \max\{x(V^+), y(V^-)\} \quad (3)$$

is the upper bound for $MaxCut$. The GAMC algorithm determines the base x^k from B over $O(m)$ time [11]. Therefore, determining $UpVal$ requires $O(mn) + n\log n) \approx O(mn)$ time, which completes the proof of the theorem.

In the case when G is a bipartite graph, it was shown above that $UpVal = MaxCut$, i.e., $UpVal$ is an accurate estimate. If G is an odd cycle, then $UpVal = MaxCut + 1$.

An odd cover of the arbitrary graph $G = (V, E)$ is the subset of edges $T \subset E$, after removing which the resulting subgraph does not contain odd cycles. Suppose $m(T)$ denotes the number of edges in T. It was shown in [6] that the set of edges $E \backslash T$ is a maximal cut if and only if $m(T)$ is a minimal number. In reality, this fact is nothing more than equality (see [11]) between the optimal values of the objective function of problems (1), (2) and dual with respect to it. For completeness, here we briefly describe the scheme of proof from [11] that $m - MaxCut = m_0$, where m_0 is the number of edges in the minimal odd cover of the graph G.

Suppose that the base x is determined with respect to the linear ordering $L = (V^+, V^-)$ of the vertices of the graph G. According to L, the edges of this graph can be oriented in such a way that the resulting digraph is an oriented acyclic one. To do this, replace each edge (v, u) with an arc (v, u) if $v \prec u$ or an arc uv if $u \prec v$ in L. The converse is also true, each acyclic orientation of the edges of the graph G determines the linear ordering of the vertices. By the determination of x by the

GAMC algorithm, it turns out that x_v is the number of arcs leaving the vertex v, and $y_v = d_v - x_v$ is the number of arcs entering the vertex v in an oriented acyclic graph. Consider the bipartite graph $H = (V^+, V^-, \delta(V^+))$ corresponding to the base x. It is clear that H contains no edges with final vertices in V^+ or V^-, i.e., by removing an edge with final vertices from V^+ or V^-, the cut $\delta(V^+)$ is determined. By the determination of the base x, it turns out that $y(V^+)$ and $x(V^-)$ are the number of the removed edges with final vertices in V^+ and in V^-, respectively, i.e., y $y(V^+) + x(V^-)$ is the total number of edges removed. Hence,

$$m_0 \le m - |\delta(V^+)| = y(V^+) + x(V^-).$$

In the case when $L = L_*$, for the base x^* and $y^* = d - x^*$ we obtain that

$$y^*(V^+) + x^*(V^-) = m_0. \quad (4)$$

A useful consequence follows from this equality. It is clear that for the optimal solution of problem (1), (2)

$$MaxCut = Cut(x^*) = x^*(V^+) - y^*(V^+) = y^*(V^-) - x^*(V^-).$$

Therefore,

$$\frac{MaxCut}{R} = 1 - \frac{r}{R}$$

for each case when $R = x^*(V^+)$, $r = y^*(V^+)$ or $R = y^*(V^-)$, $r = x^*(V^-)$. Since

$$y^*(V^+) + x^*(V^-) \le y(V^+) + x(V^-) \text{ and } x(V) = y(V)$$

for the base x determined by the GAMC algorithm, then $UpVal \ge R$ from the determination of $UpVal$. Therefore, it follows from the last equality that

$$\frac{MaxCut}{UpVal} \le \frac{MaxCut}{R} = 1 - \frac{r}{R} \le 1 - \frac{r}{UpVal}.$$

It follows from equality (4) that at least one inequality: $y^*(V^+) \ge m_0/2$ or $x^*(V^=) \ge m_0/2$ holds, i.e., $r \ge m_0/2$. Therefore,

$$\frac{MaxCut}{UpVal} \le 1 - \frac{m_0}{m_1},$$

where $m_1 = 2UpVal$. On the basis of this inequality, we prove the following result concerning the solution of an arbitrary ϵ–approximate greedy type algorithm (for the determination, see [13]).

Theorem 4. *An arbitrary ϵ–approximate greedy type algorithm, determines the optimal solution to the maximum cut problem with variables for each vertex $v \in V$, if $0 \le \epsilon \le m_0/m_1$.*

Proof. By solving the maximum cut problem with variables for each vertex $v \in V$, we determine a partition of the set V into two disjoint subsets so as to maximize the number of edges with final vertices from different subsets. Suppose that, using some ϵ–approximate greedy type algorithm, a cut that separates such subsets is determined. Suppose Cut is the value of this cut. It is clear that $UpVal$ is the upper bound for Cut. Therefore,

$$\frac{UpVal - Cut}{UpVal} \le \epsilon \le \frac{m_0}{m_1},$$

from which it follows that

$$\frac{Cut}{UpVal} \ge 1 - \frac{m_0}{m_1}$$

at $\epsilon \le m_0/m_1$. Therefore, $Cut = MaxCut$, which completes the proof of the theorem.

Known models contain variables x_v for all vertices $v \in V$. Therefore, edges with final vertices in one of these subsets also contribute to the objective function. For instance, if $G = C$ is a simple odd cycle, then C has the edge (u, v) with final vertices in V^+ or V^-. Therefore, $x_t = 1 < 2 = d_t$ and $y_t = d_t - x_t = 1$, where $t = v$ or $t = u$. In this case $UpVal = |E(C)|$ and $MaxCut = |E(C)| - 1$, i.e., $m_0 = 1$. For instance, GAMC, as ϵ is an ϵ–approximate greedy type algorithm,

determines the optimal solution to the maximum cut problem on C with a relative accuracy $\epsilon = 1/|E(C)|$. It is known that the greedy algorithm is a 0.50–approximate algorithm. Since $\epsilon \geq 0.50$, for all ϵ–approximate greedy type algorithms, depending on the number of odd cycles, the value of ϵ can change within the interval [0; 0.50], for the general case of the graph G.

It is clear that the determination of the number m_0 in the general case is an NP–hard problem. Therefore, verification of the fulfillment of the condition of the theorem for the solution x determined by the GAMC algorithm is fraught with certain difficulties. If it is possible to find a new solution to problem (1), (2) with a better value than $Cut(x)$, then it is clear that x is not the optimal solution. In [14], two algorithms are proposed for converting x to a different base for finding a new base with a better value than $Cut(x)$. The implementation of these algorithms requires determining the subset of vertices in each part of the bipartite graph corresponding to the base x. At the same time, the requirement that these subsets must be sets of vertices of cycles in G is ignored. We show in the following paragraphs that these complex algorithms for transforming one base into another can be replaced by one of the known algorithms for finding the shortest path, as a result of which the speed and accuracy of solving problem (1),(2) improves.

6. Even cycles and L_*

A cycle is a simple path (without repeating vertices), in which the initial and ending vertices coincide. If the number of vertices in the cycle is even, then this cycle will be called an *even cycle*. Let us show that to find the optimal solution x^* by the greedy algorithm it is sufficient only to determine L_* by ordering the vertices of the even cycles of the graph G. Based on this fact, we will also show that to improve the solution of problem (1), (2) determined by the GAMC algorithm, we can apply the shortest path algorithm.

Theorem 5. *To find the optimal solution x^* by a greedy type algorithm, it it sufficient to determine the optimal ordering L_* taking into account only the vertices of even cycles of the graph G.*

Proof. Let H_* be a bipartite graph corresponding to the optimal solution x^* of problem (1), (2). It is clear that all cycles in H_* are even. Let us show that even cycles in H_* are the unions of two simple even or odd cycles of the graph G. First, we show that an even cycle in G is either a simple cycle or the union of two simple even or odd (chordless) cycles containing edges that are not are common to them. Let C and T be two simple odd or even cycles. Let h be the number of common edges on C and T. Assume that the cycles C and T are odd with $2c+1$ and $2t+1$ edges, respectively. Then their union contains $2c+2t+2-2h$ even number of edges. Now assume that C and T are simple even cycles that have $2c$ and $2t$ edges, respectively. In this case, their union contains $2c+2t-2h$ even number of edges. Since the optimal solution x^* of problem (1), (2) corresponds to the same maximum cut in G and H_*, the optimal ordering L_* of the vertices of the graph G can be determined with respect to H_*, i.e. when determining L_*, only the order of the vertices of the even cycles of the graph G can be taken into account, taking into account their contribution to objective function (1).

By this theorem, in determining L_*, it is sufficient to find the order of the vertices v of even cycles taking into account the contribution of the variables x_v to objective function (1). In the general case, defining L_* in this way is equivalent to the problem of finding the minimum odd cover of a graph G, after removing the edge of which all cycles are even in G. It was noted above that the latter is an NP-hard problem. Let us show that one of the ways to determine L_* is to apply the shortest path algorithm as follows. Suppose that the base x and the subsets V^+, V^- are determined by the GAMC algorithm with respect to the linear ordering $L = (V^+, V^-)$ of the vertices. It is clear that the edges of the shortest path between two vertices from V^+ or V^- in the bipartite graph $H = (V^+, V^-, \delta(V^+))$ lie on an even cycle. Thus, in determining L_*, we take into account the order of the

vertices of the even cycles of the graph G. Let h_v be the degree of the vertex v in the bipartite graph H. Since the quantity $|\delta(V^+)|$ does not depend on the order of the vertices inside the subset V^+ and V^-, the degree of the vertices h_v, which means that the value h_v will not change for an arbitrary way of ordering of the vertices of these subsets. Based on this property of the vertices of the graph H, we prove the following result as a detailed presentation of the above method.

Lemma 2. *The optimal linear ordering $L*$ of the vertices, taking into taking into account G, can be determined using the shortest path algorithm between the vertices from V^+ or V^- in the graph H.*

Proof. It is not hard to see that the GAMC algorithm determines subsets V^+ and V^- such that $h_v > d_v - h_v$ for all vertices v from V^+ and V^-. Final vertices of some edges of odd simple cycles in G belong either to V^+ or V^-, i.e., they are absent in H. Obviously, if $H \neq H_*$, then by moving some vertices from V^+ to V^- and from V^- to V^+, we can determine the set of vertices of each part H_*. For definiteness, assume that the vertices $v, u \in V^-$ are to be moved to V^+, i.e., moving these vertices to V^+ will improve the value of (1). Here it is required that $(v, u) \notin E$. Indeed, suppose that $(v, u) \in E$. Cases are possible when V^+ contains at least one vertex w adjacent with v and u, or there are no adjacent vertices in V^+. In the first case, when moving v and u in V^+, it is clear that w must be moved to V^- in a different way, obviously, the value of (1) will decline. Then, since $h_w > d_w - h_w$ and $(v, u) \in E$, when moving w in V^-, the number of edges in the odd cover of the graph will also increase G, i.e., the value of (1) will decline. In the second case, it follows from $h_i > d_i - h_i$ for $i = v, u$ that the value of (1) will also decline.

Now consider the path $\pi = (v = v_1, \ldots, u = v_k)$ between the vertices v and u in H. Since k is an odd number, it is clear that v and u move to V^+ by including v_i in $V^+$and v_{i+1} in V^- for $i = 1, \ldots, k-1$, i.e., the vertices $\pi \cap V^+$ are moved to V^-, and the vertices from $\pi \cap V^-$ in V^+. The path π must be the shortest, otherwise, taking into account that $h_i > d_i - h_i$ for $i \in V$, it can be replaced by the shortest path without changing the value of objective function (1). The edges of the path π lie on some even cycle of the graph H. This cycle is also an even cycle of the graph G. Thus, in determining $L*$, even cycles of the graph G are taken into account. It is clear that iteratively improving the value (1) by including the vertices of the path π from one to another part of the current bipartite graph, we can determine $L*$.

According to this lemma, it is necessary to fix the vertices v and u in the proof of the lemma in the indicated way, the movement of which improves the value of objective function (1). In fact, finding such vertices is also a difficult task. Therefore, after determining the subset V^+ and V^- by the GAMC algorithm, the shortest path algorithm can be applied as follows. Let $\rho(v) = 2h_v - d_v$. All pairs of vertices $v, u \in V^+$ and $v, u \in V^-$, such that $(v, u) \notin E$, are ordered in ascending order of the value of the sum $\rho(v) + \rho(u)$. For simplicity of presentation, assume that $v, u \in V^-$ is the first pair of vertices in their ordering. Let π be the shortest path between vertices v and u in a bipartite graph in H. After moving a vertex from $\pi \cap V^+$ and $\pi \cap V^-$ to V^- and V^+, respectively, a new bipartite graph H_1 is constructed, thereby a new base x^1 is determined. If $Cut(x) > Cut(x^1)$, the shortest path between the next pairs of vertices in their ordering is determined. If $Cut(x) \leq Cut(x^1)$, this process is repeated again for $x = x^1$ and bipartite graph $H = H_1$.

A fairly straightforward way of estimating the time complexity of this algorithm is to cover all possible cases. Indeed, if the value of the function $Cut(x)$ does not improve during the entire operation of the algorithm, then the algorithm determines the shortest path no more than $O(n^2)$ times. In case of improvement, the value of $Cut(x)$ increases by at least a unit, i.e. the shortest path is determined no more than $O(m)$ times. Therefore, the time complexity of the algorithm is estimated as $O(Mn^2)$, where M is the complexity of finding the shortest path. In this case, the amount of required computing resources is estimated as M. These simple features of the algorithm contribute to its application for solving practical problems of high dimensionality.

7. Stable set and minimum vertex cover problems

Finding a stable set of vertices for a simple graph, at first glance, seems to be a very simple task and therefore can be solved efficiently. In fact, unlike the maximum cut problem, it is NP-hard on planar graphs. Moreover, it is NP-hard even for planar graphs with maximum vertex degree no more than 3 [15] or for planar graphs of large cover [16]. It is known that this problem can be solved in polynomial time only in some subclasses of planar graphs, such as outerplanar graphs or planar graphs of bounded chordality [17].

In addition to these differences, there are other differences between the constraints of their admissible solutions in the known models of these problems (see [7, 5]), although the models of both problems are formulated in terms of the theory of linear integer programming problems with 0,1 variables. As a rule, stable set models as linear problems with 0,1 variables are used only to determine the upper bound for the functional. To find its solution, mainly local search is used based on heuristics related to graph topology. For instance, it was noted in [12] that by solving the problem obtained after relaxation of the stable set model with the constraints

$$x_v + x_u \le 1, (v,u) \in E, x_v \ge 0, v \in V, \tag{5}$$

on simple odd cycles, a rough upper bound is determined. To cut off non-integer solutions inherent to odd cycles, the use of the so-called inequalities of odd cycles improves the upper bound. It is proved that these constraints and conditions (5) determine an integer polyhedron for perfect graphs, i.e., the stable set problem is polynomially solvable on these graphs. Obviously, only one vertex of the clique can be in the optimal solution of the stable set problem. Therefore, by solving linear models with the clique inequality [12], we can also improve the accuracy of the upper bound for the stable set problem.

The above inequalities are not correct for the maximum cut problem. Despite such differences between the problems of the maximum cut and the stable set of vertices, we will show that the set of feasible solutions of both problems is described by identical constraints, and the upper bound for the latter problem can be determined using $UpVal$.

Let the simple graph $G = (V, E)$ be given. In terms of the base of the polymatroid, the mathematical model of the stable set problem on G has the following form: find

$$MaxSat = \max\{Sat(x) = \textstyle\sum_{v \in V} \frac{1}{d_v} x_v\} \tag{6}$$

with the constraints

$$x \in P, \tag{7}$$

$$x \in \{0, d_v\}, v \in V. \tag{8}$$

Indeed, suppose that $I \subseteq V$ is some maximal stable set by inclusion in the graph G. Consider the linear ordering $L = (I, V \backslash I)$. Since I is a maximal stable set, each vertex from $V \backslash I$ is connected by an edge with at least one vertex from I. Therefore, it follows from the determination of the base $x \in P$ by the greedy algorithm that $x_v = d_v$ for $v \in I$ and $0 \le x_v < d_v$ for $v \in V \backslash I$, i.e., if we assume $x_v = d_v$ for $v \in V \backslash I$, then $x \notin P$. Since $x \ge 0$ for $v \in V \backslash I$, if we assume $x_v = 0$ then $x \in P$. This means that $Sat(x) = |I|$.

It follows from the formula for determining the arbitrary base $x \in P$ by the greedy algorithm that in an arbitrary linear ordering, if the vertex v is not adjacent to all vertices such that $w \prec v$, then $x_v = d_v$. Therefore, constraints (7) and (8) can be written in the equivalent form

$$x(S) = |\delta(S)| = d(S), S \subseteq V. \tag{9}$$

Therefore, finding the maximum cut satisfying condition (9) is equivalent to the stable set problem. Taking into account that the problem of the stable set is NP-hard on planar graphs, the following result is valid.

Theorem 6. *The maximum cut problem with additional conditions (9), on planar graphs, is NP-hard.*

It follows from equality (9) that for some cases of the graph G the maximum stable set can be determined by solving the maximum cut problem. For instance, on simple cycles, the solution to these problems can be determined by the greedy algorithm with respect to L_*. To prove the NP-hardness of the problem of determining the maximal induced bipartite subgraph in the weighted graph G, the authors of [18] transform it into a stable set problem. It is easy to show that problem [18] is also reduced to the maximum cut problem. This means that the same algorithm can be used to find $MaxSat$ and $MaxCut$ in special graphs.

It is known that the problems of the stable set and the vertex cover are dual in the sense that if $I \subseteq V$ is the optimal solution to the former, then the complement I is the optimal solution to the latter. Using this fact, we can formulate the last problem also in terms of the base of the polymatroid P. Indeed, since

$$n - \sum_{v \in V} \frac{1}{d_v} x_v = \sum_{v \in V} \frac{d_v - x_v}{d_v},$$

the model of the vertex cover problem can be formulated as follows: find

$$\min \sum_{v \in V} \frac{d_v - x_v}{d_v},$$

with constraints (7) and (8). Many textbooks present an approximate algorithm for solving the vertex cover problem. There are also examples showing that this algorithm is not ϵ-approximate (see Chapter 17 in [13]). However, to search for a solution to the problems of stable set and vertex cover, we can use the base x determined by the GAMS algorithm. Consider a bipartite graph $H = (V^+, V^-, \delta(V^+))$. In the induced subgraphs we can easily define the maximum independent sets I^+ and I^- of vertices by the sets V^+ and V^-. If $|I^+| > |I^-|$ then $I = I^+$, otherwise $I = I^-$. Here, it is clear that the set $V \backslash I$ is a vertex cover.

8. Upper bound for $MaxSat$

The upper bound for $MaxSat$ can be calculated by solving the linear problem obtained after relaxation of constraints (8) as $0 \le x_v \le d_v$ for all $v \in V$. However, it follows from the monotonicity of the function f that the condition of non-negativity of the variables is satisfied for all x from P. In addition, it also follows from the determination of the base $x \in P$ that the condition $x_v \le d_v$ is satisfied for all bases x from P. Therefore, the relaxation of problem (6)-(8) has the following form: find

$$\max \sum_{v \in V} \frac{1}{d_v} x_v \tag{10}$$

with the constraints

$$x \in P. \tag{11}$$

This problem is solved very easily by a greedy algorithm. To do this, we first need to order the values $\frac{1}{d_v}$ in ascending order of their values, i.e., determine the linear ordering L such that if $\frac{1}{d_v} \ge \frac{1}{d_u}$, then $\frac{1}{d_v} \ge \frac{1}{d_u}$. The greedy algorithm determines the optimal solution to this problem with respect to L. Thus, the upper bound for $MaxSat$ can be found over $O(mn\ln n)$ time.

For even cycles, it is easy to show that the optimal value (10) $\frac{m}{2} = MaxSat$, i.e. the upper bound is accurate in this case. However, for the general case of graph G, optimal value (10) can be very different from $MaxSat$. Let us show that, in the general case, the upper bound can be determined by solving the following problem: find

$$\max \sum_{v \in V} \frac{1}{d_v} x_v^+ \tag{12}$$

with the constraints

$$z \in EP, \tag{13}$$

$$0 \le x_v^+ = \max\{0, z_v\}, v \in V. \tag{14}$$

Statement 1. *Optimal value (12) is the upper bound for problem (6)-(8).*

Proof. Let x be the optimal solution to problem (6)-(8). Then $x_v = d_v$ for $v \in I$ and $x_v = 0$ for $v \in V \backslash I$, where I is the maximum independent set of vertices. Consider a linear ordering $L = (I, V \backslash I)$ of vertices. With respect to $L = (I, V \backslash I)$, the greedy algorithm determines the value of the variables $z_v = d_v$ for $v \in I$. For the vertex $v \in V \backslash I$, the number z_v can be positive or negative. Therefore, $x_v^+ = d_v$ for $v \in I$ and $x_v^+ = \max\{0, z_v\} \ge 0$ for $v \in V \backslash I$ is an admissible solution to problem (12)-(14), which proves the statement.

For example, consider the stable set problem on the graph $G = W_7$, where W_7 is a seven-vertex wheel. Let $x_v = 3$ for $v \in I_3$, where I_3 is any stable set with three vertices, and $x_v = 0$ for all $v \in V \backslash I_3$. It is easy to check that x is an optimal solution to problem (12)-(14) and (6)-(8), i.e., the solution to problem (12)-(14) determines the exact upper bound when $G = W_7$.

According to (9), the upper bound for $MaxSat$ can be determined over $MaxSat$ time as follows. It is clear that $|\delta(S)| \le MaxCut$ for the cut $\delta(S)$ determined by the maximum stable set S in (9). Therefore

$$MaxSat \le \frac{MaxCut}{d_0} \le \frac{UpVal}{d_0},$$

where d_0 is the minimum degree. In particular,

$$MaxSat \le \frac{UpVal}{d}$$

for regular graphs, where d is the degrees of vertices. A more accurate estimate

$$MaxSat \le \frac{MaxCut}{d}$$

can also be calculated in polynomial time for planar regular graphs.

9. The clique problem

Finding the maximum clique in the given simple graph $G = (V, E)$ is as important a practical problem as the problems considered in the previous sections. Let Clq be an arbitrary clique with vertex set $\emptyset \ne S \subseteq V$. Since the subgraph induced with the vertex set S is complete, then$|S|(|S| - 1)/2$ is the number of edges in this subgraph. Hence, to determine the maximum clique, it is required to find the subset $\emptyset \ne$ S⊆V with the maximum number of edges. Therefore, the model of the problem in terms of polymatroids can be formulated as follows: find

$$MaxClq = \max\{Clq(S) = |S|\} \tag{15}$$

with the constraints

$$|S|(|S| - 1)/2 = g(S), \emptyset \ne S \subseteq V. \tag{16}$$

In contrast to the model of the problems considered in the previous sections, the given model of the clique problem contains quadratic constraints (16). Since g is a monotone function by definition, $g(S) > g(T)$, for all $T \subset S$, where S is the vertex set of the maximal clique. Taking into account the fact that there are polynomially bounded algorithms for maximizing a submodular function (see [8]), the search for solutions to (15) and (16) can be started from the opposite, i.e., determine S_1 as a solution to the problem

$$\max\{g(S); \emptyset \ne S \subseteq V\}.$$

If (16) is not true for S_1, then solve this problem under the additional condition $S \neq S_1$, etc. All approximate algorithms work according to this scheme. However, the search for a solution to (15) and (16) can be carried out on the basis of simple properties of cuts in G. For this, we prove the following result.

Theorem 7 . *For an arbitrary nonempty set $S \subset V$, one of the following conditions is true. The first case is when*

$$|\delta(S)| = d(S) - |S|(|S| - 1). \quad (17)$$

In this case S is the vertex set of the clique in the graph G. The second case is when

$$|\delta(S)| > d(S) - |S|(|S| - 1). \quad (18)$$

In this case S is not the vertex set of the clique in the graph G.

Proof. Suppose that the set S satisfies conditions (17). It follows from the definition of the function $f(S)$ and $g(S)$ that $f(S) + g(S) = d(S)$ and $|\delta(S)| = f(S) - g(S)$ for arbitrary $S \subseteq V$. Therefore, condition (17) and the equality $|\delta(S)| = d(S) - 2g(S)$ imply that the subgraph induced by the set S is complete.

It is clear that $g(S) \leq |S|(|S| - 1)/2$ by the definition of the function g. Therefore, if the set S does not satisfy conditions (17), then condition (18) is satisfied. Since the cut $\delta(S)$ separates the sets S and $\overline{S}$, then $g(S) < |S|(|S| - 1)/2$. This means that for an arbitrary nonempty set $S \subset V$ either condition (17) or (18) is satisfied. The last inequality also implies that the subgraph induced by the set S is not complete, which completes the proof of the theorem.

The cut $\delta(S) = d(S) - |S|(|S| - 1)$ will be called *the cut of the maximum clique*. This theorem implies that $MaxCut \geq |\delta(S)|$. Therefore, $UpVal$ is the upper bound for the cut of the maximum clique, which is also determined over $O(mn)$ time. On the basis of this theorem, the solution to the clique problem can be found by the following algorithm.

Let $\delta(S_*)$ be the minimal cut in the graph G with unit edge capacities. If $S_* = \{v\}$, then condition (17) with $S = \overline{S_*}$ implies that either $\overline{S_*}$ or V are the vertex sets of the maximum clique. Therefore, let condition (18) be true for $\overline{S_*}$.

In this case, the set $S \subset V$ is determined as follows. With respect to the linear ordering $S \subset V$, there is a base x of the polymatroid P such that

$$d_u - x_u = y_u = u - 1 \text{ for } u = 1, \dots, k \quad (19)$$

for the maximum possible number k of vertices u, where $y = d - x$. All vertices u are included in $S \subset V$, i.e., the set S is maximal with respect to inclusion. Moreover, it follows from (19) that condition (17) is true for the set S. Hence, S is the vertex set of the clique in the graph G. Further, among the vertices from S, the vertex w with the maximum value x_w is selected. All vertices from S are labeled, except for w. A new linear ordering L is determined, where w is the first vertex and the unlabeled vertices precede the labeled vertices. With respect to L, a base $x^1$is defined such that, the maximum possible number of vertices $k_1$satisfy condition (19) for $y^1 = d - x^1$. All vertices are included in $S_1 \subset V$, from S_1 the vertex $w_1 \neq w$ with the maximum value x_{w_1} is selected. After that, the process continues again with respect to the vertex $w = w_1$ of the subset $S = S_1$.

Now suppose $|S_*| > 1$. Here, if $|S_*| = 2$, then v, u, w are the vertex set of the clique (triangle), since $\delta(S_*)$ is the minimal cut, where w is an adjacent vertex with $v, u \in S_*$ and $(v, u) \in E$. Therefore, let us assume that $|S_*| \geq 3$. If condition (17) is true for S_* or $\overline{S_*}$, then a clique with a vertex set S_* or $\overline{S_*}$ is determined. Suppose that condition (18) is true for S_*, then among the vertices from S_* the vertex v is selected for which x_v is minimal. Since $\delta(S_*)$ is the minimum cut, then $x_v < y_v$ for the vertices $v \in S_*$. Therefore, in the graph G, the vertex v cannot be the vertex of a clique with a set S such that $|S| \geq |S_*|$. Therefore, the algorithm continues its work on the subgraph

obtained after removing the vertex v from G.

10. The Hamiltonian cycle problem

A cycle passing through all vertices of a simple graph $G = (V, E)$ exactly once is called Hamiltonian. Note that the proof of the NP-completeness of the Hamiltonian cycle problem follows from the transformation sequence, where the vertex cover problem is polynomially transformable into a Hamiltonian cycle problem for directed graphs, the latter, in turn, is polynomially transformable into a Hamiltonian cycle problem for undirected graphs [19]. The mathematical models of the Hamiltonian cycle problem usually contain variables for each edge. In contrast to these models, the definition of a cycle that goes through all vertices exactly once, in terms of permutations, can be considered a model with variables for each vertex. Such a model includes the definition of a Hamiltonian path between some adjacent vertices from E. Such paths can be considered as linear orderings of vertices, relative to which the greedy algorithm determines the bases corresponding to Hamiltonian cycles. In other words, there is a one-to-one correspondence between some bases of the polymatroid P and Hamiltonian cycles in the graph G. Based on this correspondence, a new model of the traveling salesman problem in terms of the extended polymatroid EP is proposed in [20]. In what follows, for brevity, a Hamiltonian cycle will be called a tour.

Let P_0 be the set of bases P which correspond to Hamiltonian cycles in G. We call a base x∈P *bypass* if $x_v = d_v$, $y_v = 0$ and $x_u = 0$, $y_v = d_u$, where v, u are the final vertices of some edge from E, and $x_w \geq 1$, $y_w \geq 1$ for the remaining vertices of the graph G. Now we will show that P_0 is a subset of the set of bypass bases from P.

Statement 2. *If the polymatroid P does not contain a bypass base, then there is no tour in the graph* G.

Proof. Suppose that P does not contain a bypass base, but there is a tour C in the graph G. Let L be a linear ordering, where the order of the vertices is the same as in the Hamiltonian path π. Suppose that the edges of the graph are oriented using a linear ordering L, as shown in Section 2. In an acyclic directed copy of the graph G (see Section 2), $x_v = d_v$, $y_v = 0$ and $x_u = 0$, $y_v = d_u$, since v is the first and u is the last vertex in L. Moreover, according to the orientation of the arcs on C, there is one arc outgoing from w and another one entering the vertex w, for all vertices $w \neq v, u$. This means that the greedy algorithm will determine a bypass base that with respect to L, which contradicts the condition of the statement.

Note that this simple statement does not imply the existence of a tour if the polymatroid P contains a bypass base. For instance, if the graph G consists of three disjoint edge paths between the vertices v and t, where $(v, t) \notin E(G)$, it is not difficult to show that the polymatroid P contains a bypass base, and there are no tours in G. In other words, P_0 is a subset of the set of bypass bases of the polymatroid P.

Obviously, if the graph G contains some tour, then the degree of the vertices $b_w = d_w - 2$ in the subgraph obtained after removing the edge of the tour. However, the opposite is not true. For instance, let the graph G contain two disjoint cycles and one edge connecting the vertices of these cycles. Despite the fact that $b_w = d_w - 2$ in the subgraph obtained after removing the edges of these cycles, there is no tour in G. However, we can remove two incident edges to each vertex $w \in V$ so that if $b_w = d_w - 2$ in the resulting subgraph, then it can be argued that G contains some tour. Consider an acyclic oriented copy of the graph G obtained after orienting the edge with respect to some linear ordering L of the vertices in Section 2 in the indicated way. The arcs outgoing from any vertex $w \in V$ are called *arcs* x, and the arcs entering an arbitrary vertex are called *arcs* $y = d - x$.

Theorem 8. *Let x be the bypass base of the polymatroid P, determined with respect to the linear ordering L of the vertices from V, and $y = d - x$. Suppose that the following arcs are removed from the acyclic directed copy of the graph G:*

1) two arcs x with an initial vertex v and an arc y with the final vertex u, v and u being adjacent vertices;

2) one arc x and y with the final and initial vertices w, respectively, $w \neq v, u$.

The graph G contains a tour if and only if $b_i = d_i - 2$ in the resulting subgraph for all vertices $i \in V$.

Proof. Therefore, we prove the part of the theorem. Note that if there exists a Hamiltonian path π between the vertices v and at least one vertex u adjacent to v, then (v, u) and the edges of this path lie on a tour in the graph G. Let the bypass base x of the polymatroid P determined with respect to some linear ordering L of vertices, where $w \neq v, u$ from V and $v \prec w$ and $w \prec u$. After removing arcs in 1 and 2 in the indicated way, let the resulting subgraph G_{vu} not contain the edge (v, u). Then, according to 1, the edge (v, u) is one of the distant arcs x or y with the initial and final vertices v, respectively. Therefore, according to 2, it is necessary to remove one arc x and y with the final and initial vertices w ≠ v, u, respectively. Otherwise, either two x arcs with the initial vertex w, or two y arcs with the final vertex w will be removed. Suppose $b_i = d_i - 2$ for all vertices i in the resulting subgraph G_{vu}. Consider any cycle C. Suppose that in L the vertices of the cycle C ordered in a row. Assume that $d_i \geq 3$ for some vertex i of the cycle C. If the removed arcs are C arcs, then we remove two arcs x with the initial vertex i, if C is an even cycle, or we remove two arcs y with the final vertex i, if C is an odd cycle. This contradicts 2. Now suppose in L the vertices of the cycle C are not ordered in a row, i.e., some vertices from C follow the vertices of another cycle F in L. In this case, it is not possible to remove in 2all arcs C or F in the indicated way, we remove two arcs x or y with some initial or final vertices of the cyclev C and F. Therefore, the removed arcs lie on a simple path connecting these cycles C and F. Since in L, the vertices of arbitrary cycles can be ordered in a row or cannot, then it follows from $b_i = d_i - 2$ that some tour exists in G.

According to this theorem, the main procedure for determining a tour is finding a bypass base x of a polymatroid P or the same as a linear ordering L. Obviously, such a base can be determined using an algorithm for finding a Hamiltonian cycle in undirected graphs [21]. However, the development of an efficient algorithm for the problem of recognizing a Hamiltonian cycle (see [3]), based on this theorem, remains relevant.

11. Conclusion

The mathematical formulation of the considered problems, in terms of polymatroids, has made it possible to reveal the functional connections between their admissible solutions. At the same time, the main emphasis is made on a detailed study of the maximum cut problem, since in this functional connection the connecting knot is the upper bound for $MaxCut$. In addition, genuine simple implementation-wise computational algorithms are proposed to solve the problems under consideration. Since these algorithms use combinations of simple procedures with known polynomial algorithms, as well as an acceptable amount of required memory, it is possible to guarantee their efficiency in solving practical problems of high dimensionality. The material of the paper requires some mathematical maturity and knowledge of combinatorial optimization and the theory of polymatroids. Experience shows that the best way to acquire such knowledge is to conduct a computational experiment on test problems. The authors are ready to work closely with specialists interested in solving test and real problems by applying the proposed algorithms.

References

[1] F. Sharifov, Perfectly matchable subgraph problem on a bipartite graph, RAIRO-Oper. Res. 44 (2010) pp.27-42.
[2] F.A. Sharifov, Perfect matching and Polymatroids, Cybernetics and Systems Analysis. 53 No.3 (2018) pp.753-758.
[3] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J.W. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press. (1972) pp.85-103.
[4] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, (1979).
[5] G. Nemhauser, L.A. Wolsey, Combinatorial Optimization, Wiley-Interscience, New York, (1988).
[6] F.O. Hadlock, Finding a maximum cut of a planar graph in polynomial time, SIAM J. Comput. 4 (1975) pp.221-225.
[7] Ahmad Sharieh, Wagdi Al Rawagepfeh, Mohammed H. Mahafzah, Ayman Al Dahamsheh, An Algorithm for Finding Maximum Independent Set in a Graph, European Journal of Scientific Research. ISSN 1450–216X. 23 No.4 (2008) pp.586-596.
[8] A. Schrijver, Combinatorial Optimization, Polyhedra and Efficiency Algorithms and Combinatorics, Springer. 24 Chapter 69 (2003) pp.1208-1217.
[9] A. Hertz, Polynomially solvable cases for the maximum stable set problem Discrete Appl. Math. 60 (1995) pp.195-210.
[10] F.A. Sharifov, Finding the Maximum Cut by the Greedy Algorithm Cybernetics and Systems Analysis. 54 (2018) pp.737-743.
[11] F. Sharifov, L. Hulianytskyi, Cuts in Undirected Graphs. I, Cybernetics and Systems Analysis. 56 (2020) pp.559-565.
[12] M. Grotschel, L. Lovasz, A. Schrijver, Geometric algorithms and combinatorial optimization, Springer-Verlag, Berlin, (1988).
[13] Х. Пападимитриу, К. Стайглиц, Комбинаторная оптимизация, Алгоритмы и сложность, Москва “Мир”, (1985) p.510. [In Russian: H. Papadimitriou, K. Steiglitz, Combinatorial Optimization, Algorithms and Complexity, Moscow, Mir]
[14] F. Sharifov, L. Hulianytskyi, Cuts in Undirected Graphs. II, Cybernetics and Systems Analysis. 56 (2020) pp.745-752.
[15] Kanj Iyad A. and Zhang Fenghui, (2010) Independent Set on graphs with maximum degree 3. https://via.library.depaul.edu/tr/14
[16] H.A. Bodlaender, Partial k-Arboretum of Graphs with Bounded Treewidth, Theor. Comput. Sci. 209 (1998) pp.1-45.
[17] M. Lu, H. Liu and F. Tian. Laplacian spectral bounds for clique and independence numbers of graphs, J. Combinatorial Theory B. 97 No.5 (2007) pp.726-732.
[18] Denis Cornas and A. Ridha Mahjoub, The maximum induced bipartite subgraph problem with edge weights, SIAM J. Discrete Math. 21 No.3 (2007) pp.662-675.
[19] Ахо А. Хопкрофт, Дж. Ульман Дж., Построение и анализ вычислительных алгоритмов, Москва, Мир, (1979) p.535. [In Russian: Aho A. Hopcroft, J. Ullman J., The Design and Analysis of Computer Algorithms, Moscow, Mir]
[20] H. Kutucu, Bases of polymatroids and problems on graphs, Turkish Journal of Electrical Engineering Computer Sciences. 28 (2020) pp.1905-1915.
[21] Н. Кристофидес, Теория графов, Алгоритмический подход, Москва, Мир, (1978) p.432. [In Russian: N. Christofides, Graph Theory: An Algorithmic Approach, Moscow, Mir]