Sigma J Eng & Nat Sci 36 (3), 2018, 835-848

Sigma Journal of Engineering and Natural Sciences
Sigma Mühendislik ve Fen Bilimleri Dergisi

Research Article

LINEAR PROGRAMMING PROBLEMS WITH FUNDAMENTAL CUT MATRICES

Firdovsi SHARIFOV[1], Hakan KUTUCU*[2]

[1]*V.M.Glushkov Institute of Cybernetics, Kiev-UKRAINE;* ORCID: 0000-0001-8768-3649
[2]*Karabük University, Department of Computer Engineering, KARABUK;* ORCID: 0000-0001-7144-7246

Received: 05.04.2017 Revised: 26.10.2017 Accepted: 11.04.2018

ABSTRACT

In the paper, we consider a linear programming problem with con-straint matrices whose rows are 0, 1 characteristics vectors of fundamental cuts in a given undirected graph $G = (V, E)$. We prove that the simplex algorithm finds an optimal solution in at most m-n+1 (m = $|E|$, n = $|V|$) iterations. We also consider the question whether a given binary matrix is a 0, 1 characteristic vector of fundamental cuts in the graph G.
Keywords: Network design, submodular function, fundamental cut sets, linear programming.
MSC 2010: 90C35, 05B35, 90C05

1. INTRODUCTION

Let $G = (V, E)$ be a connected, simple and undirected graph with node set V and edge set E. Throughout the paper, we denote the number of nodes in G by n, that is, $n = |V|$ and denote the number of edges in G by m, that is, $m = |E|$. Let T be a spanning tree of G. We denote the edges of T by $t_1, \ldots, t_{n-1}$. Removing any edge t_k splits T into two subtrees whose node sets are denoted by V_k and $\overline{V_k} = V \backslash V_k$. The cut separating V_k and $\overline{V_k}$ in the graph G is called the fundamental cut (FC) corresponding to t_k. We use $\delta(S)$ to denote the cut separating a subset $\emptyset \neq S \subset V$ and $\bar{S} = V \backslash S$ and use $x\big(\delta(S)\big) = \sum\{x_e; e \in \delta(S)\}$. The set $\mathcal{F}(T) = \{V_1, \ldots, V_{n-1}\}$ is called fundamental cut sets of the spanning tree T. Let A be an $(n - 1) \times m$ binary matrix whose columns are indexed by the edges of G and rows are 0-1 characteristic vectors of the fundamental cuts of G corresponding to edges of T. This matrix is called *FC* matrix. In this paper, we show that the simplex algorithm finds an optimal solution to a linear programming problem with an *FC* constraint matrix and a non-negative right-hand side vector in at most $m - n + 1$ iterations. We also give some characteristics of an *FC* matrix.

Despite the many recent approaches, the simplex method still remains to be one of the most efficient algorithm in practice. Computational tests show that the expected number of pivot steps (simplex algorithm iterations) for solving the majority of practical problems is a linear function of the number of variables [2], [10], [13]. However, there are examples (see [7]) for which the number of the simplex pivot steps is exponential. Determining whether the number of the simplex

* Corresponding Author: e-mail: hakankutucu@karabuk.edu.tr, tel: (370) 418 70 50 / 3870

method pivot steps can be bounded by a polynomial of the input size is a very difficult problem in general.

The *FC* matrices defined above are viewed as representations of matroids in combinatorial theory. A representation of a matroid *M* is a matrix *A* with entries over some field such that there is a one-to-one correspondence between the columns of *A* and the ground set of *M*, and a set of columns in A are linearly independent (as vectors) iff the corresponding set is independent in *M*. If *M* can be represented by a matrix *A* with entries over the two-element field, *GF*(2), then it is called a binary matroid. When the matrix *A* is totally unimodular, *M* is called a regular matroid. It is well known that the graphic matroid *M* = (*E*, *F*) and its dual matroid are regular, where *E* is the edge set of *G* and *F* contains any forest, in particular, any spanning tree in *G*. Any spanning tree *T* in *G* is a base of the graphic matroid *M* = (*E*, *F*). Thus, an *FC* matrix is the representation of the graphic matroid *M* = (*E*, *F*) over the field *GF*(2) with respect to its base *T*. Many other results on binary and regular matroids can be found in [9], [12], [14]. Contrary to the regular matrix representation of *M*, its *FC* matrix is not unimodular, in general. For example, consider the graph *G* in Figure 1, which includes a spanning tree *T* shown as bold edges. It is easy to see that the following matrix

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

is an *FC* matrix of *G*, whose columns indexed by edges (1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4) from left to right, and rows are 0-1 characteristic vectors of the fundamental cuts corresponding to edges (1, 2), (2, 3), (2, 4) of the tree *T*. It is not difficult to see that this matrix is a representation of the graphic matroid *M* = (*E*, *F*) of the graph *G* in Figure 1 with respect to the base *T*. However, it is not a unimodular matrix, since the determinant of the submatrix

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

equals to 2 ≠ -1, 1. Here, we should mention, as in the case of signed matrices [14], after replacing some of 1's with -1 in each column, the matrix above will be totally unimodular. However, in this paper, we consider only *FC* matrices with 0 and 1 entries.

Now, consider another spanning tree T_0 of the graph *G* in Figure 1 with edges *a* = (1, 4), *b* = (3, 4) and *c* = (2, 3). Let e_{ij} denote the edge connecting vertices *i* and *j*. For the graph *G*, the following *FC* matrix with respect to the tree T_0

$$\begin{array}{c} \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{cccccc} e_{12} & e_{13} & e_{14} & e_{23} & e_{24} & e_{34} \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

is a totally unimodular matrix, since all 1's in each column are consecutive [10]. It is easy to see that the tree T_0 is a hamiltonian path between the nodes 1 and 2. In Section 4, we show that an *FC* matrix of some graph with respect to a hamiltonian path (spanning tree) is totally unimodular. Special cases of the graph *G* and the right-hand side vector allow to get an integer solution to above mentioned network design problems by solving linear programming problems with *FC* constraint matrices iteratively for different spanning trees.

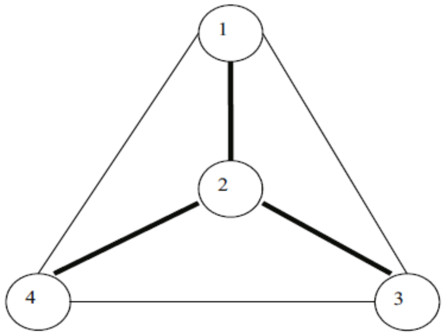

Figure 1. A spanning tree T of graph G.

Let I be an identity square matrix and let N be a binary matrix, where the number of rows of I and N are the same. As an inverse question, in Section 4, we will also prove that if each column of N has only two nonzero entries, then $A = (I, N)$ is an FC matrix, which is the representation of some graphic matroid with respect to a star-spanning tree base. If N has odd number of rows and columns (the odd cycle), then $A = (I, N)$ is not a balanced matrix by the definition [1]. As characterization of FC matrices, we prove that linear programming problems with an FC constraint matrix is strongly polynomial solvable. Thus, determining whether a given matrix in the form $A = (I, N)$ is an FC matrix presents some interest.

The paper is organized as follows. Section 2 includes a brief description of some network design problems in solving which linear programing (LP) problems with FC matrices are used as subroutine. In Section 3, we show that some linear programming problems can be written as in terms of the fundamental cuts. Then, in Section 4, we give the complexity of the simplex algorithm for these linear programming problems. We discuss some characterizations of an FC matrix in terms of special submatrices in Section 5. Finally, we give some concluding remarks and some directions for future work in Section 6.

2. NETWORK DESIGN PROBLEMS: MOTIVATION

In [8] and [11], the authors introduced the following general minimum cost network design problem (NDP) on a given graph $G = (V, E)$,

$$\min \sum_{e \in E} c_e x_e \,,$$

subject to

$$x(\delta(s)) \geq f(S), \quad \forall\, \emptyset \neq S \subset V,$$
$$x_e \geq l_e \geq 0, \quad \forall\, e \in E,$$
$$x_e \in \mathbb{Z}, \quad \forall\, e \in E,$$

where $f(S)$ is a symmetric submodular function defined on the subsets of V.

The theory of linear programming [10] states that a linear program with exponential number of inequalities can be solved in polynomial time if and only if the separation problem associated with these inequalities is polynomially solvable. This result has theoretical consequences. For solving network design problems, approximation solutions and polyhedral approaches use the combination of separation algorithms and linear programming solvers. The difficulty with this approach to solve NDP is that the separation problem

$$\Delta(x, f) = \min\{x(\delta(s)) - f(S);\ \emptyset \neq S \subset V\}$$

is *NP*-hard, for reason that when *f*(*S*) is a cut function as a special case of symmetric submodular functions, $\Delta(x, f)$ is the well known maximum cut problem. However, $\Delta(x, f)$ is reduced to the minimum cut problem when *f*(*S*) is a modular function [11].

In order to use the same techniques in solving NDP, it is necessary to find the capacity x_e of edges $e \in E$ and then to test whether the capacities x_e satisfy the constraints for any $\emptyset \neq S \subset V$. Therefore, we fix some spanning tree *T* in the graph *G* and set $b_t = f(V_t) - l\big(\delta(V_t)\big)$ for each *t* in *T*, then the capacity $x_e = y_e + l_e$ of edges $e \in E$ can be defined by solution $y = (y_e; e \in E)$ of the linear program $\varphi_1(T)$ (see Section 3) with the *FC* constraint matrix. In order to test whether the capacities x_e satisfy the constraints for any $\emptyset \neq S \subset V$, one can use a combination of the branch and bound techniques and results derived in solving $\Delta(x, W)$ for a modular function W approximated *f*. If the answer is yes and all capacities are integer, then the algorithm terminates. If any constraint for the cut (determined by some node subset) is violated, then the algorithm finds a new spanning tree T_1 whose fundamental cut set contains this subset and again find an optimal solution to $\varphi_1(T_1)$. This process iterates until all constraints hold.

Another network design problem is the optimum communication spanning trees (OCST) [5] which can be used as subproblem to solve linear programing problem with FC matrices. Let a set *V* of *n* nodes and a set of requirements r_{vw} for pair of distinct nodes *v* and *w* in *V* be given. The OCST problem is to build a spanning tree connecting these *n* nodes such that the total cost of the spanning tree is minimum among of all spanning trees. The cost of communication for a pair of nodes *v* and *w* is r_{vw} multiplied by the sum of the distances c_{ij} of edges on the unique path connecting *v* and *w* in the spanning tree. The cost of a spanning tree is the sum of communications for all pair of nodes in the spanning tree. In [5], it is noted without a proof that the OCST problem is hard to solve, in general.

Let *G* = (*V*, *E*) be a complete graph on *n* nodes with weight c_{ij} of edges and let *R* = (*V*, *E*(*R*)) denote a graph of given requirements, that is, $(i, j) \in E(R)$ if $r_{ij} > 0$. Let $\delta_R(S)$ denote a cut in the graph *R* determined by subset $\emptyset \neq S \subset V$. The OCST problem on graph *G* can be formulated as follow to find a spanning tree T_* of minimum cost.

$$c(T_*) = \sum_{e \in E} c_e x_e \longrightarrow min\,,$$

subject to

$$x_e = r\big(\delta_R(V_e)\big), \;\; V_e \in \mathcal{F}(T_*).$$

For graphs *G* with cyclomatic number $\upsilon(G) = m - n + 1$ bounded by some small constant, the spanning tree T_* can be found by enumerating all spanning trees in *G*.

In order to solve the OCST problem in general, one can use the following observations on the problem of OCST. The first observation is that, for a spanning tree T_0 of the graph *G*, as a feasible solution to the OCST problem, the vector *y* with components $y_t = b_t$ for each $t \in T_0$ and $y_e = 0$ for all $e \in E\backslash T_0$ is the initial basic feasible solution to the problem $\varphi_1(T_0)$ with the right hand vector $b = (b_t = r\big(\delta_R(V_t)\big);\;\; t \in T_0)$ and *y* satisfies the above constraints when $T_* = T_0$.

Let $\mathcal{T}(T_0)$ denote the set of spanning trees derived from tree T_0 by adding any edge $e \in E\backslash T_0$ to T_0 and deleting an edge $t \in T_0$ on the unique cycle in $T_0 \cup \{e\}$. The second observation is that if the basic solution $y = (y_e; e \in E)$ is an optimal solution to $\varphi_1(T_0)$, then $c(T_0) \leq c(T)$ for any $T \in \mathcal{T}(T_0)$. Note that heuristics based on Proposition 2 and 3 (see Section 3) can be used for defining the spanning tree T_0.

It easy to see that the inequality $c(T) \leq \varphi_1(T)$, as cutting plane, allows to eliminate $O(n\upsilon(G))$ spanning trees from the set of feasible solutions. Taking into account $\upsilon(G) = O(n^2)$ for a complete graph *G*, the use of inequality $c(T) \leq \varphi_1(T)$ essentially reduces the number of branching iterations for solving the OCST problem on *G*.

Due to the second observation, we get a generalization of the following result. For the case when r_{ij} is any non-negative number and $c_{ij} = 1$ for all edges (i,j), it was shown that the well-known Gomory and Hu algorithm constructed spanning tree *T(GH)* of the complete graph with the capacity r_{ij} of edges is an OCST [5]. Indeed, from Proposition 2, it follows that if non-negative cost of edges satisfy (8) with respect to $T(GH)$, then the above mentioned intial basic feasible solution for $T(GH) = T_0$ is the optimal solution to the problem $\varphi_1(T(GH))$ and $c\big(T(GH)\big) \leq c(T)$ for any spanning tree T in G, that is, $T(GH) = T_*$.

Moreover, by the first observation, the condition that a spanning tree T itself is an optimal solution to $\varphi_1(T)$ as the fixed point mapping $\varphi_1(T) \rightarrow T$ allows to apply the theory of fixed point mappings to solve the OCST problem. We leave this future work.

3. LP PROBLEMS WITH AN FC MATRIX

Consider the following linear programming problem

$$\min cx,$$

subject to

$$Ax = f,$$
$$x \geq l \geq 0,$$

for a given vectors c, $l \in R^E$ and $f \in R^T$, where $l \geq 0$, $f \geq 0$. Let T be a spanning tree with edges $t_1, \ldots, t_{n\text{-}1}$. This LP problem can be written as the following problem in the terms of the fundamental cuts $\delta(V_k)$ for all $V_k \in \mathcal{F}(T)$.

$$\min \sum_{e \in E} c_e x_e ,$$

subject to

$$x\big(\delta(V_k)\big) = f_k, \quad \forall\, V_k \in \mathcal{F}(T),$$
$$x_e \geq l_e \geq 0, \qquad \forall\, e \in E.$$

It is convenient to write it in the variables $y_e = x_e - l_e$ as follows.

$$\varphi_1(T) = \min \ \textstyle\sum_{e \in E} c_e y_e , \tag{1}$$

subject to

$$y\big(\delta(V_k)\big) = b(t_k), \quad \forall\, k = 1, \ldots, n-1, \tag{2}$$

$$y_e \geq 0, \ \forall\, e \in E, \tag{3}$$

where $b(t_k) = f_k - l\big(\delta(V_k)\big)$ for all $k = 1,...,n\text{-}1$ and we will denote the vector $(b(t_k); k = 1, \ldots, n-1)$ by b for short. Since each edge t_k is an edge of the cut $\delta(V_k)$, then the matrix A can be represented in the form (I, N) by permuting columns, where I is $(n - 1) \times (n - 1)$ identity matrix whose columns are indexed by edges of the tree T. The rows of the matrix A are 0 or 1 characteristic vectors of the cuts $\delta(V_k)$. A column h of N contains 1 in a row $\delta(V_k)$ if the edge t_k is an edge of the cycle $C(h)$ obtained by adding the edge $h \in E \setminus T$ to the tree T. Conversely, if t_1, ..., t_p are edges of the cycle $C(h)$, then h is just an edge of the cuts $\delta(V_1), \ldots, \delta(V_p)$. So, we have the following property;

Property 1 For any edge $h \in E \backslash T$, if t_1, ..., t_p are edges of a cycle $C(h)$, then the column h has just 1 in the rows corresponding to cuts $\delta(V_k), k = 1, \ldots, p$. This implies the linear dependence of the vector columns indexed by t_1,..., t_p and h.

For any $e \in E$, we write $e \in C(h)$ if e is an edge of the cycle $C(h)$. For edges $e, h \in E \setminus T$, if the cycle $C(e)$ contains only edges $t_k \in C(h)$, the cycle $C(e)$ is called a *subcycle* of $C(h)$, that is the edge e is a chord of $C(h)$. Let a^e denote the column e of the matrix N for each edge $e \in E \setminus T$. It follows that $a^h \geq a^e$ for any chord e of the cycle $C(h)$. We also assume that $b(t_k) \geq 0$ for each $V_k \in \mathcal{F}(T)$, otherwise the problem (1)-(3) is infeasible. Hence, the variables $y_{t_1}, \dots, y_{t_{n-1}}$ corresponding to the columns of I form basis for (1)-(3). The following propositions will be used to solve the problem (1)-(3).

Proposition 2 For an edge $h \in E \setminus T$, if

$$\sum_{t_k \in C(h)} c_{t_k} \leq c_h, \tag{4}$$

then the edge h can be deleted in the graph G.

Proof. The dual problem of (1)-(3) can be given as

$$\max \quad \sum_{t_k \in T} b(t_k) u(t_k)\,, \tag{5}$$

subject to

$$\sum_{\delta(V_k) \ni e} u(t_k) \leq c_e\,, \quad e \in E \tag{6}$$

in which conditions $u(t_k) \leq c_{t_k}$ are associated with the edges $t_k \in T$. By Property 1,

$$\sum_{\delta(V_k) \ni e} u(t_k) = \sum_{t_k \in C(e)} u(t_k) \leq \sum_{t_k \in C(h)} c_{t_k} \leq c_h,$$

the left side of the constraints (6) for $e = h$ is the sum of those for $e = t_k \in C(h)$. Therefore, the edge h can be deleted in G.

Proposition 3 Let $C(h)$ be any cycle and let e be *a* chord of $C(h)$, where $e, h \in E \setminus T$. If

$$\sum_{t_k \in C(h)} c_{t_k} - c_h \leq \sum_{t_k \in C(e)} c_{t_k} - c_e, \tag{7}$$

then the edge h can be deleted in the graph G.

Proof. The dual problem (5)-(6) can also be represented as

$$\max \quad \sum_{t_k \in T} b(t_k) z(t_k)\,,$$

subject to

$$\sum_{\delta(V_k) \ni e} z(t_k) \geq \sum_{t_k \in C(e)} c_{t_k} - c_e, \qquad e \in E \setminus T$$

$$z(t_k) \geq 0, \quad V_k \in \mathcal{F}(T)$$

by setting

$$z(t_k) = c_{t_k} - u(t_k).$$

Since the edge e is *a* chord of $C(h)$, $a^h \geq a^e$, that is,

$$\sum_{\delta(V_k) \ni h} z(t_k) \geq \sum_{\delta(V_k) \ni e} z(t_k).$$

It follows that

$$\sum_{\delta(V_k) \ni h} z(t_k) \geq \sum_{\delta(V_k) \ni e} z(t_k) \geq \sum_{t_k \in C(e)} c_{t_k} - c_e \geq \sum_{t_k \in C(h)} c_{t_k} - c_h$$

which is the last inequality (7). Therefore, we can delete the constraints (2) for $e = h$. In other words, the edge h can be deleted in the graph G. ■

By Proposition 3, if (7) holds for one of chords $e_1, \ldots, e_p \in E \setminus T$ of any cycle $C(h)$, then the constraint (2) for $e = h$ can be deleted. Hence, we have

$$\sum_{t_k \in C(h)} c_{t_k} - c_h > \sum_{t_k \in C(e_i)} c_{t_k} - c_{e_i} \tag{8}$$

for any cycle $C(h)$ ($h \in E \setminus T$) and its chords $e_1, \ldots, e_p \in E \setminus T$.

4. THE COMPLEXITY OF PROBLEM (1)-(3)

Using Propositions 2 and 3, we delete the edges $e \in E \setminus T$ in the graph G for which the conditions (4) and (7) hold. It is clear that this can be carried out in $O(m - n + 1)$ time. In the result, we have

$$\sum_{t_k \in C(h)} c_{t_k} > c_h, \quad \forall\, h \in E \setminus T, \tag{9}$$

and the condition (8) holds for any cycle $C(h)$ and its chords $e \in E \setminus T$. Although theoretically the number of pivot steps of the simplex algorithms is exponential, the following theorem states that the number of the classical simplex algorithm with Dantzig's pivot rule is bounded by a linear function of the number of edges for solving the linear program (1)-(3). It seems that to find a polynomial time simplex method and the closely related Hirsch conjecture proof is a hard problem in general.

Theorem 4 The simplex algorithm finds an optimal solution to (1)-(3) in at most $m - n + 1$ iterations.

Proof. Without loss of generality, we assume $b > 0$, since if $b(t_k) = 0$ for some t_k, then $x_{t_k}^0 = l_{t_k}$ in an optimal solution and hence the edges of the cut $\delta(V_k)$ can be deleted in G and the problem (1)-(3) can be solved independently for each connected component. Since $b > 0$, we can take

$$y_{t_k}^0 = b(t_k) \text{ for all } t_k \in T,$$
$$y_e^0 = 0 \text{ for all } e \in E \setminus T$$

as the initial basic feasible solution for the problem (1)-(3). Hence (1)-(3) has an optimal solution by the theory of linear programming. According to (9) the reduced cost is

$$\bar{c_e} = c_e - \sum_{t_k \in C(e)} c_{t_k} < 0,$$

for each $e \in E \setminus T$. Because y_e can be selected as the entering variable for any $e \in E \setminus T$. From (8) it follows that the classical simplex algorithm selects y_h for which

$$\bar{c_h} = \min\{\bar{c_e}; e \in E \setminus T\},$$

for entering to the basic y^0 and $y_e^0 = 0$ for any chord e of the cycle $C(h)$. Moreover $y_{t_q}^0 = 0$, since $y_{t_q} = 0$ is a leaving variable for some edge $t_q \in C(h)$, since columns for edges $t_q \in C(h)$ and h as vectors are linearly dependent by Property 1.

Let the simplex algorithm proceeds to a basic solution $y^1 = (y_e^1; e \in E)$ by choosing to bring y_g into basic $\bar{y}$ and removing the variable y_h from the basic $\bar{y}$ for first time. Then the following two cases are possible for positions of the edges h and g with respect to fundamental cuts.

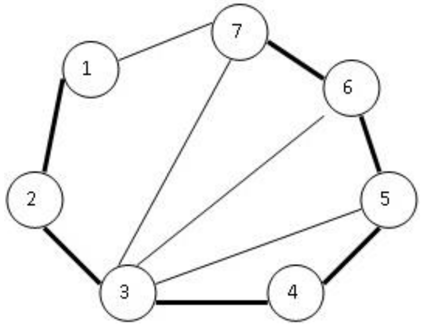

Figure 2. h = (1, 7), t_q = (6, 7), g = (3, 7)

Case 1: In some basis solutions generated before y^1, when y_h became a basic variable and y_{t_q} non-basic variable, there is a subcycle $C(e_0)$ of $C(H)$, such that $y_t > 0$ for edges t in $C(e_0) \cap T$ and $t_q \notin C(e)$. Since $y_{t_q} = 0$ the subcycle $C(e_0)$ as cycle $C(h)$, the edge e_0 is entering to some basic solution and some edge $t_1 \in C(e_0) \cap T$ is leaving. This can be repeated for some subcycle of $C(e_1)$ of $C(e_0)$ such that $t_1 \notin C(e_1)$. We assume that this process is carried out for edges e_i, where $i = 0,1,\ldots,p$ and $g = e_p$. By Proposition 2 and

$$\sum_{t_k \in C(e_i)} c_{t_k} - c_{e_i} > 0,$$

for edges $e = e_0, e_1,\ldots,e_{p-1}$, the variables y_{e_i} were in some basic solutions generated before y^1. Since y_h is leaving basic variable from the basic solution $\bar{y}$ and y_g is entering basic variable to $\bar{y}$, $C(g) = C(e_p)$ is the last of such subcycles that the following inequality

$$\sum_{e_i}\Big(\sum_{t_k \in C(e_i)} c_{t_k} - c_{e_i}\Big) > \sum_{t_k \in C(h)} c_{t_k} - c_h$$

holds.

Figures 2 displays a piece of T whose edges are bold lines, where y_t^1 for bold edges are in the basic solution y^1 except t_q = (6,7). When $y_h = y_{17}$ entered to the basic solution, y_{t_q} became a non-basic variable. In some solutions generated before y^1, for edges e and t on the subcycles $C(3,6)$ and $C(3,5)$ of the cycle $C(1,7)$, the variables y_e and y_t became a basic and a non-basic variable, respectively, in the following order;

- the y_{36} became a basic variable and t_{56} became non-basic variable on the subcycle $C(3,6)$ of $C(1,7)$;
- the y_{35} became a basic variable and t_{45} became non-basic variable on the subcycle $C(3,5)$ of $C(1,7)$;

After y_{36} and y_{35} became basic variables, $y_g = y_{37}$ is entering variable to the basic solution y^1 and $y_h = y_{17}$ is a leaving variable from y^1 with respect to the cycle with edges g = (3,7), h = (1,7), t = (1,2), t = (2, 3).

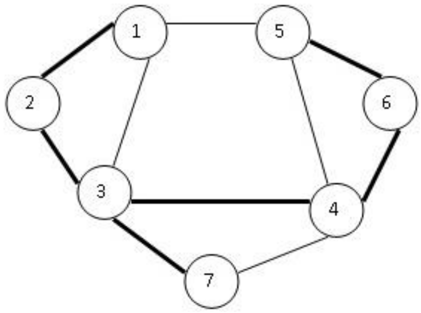

Figure 3. h = (1, 5), t_q = (3, 4), g = (4, 7)

Case 2: The edge t_q is on the cycles $C(h)$ and $C(g)$ that are not subcycles of $C(h)$. As in Case 1, in some basic solutions generated before y^1, for some edges t in $T \cap C(h)$, the variable y_t and y_e became a non-basic and a basic variable, respectively, for some subcycles $C(e)$ of $C(h)$. Differently from Case 1, there is a cycle C in G such that the edges h, g and edges e of these subcycles are on C. Figure 3 indicates that the cycle C with edges h, (1, 3), (3, 7), g, (4, 5) for which y_e^1 (e = (1,3), (4,5), h) and $y_t^1 = y_{37}^1$ (t = (3,7) $\in T$) are basic variables.

Now, from that y_g is an entering basic variable to $\bar{y}$ implies

$$\sum_{\delta(V_k)\ni g} u(t_k) > c_g,$$

for the simplex multipliers $\{u(t_k)\}$ as trial solution to the dual of (1)-(3) which means that the dual constraint associated with the variable y_g is violated. To enter y_g to the basic solution, the simplex algorithm defines new multipliers $\{u(t_k)\}$ satisfying the following equation

$$\sum_{\delta(V_k)\ni g} u(t_k) = c_g.$$

This means that in both cases,

$$\sum\{u(t_k);\ \text{for edges } t_k \text{ on } C(h) \text{ and } C(g)\}$$

is decreasing after the simplex algorithm defined the basic y^1. Therefore, each y_h can be a basic variable only once and hence the simplex algorithm finds an optimal solution to the problem (1)-(3) in at most $m - n + 1$ iterations. ■

Consider the problem (1)-(3) on the graph G in Figure 2. Let T be a spanning tree with bold edges (1,4), (2,5), (3,6), (4,5), (5,6) and then

$$\mathcal{F}(T) = \{V_{14} = \{1\}, V_{25} = \{2\}, V_{36} = \{3\}, V_{45} = \{1,4\}, V_{56} = \{3,6\}\}.$$

The edges crossing with the dashed lines represent edges of corresponding fundamental cuts whose numbers are shown in rectangles at the end of the dashed lines. Let $b(14) = 5$, $b(25) = 6$, $b(36) = 6$, $b(45) = 7$ and $b(56) = 8$ for the fundamental cut sets in $\mathcal{F}(T)$. The number next to each edge e indicates its cost c_e. Since conditions (4) and (8) do not hold at any edge of the cycles $C(12)$, $C(13)$, $C(23)$, $C(46)$, any edge cannot be deleted in G. In this example, $y_{14} = 5$, $y_{25} = 6$, $y_{36} = 6$, $y_{45}=7$ and $y_{56} = 8$ are initial basic variables corresponding to the edges in T. In the first iteration, the variables y_{13} and y_{14} are entering and leaving basic variables, respectively. In the next three iterations, y_{23} and y_{36}, y_{46} and y_{56}, y_{12} and y_{25} become basic and non-basic variables, respectively. The optimal solution is $y_{13}^* = 2.5, y_{32}^* = 3.5, y_{12}^* = 2.5$, $y_{46}^* = 2$ and $y_e^* = 0$ for the other edges.

As a conclusion of Section 2, let $P(T)$ be a polytope defined by (2) and (3) and let its vertices v_T and v_{op} be an initial and an optimal basic solutions, respectively. By Theorem 4, it follows that the minimum distance between the vertices v_T and v_{op} is not greater than $m - n + 1$ in $P(T)$. Theorem 4 states that the Hirsch conjecture [3] holds for the facet containing vertices v_T , v_{op}, if $m \leq 3(n - 1)$ in a graph G.

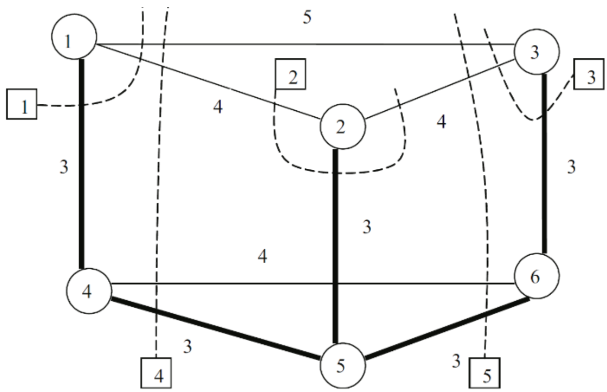

Figure 4. *A* graph *G*, a spanning tree *T* and five fundamental cuts.

It is not difficult to give some examples to show that (1)-(3) has no integer-valued optimal solutions for $\mathcal{F}(T)$ with respect to some spanning tree *T*. When *T* is a Hamiltonian path, all 1's are consecutive in each column of the matrix *A* (see Proposition 6) from which it follows that the matrix *A* is unimodular [10]. This means that the problem (1)-(3) has an integer valued solution for an integer vector *b*, when *T* is any Hamiltonian path. The problem (1)-(3) with the vector $b = 1$ is the LP relaxation of the $\mathcal{F}(T)$Constrained Forest Problem (CFP) [4] when the proper function $f(S) = 1$ ($f: 2^V \to \{0,1\}$ for the fundamental cut sets $S \in \mathcal{F}(T)$). Hence, if the tree *T* is any Hamiltonian path, then the LP relaxation of CFP has an integer-valued optimal solution.

5. CHARACTERIZATION OF THE *FC* MATRICES

Let $A = (I,N)$ be a given binary matrix. Is there an undirected simple graph $G = (V,E)$ for which the columns of *A* can be indexed by edges in *E*, so that the rows of *A* are 0, 1 characteristics vectors of fundamental cuts corresponding to the edges of some spanning tree in *G*? In other words, whether the matrix *A* represents a graphic matroid with respect to some its base. The following lemma allows to construct a graph for some particular cases of the matrix $A = (I,N)$.

Lemma 5 If a binary matrix $A = (I,N)$ has n - 1 rows and $m \leq (n - 1)n/2$ columns, and each column of *N* contains two 1's and no identical columns in *N*, then it can be represented by graph a $G = (V,E)$ with *n* nodes and *m* edges with respect to (n - 1) star-spanning tree *T* in *G*.

Proof. Let *T* be a star with n - 1 edges and *n* nodes. First we index rows of *A* and columns of *I* by edges of the tree *T*. Then we construct the graph $G = (V,E)$ by adding edges to *T*: for any column *j* of *N* containing 1's in rows *t* and *h*, we add an edge *j* to the tree *T* so that it connects leaves of edges *t* and *h*.

Since there is no any identical columns in *N*, the graph *G* has no parallel edges. $m \leq (n - 1)n/2$ can be written as $m - (n - 1) \leq (n - 1)(n - 2)/2$ which means that the columns of *N* can be indexed by no tree edges of a simple undirected graph *G*. Suppose that a row *t* in *A* has 1's in columns j_1, j_2,..., j_p. In *G*, since edges j_1, j_2,..., j_p connect the leaf of the edge *t* to the leaves of the other edges in T, then the cut corresponding to the edge t contains all those edges. So, the row t is the characteristic vectors of the fundamental cut corresponding to the edge t in T and the proof is complete. ■

Proposition 6 *Let* $T = (V,E(T))$ be *a* Hamiltonian path connecting some two nodes $v,w \in V$ in a graph $G = (V,E)$. The *FC* matrix of the graph G with respect to the tree T is totally unimodular.

Proof. Without loss of generality, we may assume that the Hamiltonian path $T = ((v = 1, 2), (2, 3), \ldots, (n\text{-}1, n = w))$ connects the nodes v and w. Consider an *FC* matrix whose ith row is indexed by the edge $(i, i + 1)$ in T for $i = 1, \ldots, n - 1$ and the columns are indexed as follows: the first $n - 1$ columns are indexed by the edges in T and then the other $m - n + 1$ columns are indexed by the edges $(1, i)$, for $3 \leq i \leq n$, the edges $(2, i)$, for $4 \leq i \leq$ n and so on.

Thus, the row of the *FC* matrix corresponding to the cut determined by deleting the edge (1, 2) contains 1's in the columns (1, 2) and $(1, i)$ for $3 \leq i \leq n$. The row (2, 3) corresponding to the cut determined by deleting the edge (2, 3) contains 1's in all columns $(1, i)$ for $3 \leq i \leq n$, since nodes 1 and 2 are on the same side of the cut. Similarly, the row (3, 4) corresponding to the cut determined by deleting the edge (3, 4) contains 1's in all columns $(1, i)$ for $4 \leq i \leq n$ and so on. That is, the columns $(1, i)$ $(3 \leq i \leq n)$ of the *FC* matrix contain the 1's consecutively. Since the first $(n - 1)$ columns are unit vectors, this *FC* matrix is in the form $A = (I,N)$. By Heller and Hoffman, the *FC* is a totally unimodular matrix [10]. ■

Let $A = (I,N)$ be an *FC* matrix whose columns are indexed by 1,…,m and rows are indexed by $1,\ldots, p$, where $p = n - 1$. Suppose that a column k in N contains 1's in rows $k_1, \ldots, k_q$ in N. In other words, the column k *induces* a q x $(m - p)$ submatrix $A(k)$ of N with the rows $k_1, \ldots, k_q$. If there exist a reordering of the rows of $A(k)$ such that the permuted submatrix has no column containing 0's between 1's, then we call it an *HP* (Hamiltonian path) matrix.

Since A is an *FC* matrix, there is a graph $G = (V,E)$ with $|E| = m$ and $|V| = n$ such that the rows of A are 0, 1 characteristics vectors of fundamental cuts corresponding to some spanning tree of G. Now, let $A(k)$ and $A(h)$ be *HP* matrices induced by the columns k and h of N and let $(kh)_1, \ldots, (kh)_t$ be common rows of $A(k)$ and $A(h)$. This means that the edges corresponding to the rows $(kh)_1, \ldots, (kh)_t$ are on the cycles $C(k)$ and $C(h)$ created by adding the edges e_k and e_h to the spanning tree. Suppose that these rows were reordered as $(kh)_1, \ldots, (kh)_t$ to transform $A(k)$ and $A(h)$ to *HP* matrices. If the common rows can be reordered as $(kh)_1, \ldots, (kh)_t$ or $(kh)_t, \ldots, (kh)_1$ to transform $A(h)$ and $A(k)$ to *HP* matrices, we call k and h tree-common columns.

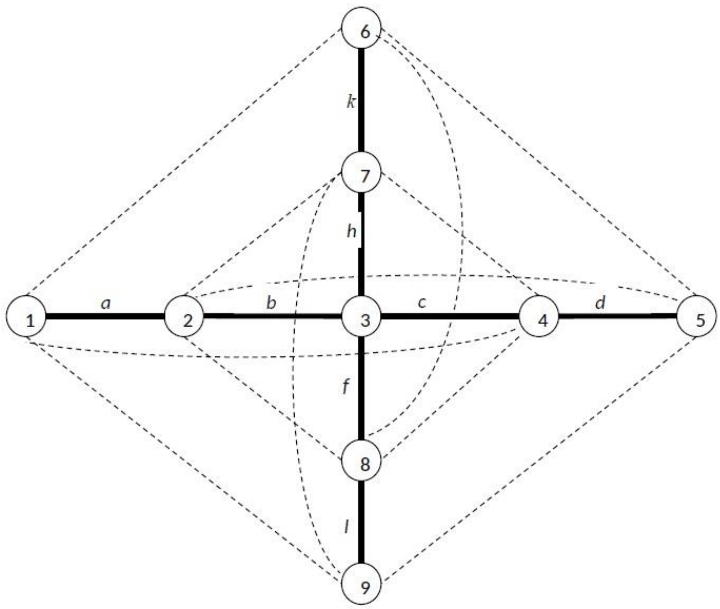

Figure 5. *A* graph *G* (bold and dashed lines) and a spanning tree *T* (bold lines).

In order to explain the key moments of the tree-common columns, first consider the graph of Figure 3, where the bold lines represent the edges of the spanning tree. To make a distinction between the edges of T and the others, we denote the edges of *T* by *a* = (1, 2), *b* = (2, 3), *c* = (3, 4), *d* = (4, 5), *k* = (6, 7), *h* = (3, 7), *f* = (3, 8), *e* = (8, 9) and denote the other edges (dashed lines) with end nodes *i* and *j* by e_{ij} in *G*. The matrix *N* is as follows:

$$N=\begin{array}{cc} & \begin{array}{cccccccccccc} e_{14} & e_{16} & e_{19} & e_{25} & e_{27} & e_{28} & e_{47} & e_{48} & e_{56} & e_{59} & e_{68} & e_{79} \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \\ k \\ h \\ f \\ l \end{array} & \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

If the rows of the submatrix induced by the column e_{16} are reordered as *abhk*, then it becomes an *HP* matrix. The submatrix induced by the column e_{68} is an *HP* matrix with the reordering *khf* of its rows. The column e_{56} induces a submatrix which is an *HP* matrix after reordering its rows as dchk. Thus, the columns e_{16} and e_{56} are tree-common columns.

Theorem 7 An FC matrix satisfies the following conditions:

1. Its columns induce an *HP* matrix.
2. If submatrices induced by any pair of distinct columns have common rows, then these columns are tree-common.

Proof. Let $A = (I,N)$ be an *FC* matrix. There exist unknown undirected graph $G = (V,E)$: the columns of A can be indexed by edges of G and the rows are 0, 1 characteristics vectors of the fundamental cuts corresponding to the edges of some spanning tree T in G. Let k be the first column in N with the maximum number of ones. Then the edge k cannot be a chord of any cycle and the nodes of the cycle $C(k)$ induce the subgraph G_k in which the edges $t \in C(k)$ are on a Hamiltonian path P_k. Because the submatrix $A(k)$ induced by the column k is an *HP* matrix by Proposition 6. Now, let h be the second column in N (if exists) with the maximum number of ones such that no all rows of the submatrix $A(h)$ are the rows of $A(k)$. Similarly, it can be shown that $A(h)$ is an *HP* matrix, too. Therefore, by iterating this, it can be shown that the selected columns induce *HP* submatrices. Since the edges for any non-selected columns are chords of cycles created by edges for the selected columns, the submatrices induced by the first type columns are *HP*.

Suppose that submatrices $A(h)$ and $A(k)$ have common rows $(kh)_1, \ldots, (kh)_q \in T$. This means that these edges are on a subpath P_{kh} in the tree T, since P_k and P_h are paths in T. The subpath P_{kh} is in T, hence, k and h are tree-common columns that complete the proof. ■

Note that the converse of Theorem 7 is not true, for example, the well known dual Fano matrix [9] satisfies the conditions 1 and 2 of Theorem 7, however, it is not an *FC* matrix. The well known Seymour's matrices R_{10} and R'_{10} [12] are not an *FC* matrix, since they contain rows which do not induce an *HP* matrix. Any *FC* matrix can be regarded as the representation of a graphic matroid as binary matroid. An *FC* matrix is not generally a totally unimodular matrix. Therefore, this representation is not always regular for a graphic matroid.

6. CONCLUSION

The problem of graph visualization is important in computer science [6]. The proof of Theorem 7 can be used to design an algorithm for drawing graphs represented by *FC* matrices.

We observed that when the matrix A is an *FC*-matrix of the Hamiltonian-spanning tree,

$$P = \{x \in R^m; Ax = f, x \geq l \geq 0\}$$

is an integral polytope [10] by Proposition 6. In terms of the linear complexity of the simplex algorithm for the problem

$$\min\{cx; x \in P\}, \tag{10}$$

it appears that one of the most promising approaches for network design problems [8], [11] is to find an integer solution of (10) in each iteration of the branch and cut algorithm. Therefore, characterization of all spanning trees of the graph G, for which P is an integral polytope, can be used to get new polyhedral results for reducing subproblems encountered in solution of the above-mentioned network design problems.

Another suggested direction for future investigation is to strength Theorem 4, i.e., to prove that the simplex algorithm finds a solution to the problem (10) in almost linear time for a binary matroid matrix A and a given non-negative vectors c and f.

REFERENCES

[1] C. Berge: Balanced matrices and property (G), Mathematical Programming Studies 12 (1980), 163-175.

[2] K.H. Borgwardt: The simplex method: A probabilistic Analysis, Algorithms and Combinatorics 1, Springer-Verlag, (1987).

[3] F. Fritzsche and F.B. Holt: More polytope meeting the conjectured Hirsch bound, Discrete Math., 205 (1999), 77-84.

[4] M.X. Goemans and D.P. Williamson: A General Approximation Technique for Constrained Forest Problems, SIAM Journal on Computing, 24:2 (1995), 296-317.

[5] T.C. Hu: Optimum communication spanning trees, SIAM Journal on Computing, 3:3 (1974), 188-195.

[6] M. Kaufmann and D. Wagner: Drawing Graphs: Methods and Models, Springer, (2001).

[7] V. Klee and G.J. Minty: How good is the simplex algorithm?, Inequalities-III, Academic Press, New York (1972), 159-175.

[8] K.V. Marintseva, F.A. Sharifov, G.N. Yun: A problem of airport capacity definition, Aeronautic, 5 (2013), 1-13.

[9] J. Oxley: Matroid theory, Oxford University Press, 2nd Ed., New York, (2011).

[10] Schrijver A.: Combinatorial optimization: Polyhedra and efficiency in Algorithms and Combinatorics, Spinger-Verlag, Berlin, Heidelberg, 24, (2003).

[11] F.A. Sharifov and L. Hulianytskyi: Models and Complexity of Problems of Design and Reconstruction of Telecommunication and Transport Systems, Cybernetics and Systems Analysis, 50:5 (2014), 693-700.

[12] P.D. Seymour: Decomposition of Regular Matroids, Journal of Combinatorial Theory, Series B, 28:3 (1980), 305-359.

[13] M.J. Todd: Polynomial expected behaviour of a pivoting algorithm for linear complementarity and linear programming problems, Technical Report No. 595, School of Operations Research and Industrial Engineering, Cornell University, (Ithaca, NY, November 1983), Mathematical Programming, 35 (1986), 173-192.

[14] K. Truemper: Matroid Decomposition, Academic Press, (1992).