Contemporary Mathematics
http://ojs.wiserpub.com/index.php/CM/

Research Article

Optimum Flows in Directed Planar Dynamic Networks. The Dynamic Approch

Camelia Schiopu[1*], **Eleonor Ciurea**[2]

[1]Department of Management and Economic Informatics, Transilvania University of Braşov, Braşov, Romania
[2]Department of Mathematics and Computer Science, Transilvania University of Braşov, Braşov, Romania
E-mail: camelia.s@unitbv.ro

Received: 30 July 2024; **Revised:** 14 September 2024; **Accepted:** 9 October 2024

Abstract: Optimum flows in directed planar dynamic networks are essential for several reasons, impacting a variety of fields. These networks present unique challenges that require advanced optimization techniques to ensure efficient and reliable performance. In this paper we consider a time-varying directed planar network without parallel arcs and loops, where a flow must take a certain time to traverse an arc. The problem is to find an optimal (minimal or maximal) solution to send the optimum (minimum or maximum) flow from the source node to the sink node, within a given time T.

***Keywords*:** minimum flow, maximum flow, static network, dynamic network

MSC: 35A01, 65L10, 65L12, 65L20, 65L70

1. Introduction

The networks are among the most basic and powerful objects to model relations and processes across a diverse array of disciplines, including sociology, economy, physics, chemistry, computer science, biology, and engineering, as well as many other fields. It offers a framework for modeling complex systems and interactions, making them indispensable in both theoretical research and practical applications. This paper is a synthesis of papers [1–3].

The literature on network flow problems is extensive, and over the past 50 years researchers have made continuous improvement to algorithms for solving several classes of problems. In recent years, there has been an increase in work focused on the algorithmic aspects of network flow problems. These contributions have highlighted how the use of clever data structures and careful analysis can improve the theoretical performance of network algorithms.

The planar networks are an important family of networks that has been extensively studied and has many practical applications in different fields. For instance, they play a crucial role in transportation networks, where they help optimize routes and reduce congestion. In geographical routing for communication networks, planar networks facilitate efficient data transmission. In the field of computer vision, they aid in image processing and object recognition. These examples underscore the wide-ranging utility of planar networks across different domains.

Flow variation over time is an important feature in network flow problems. This temporal dimension is particularly relevant in applications such as road and air traffic control, where real-time adjustments are necessary to manage flow effectively. Similarly, in production systems, the timing of material and product flows can significantly impact overall

DOI: https://doi.org/10.37256/cm.6120255401

efficiency. Communication networks, including the Internet, rely on dynamic flow management to ensure smooth data transmission. Financial flows also exhibit temporal variation, with the timing of transactions affecting market dynamics. In these contexts, it is essential to consider not only the amount of flow to be transmitted but also the time required for its transmission through the network's arcs.

The study of optimum flows in directed planar dynamic networks has garnered significant attention, given its critical applications in various fields. Foundational theories such as Ford and Fulkerson's Max-Flow Min-Cut Theorem laid the groundwork for understanding network flow optimization [4]. The Maximum Flow-Minimum Cut Theorem is a fundamental result in network flow theory. It establishes a relationship between two key concepts in a flow network: the maximum flow that can be sent from a source node to a sink node, and the minimum cut, which represents the smallest capacity that, if removed, would prevent any flow from reaching the sink. The Maximum Flow-Minimum Cut Theorem provides a powerful tool for optimizing the flow of resources through networks, as in the example of transportation systems (e.g., road networks, supply chains). In this case the theorem helps identify bottlenecks that limit the flow of goods. By identifying the minimum cut (the set of critical links or paths), efforts can be focused on improving infrastructure or rerouting traffic to maximize the overall flow of goods. Planar networks benefit from unique properties highlighted by Lipton and Tarjan's Separator Theorem, which facilitates efficient algorithmic solutions [5]. Dynamic network flow models, introduced by Cooke and Halsey, consider the time-varying nature of flows, complicating traditional optimization problems [6].

This paper aims to explore these dimensions of network flow problems, with a particular focus on planar networks. This paper is structured as follows. Section 2 and Section 3 are presented notations, definitions, and results from general static and dynamic networks which are necessary for our purposes. These sections lay the groundwork for understanding the more complex dynamic scenarios discussed later. Section 4 presents the problem of minimum and maximum flow, respectively in planar static networks. Section 5 delves into the problem of optimum flow in directed planar dynamic networks, presenting a dynamic approach based on the findings from previous studies. This section synthesizes the insights from papers [2, 3] offering a comprehensive overview of current methodologies. Finally, Section 6 illustrates the theoretical concepts with a practical example, demonstrating the application of these principles in a real-world context. Through this structured approach, the paper aims to contribute to research on network flow problems and their solutions.

2. Optimum flows in general static networks

The notions and the results presented in this section are taken over from works [7–9].

Let $G = (N, A, l, u, 1, n)$ be a general static network with the node set $N = \{1, \ldots, i, \ldots, j, \ldots, n\}$, the arc set $A = \{a_1, \ldots, a_k, \ldots, a_m\}$, $a_k = (i, j)$, the lower bound function $l : A \to \mathbb{N}$, the upper bound (capacity) function $u : A \to \mathbb{N}$ where $\mathbb{N}$ is the natural set, 1 the source node and n the sink node.

For a given pair of not necessarily disjoint subset $X \subset N, Y \subset N$ we use the notation $(X, Y) = \{(i, j)|(i, j) \in A, i \in X, j \in Y\}$ and for a function $f: A \to \mathbb{N}$ we use the notation $f(X, Y) = \sum_{(X,Y)} f(i, j)$.

A flow in network G is a function $f: A \to \mathbb{N}$ satisfying the next conditions:

$$f(i, N) - f(N, i) = \begin{cases} v, & \text{if } i = 1 \\ 0, & \text{if } i \neq 0, n \\ -v, & \text{if } i = n \end{cases} \tag{1}$$

$$l(i, j) \leq f(i, j) \leq u(i, j), \quad \text{for all } (i, j) \in A \tag{2}$$

with the value of the flow $v \geq 0$ and $f(i, j) = 0$ for the pairs $(i, j) \notin A$.

The optimum (minimum or maximum) flow problem consists in determining a flow f for which v is optimized (minimized or maximized).

A cut is a partition of N set into two proper subsets S and $T = N - S$. We represent this cut using the notation $[S, T]$. An arc $(i, j) \in A$ with $i \in S$ and $j \in T$ is a forward arc of the cut and an arc (j, i) with $i \in S$ and $j \in T$ is a backward arc of the cut. Let (S, T) denote the set of forward arcs, and let (T, S) denote the set of backward arcs in the cut. We have $[S, T] = (S, T) \cup (T, S)$. We refer to a cut as a $1 - n$ cut if $1 \in S$ and $n \in T$.

If a function f verifies (1) then f is a flow and if it also verifies (2) then f is a feasible flow. Whereas the optimum flow problem with zero lower bounds always has a feasible solution (since the zero flow is feasible), the problem with nonnegative lower bounds could be infeasible. Any optimum flow algorithm for problems with nonnegative lower bounds has two phases:

1. to determine a feasible flow if one exists;
2. converts a feasible flow into an optimum flow.

We transform the maximum flow problem into a circulation problem by adding an arc $(n, 1)$ with $l(n, 1) = 0, u(n, 1) = \infty$. This arc carries the flow sent from node 1 to node n back to node 1. Clearly, the maximum flow problem admits a feasible flow if and only if the circulation problem admits a feasible flow.

The feasible circulation problem is to identify a flow f satisfying the following constraints:

$$f(i, N) - f(N, i) = 0 \quad \text{for all} \quad i \in N \tag{3}$$

$$l(i, j) \leq f(i, j) \leq u(i, j), \quad \text{for all} \quad (i, j) \in A \tag{4}$$

Theorem 1 (Circulation Feasibility Conditions) A circulation problem with nonnegative lower bounds on arc flows is feasible if and only if for every set X of nodes $l(\bar{X}, X) \leq u(X, \bar{X}), \bar{X} = N - X$.

For the minimum flow problem, we define the capacity $k[S, T]$ of the $1 - n$ cut $[S, T]$ as:

$$k[S, T] = l(S, T) - u(T, S) \tag{5}$$

We refer to a $1 - n$ cut $[S, T]$ which has the maximum capacity among all $1 - n$ cuts as a maximum cut.

Theorem 2 (Min-Flow Max-Cut Theorem) If there exists a feasible flow in the network $G = (N, A, l, u, 1, n)$, the value of the minimum flow from a source node 1 to a sink node n is equal to the capacity of the maximum $1 - n$ cut.

For the maximum flow problem, we define the capacity $K[S, T]$ of the $1 - n$ cut $[S, T]$ as:

$$K[S, T] = u(S, T) - l(T, S) \tag{6}$$

A $1 - n$ cut $[S, T]$ which has the minimum capacity among all $1 - n$ cuts is a minimum cut.

Theorem 3 (Max-Flow Min-Cut Theorem) If there exists a feasible flow in the network $G = (N, A, l, u, 1, n)$, the value of the maximum flow from a source node 1 to a sink node n is equal to the capacity of the minimum $1 - n$ cut.

The concept of the residual network plays a central role in the development of all the optimum flow algorithms. The residual capacity of an arc (i, j) for a maximum flow problem is

$$r(i, j) = u(i, j) - f(i, j) + f(j, i) - l(j, i) \tag{7}$$

and for minimum flow problem is

$$r(i,\, j) = u(j,\, i) - f(j,\, i) + f(i,\, j) - l(i,\, j) \tag{8}$$

We refer to the network $\tilde{G} = (N,\, \tilde{A},\, r,\, 1,\, n)$ consisting of the arcs with positive residual capacities of the residual network (with respect to the flow f).

The optimum flow algorithms work with only residual capacities [9]. These algorithms end with optimal residual capacities. From these residual capacities, we can construct optimum flow. For maximum flow, we have:

$$f(i,\, j) = l(i,\, j) + \max\{0,\, u(i,\, j) - r(i,\, j) - l(i,\, j)\} \tag{9}$$

and for the minimum flow, we have:

$$f(i,\, j) = l(i,\, j) + \max\{0,\, r(i,\, j) - u(j,\, i) + l(j,\, i)\} \tag{10}$$

Clearly, the results present in this section are also valid for particular networks (planar and bipartite).

3. Optimum flows in general dynamic networks

The notions and the results presented in this section are taken over from works [7, 9–14].

Dynamic network models arise in many problem settings, including production-distribution systems, economic planning, energy systems, traffic systems, and building evacuation systems.

Let $\mathbb{N}$ be the natural number set and let $H = \{0,\, 1,\, \ldots,\, T\}$ be the set of periods, where $T \in \mathbb{N}$ is a finite time horizon. Let $D = (N,\, A,\, h,\, e,\, q,\, H)$ be a general dynamic network with the node set $N = \{1,\, \ldots,\, i,\, \ldots,\, j,\, \ldots,\, n\}$, the arc set $A = \{a_1,\, \ldots,\, a_k,\, \ldots,\, a_m\}$, $a_k = (i,\, j)$, the transit time function $h : A \times H \to \mathbb{N}$, the time lower bound function $e : A \times H \to \mathbb{N}$, the time upper bound function $q : A \times H \to \mathbb{N}$, $e(i,\, j;\, t) \leq q(i,\, j;\, t)$, for all $(i,\, j) \in A$ and for all $t \in H$.

The optimum (minimum or maximum) dynamic flow problem for T time periods is to determine a dynamic flow function $g : A \times H \to \mathbb{N}$, which should satisfy the following conditions in dynamic network $D = (N,\, A,\, h,\, e,\, q,\, H)$:

$$\sum_{t=0}^{T} \left(\sum_{j} g(1,\, j;\, t) - \sum_{k} \sum_{\tau} g(k,\, 1;\, \tau) \right) = w \tag{11}$$

$$\sum_{j} g(i,\, j;\, t) - \sum_{k} \sum_{\tau} g(k,\, i;\, \tau) = 0, \quad i \neq 1,\, n, \quad t \in H \tag{12}$$

$$\sum_{t=0}^{T} \left(\sum_{j} g(n,\, j;\, t) - \sum_{k} \sum_{\tau} g(k,\, n;\, \tau) \right) = -w \tag{13}$$

$$e(i,\ j;\ t) \le g(i,\ j;\ t) \le q(i,\ j;\ t),\quad (i,\ j) \in A,\quad t \in H \tag{14}$$

$$opt \quad w, \tag{15}$$

where $\tau = t - h(k,\ i;\ \tau)$, $w = \sum_{t=0}^{T} v(t)$, $v(t)$ is the flow value at time t, $g(i,\ j;\ t) = 0$ for all $t \in \{T - h(i,\ j;\ t) + 1,\ \ldots,\ T\}$ and opt is min or max.

In the most general dynamic model, the parameter $h(i) = 1$ is waiting time at node i and the parameters $e(i;\ t)$, $q(i;\ t)$ are lower bound and upper bound for flow $g(i;\ t)$ that can wait at node i from time t or $t+1$. This most general dynamic model is not discussed in this paper.

Obviously, the problem of finding an optimum flow in a dynamic network $D = (N,\ A,\ h,\ e,\ q,\ H)$ is more complex than the problem of finding an optimum flow in a static network $G = (N,\ A,\ l,\ u)$. Happily, this complication can be resolved, through static approach, by rephrasing the problem in dynamic network D into a problem in static network $R_0 = (V_0,\ E_0,\ l_0,\ u_0)$ with multiple source nodes and multiple sink nodes or in static network $R_1 = (V_1,\ E_1,\ l_1,\ u_1)$ with a single source node and a single sink node. The network R_1 can be obtained by two methods:

1. using static shortest path [7];
2. using dynamic shortest path [10].

We present only the second method [3].

Let $d(1,\ i;\ t)$ be the length of the dynamic shortest path at time t from the source node 1 to the node i, and let $d(i,\ n;\ t)$ be the length of the dynamic shortest path at time t from the node i to the sink node n, with respect to h in the dynamic network D. Let us consider $H_i = \{t | t \in H,\ d(1,\ i;\ t) \le t \le T - d(i,\ n;\ t)\}$, $i \in N$, and $H_{i,\ j} = \{t | t \in H,\ d(1,\ i;\ t) \le t \le T - h(i,\ j;\ t) - d(j,\ n;\ \theta)\}$, $(i,\ j) \in A$. The multiple source, multiple sinks static reduced expanded network $R_0 = (V_0,\ E_0,\ l_0,\ u_0)$ has $V_0 = \{i_t | i \in N,\ t \in H_i\}$, $E_0 = \{(i_t,\ j_\theta) | (i,\ j) \in A,\ t \in H_{i,\ j}\}$, $l_0(i_t,\ j_\theta) = e(i,\ j;\ t)$, $u_0(i_t,\ j_\theta) = q(i,\ j;\ t)$, $(i_t,\ j_\theta) \in E_0$. The static reduced expanded network $R_1 = (V_1,\ E_1,\ l_1,\ u_1)$ is constructed from the network R_0 as follows: $V_1 = V_0 \cup \{0,\ n+1\}$, $E_1 = E_0 \cup \{0,\ 1_t | 1_t \in V_0\} \cup \{(n_t,\ n+1) | n_t \in V_0\}$, $l(0,\ 1_t) = l(n_t,\ n+1) = 0$, $u(0,\ 1_t) = u(n_t,\ n+1) = \infty$, $1_t,\ n_t \in V_0$ and $l(i_t,\ j_\theta) = l_0(i_t,\ j_\theta)$, $u(i_t,\ j_\theta) = u_0(i_t,\ j_\theta)$, $(i_t,\ j_\theta) \in E_0$.

The optimum flow problem for T time periods in the dynamic network D as stated in the conditions (11)-(15) is equivalent with the maximum flow problem in the static reduced expanded network R_1, as follows:

$$\sum_{j_\theta} f_1(i_t,\ j_\theta) - \sum_{k_\tau} f_1(k_\tau,\ i_t) = \begin{cases} v_1, & \text{if } i_t = 0 \\ 0, & \text{if } i_t \ne 0,\ n+1 \\ -v_1, & \text{if } i_t = n+1 \end{cases} \tag{16}$$

$$l_1(i_t,\ j_\theta) \le f_1(i_t,\ j_\theta) \le u_1(i_t,\ j_\theta),\quad (i_t,\ j_\theta) \in E_1 \tag{17}$$

$$opt \quad v_1, \tag{18}$$

where by convention $i_t = 0$ for $t = -1$ and $i_t = n+1$ for $t = T+1$.

If T is very large, then the static reduced expanded network R_1 becomes very large and the number of calculations required to find an optimum flow in the network R_1 becomes prohibitively large. Happily, Ford and Fulkerson [9] have

devised an algorithm that generates a correct flow in a dynamic network D. This algorithm works only when $e = 0$ and h, q are constant over time. If h, e, q are constant over time, then a dynamic network D is said to be stationary.

The algorithm for maximum dynamic flow in stationary dynamic network $D = (N, A, h, 0, q)$ is presented below.

Algorithm 1 Algorithm for maximum dynamic flow in a stationary dynamic network.

1: MDFSDN
2: BEGIN
3: AMVMCSF(G, $\overset{*}{f}$);
4: ADSFEF($\overset{*}{f}$, $r(P_1), \ldots, r(P_k)$);
5: ARPF($r(P_1), \ldots, r(P_k)$);
6: END.

The procedure AMVMCSF performs the algorithm for maximum value and minimum cost flow $\overset{*}{f}$ in static network $G = (N, A, c, u)$, where $c(i, j) = h(i, j)$, $u(i, j) = q(i, j)$, $(i, j) \in A$. For statements, we suppose that Klein's algorithm variant is used (minimum mean cycle canceling algorithm, see [7]). This algorithm has the complexity $O(n^2m^3 \log n)$. The procedure ADSFEF performs the algorithm for decomposition of static flow $\overset{*}{f}$ in elementary flows (path flows) with $r(P_s)$ the flow along of path P_s, $s = 1, \ldots, k$ from source node 1 to sink node n. This algorithm has the complexity $O(m^2)$. We remark that $c(P_s) \leq T$, $s = 1, \ldots, k$ is necessary. The procedure ARPF performs the algorithm to repeat each path flow, starting out from source node 1 at time periods 0 and repeating it after each time period as long as there is enough time left in the horizon for the flow along the path to arrive at the sink node n. This algorithm has complexity $O(nT)$. Hence, the algorithm MDFSDN has complexity $O(\max\{n^2m^3 \log n, nT\})$. The dynamic flow obtained with the algorithm MDFSDN is called temporally repeated flow and has the value:

$$\overset{*}{v}_1 = (T+1)\overset{*}{v} - \sum_A h(i, j)\overset{*}{f}(i, j) \tag{19}$$

where $\overset{*}{v}$ is value of maximum flow and minimum cost static flow $\overset{*}{f}$.

In stationary case the dynamic distances $d(1, i; t)$, $d(i, n; t)$ become static distances $d(1, i)$, $d(i, n)$.

We remark that the papers [11, 12] treat two particular cases of minimum flows in general dynamic networks.

4. Optimum flows in planar static networks

The notions and the results presented in this section are taken over from works [1, 7, 9, 15–19].

In this section we consider that the static network $G = (A, N, l, u, 1, n)$ is planar.

A digraph $G = (N, A)$ is said to be planar if we can draw it in a two dimensional plane so that no two arcs intersect each other. For more explanations see the works [1, 9, 15, 18]. Researchers have developed very efficient algorithms (in fact, linear time algorithms) for testing the planarity of a digraph. A face of G is a region of the plane bounded by arcs that satisfies the condition that any two points in the region can be connected by a continuous curve that meets no nodes and arcs. The boundary of a face x is the set of all arcs that enclose it. The face x and y are said to be adjacent if their boundaries contain a common arc. The planar digraph G has an unbounded face. We recall two well known properties of planar digraphs.

Theorem 4 (Euler's Formula) If a connected planar digraph has n nodes, m arcs, and φ faces, then $\varphi = m - n + 2$.

Theorem 5 If a connected planar digraph has n nodes, and m arcs, then $m < 3n$.

Theorem 5 shows that every planar digraph is very sparse i.e., $m = O(n)$. This result, by itself, improves the running times for most network flow algorithms.

Our discussion in this paper applies to a special class of planar digraphs known as $(1, n)$ planar digraphs. This class has the property that the source node 1 and the sink node n lie on the boundary of unbounded face.

We make the following assumption: if $(i, j) \in A$, then $(j, i) \notin A$. This assumption is nonrestrictive because if $(j, i) \in A$, then we replace the arc (j, i) with two arcs (j, k) and (k, i) with $l(j, k) = l(k, i) = l(j, i)$, $u(j, k) = u(k, i) = u(j, i)$.

We define the dual directed network $G' = (N', A', l', u')$ of a directed $(1, n)$ planar network $G = (N, A, l, u)$ as follows. We first draw an arc $(n, 1)$ with $l(n, 1) = u(n, 1) = 0$. The arc $(n, 1)$ divides the unbounded face of G into two faces: a new unbounded face and a new bounded face. Then we place a node i' inside each face x of the network G. Let $1'$ and n', respectively, denote the nodes in the network G' corresponding to the new bounded face and the new unbounded face. Each arc $(i, j) \in A$ lies on the boundary of the two faces x and y. Corresponding to this arc, the dual directed network contains two opposite arcs (i', j') and (j', i'). For minimum flow problem, if the arc (i, j) is a clockwise arc in the face x, we define $l'(i', j') = l(i, j)$ and $u'(j', i') = -u(i, j)$. We define l' and u' in the opposite manner if arc (i, j) is a counterclockwise arc in the face x. The dual directed network G' contains the arcs $(1', n')$ and $(n', 1')$ which we delete from the network G'. We have $N' = \{1', \ldots, i', \ldots, n'\}, A' = \{(i', j'), (j', i') | (i, j) \in A\}$ with $n' = |N'| = m - n + 3$ and $m' = |A'| = 2m$.

The algorithm for minimum flow $\overset{\circ}{f}$ in a static $(1, n)$ planar network $G = (N, A, l, u)$ is presented below ([1]):

Algorithm 2 The algorithm for minimum flow in a static $(1, n)$ planar network $G = (N, A, l, u)$.

1: mFSPN
2: BEGIN
3: BELLMAN-FORD (G', d');
4: **for** $(i, j) \in A$ **do**
5: $\quad \overset{\circ}{f}(i, j) := d'(j') - d'(i')$;
6: **end for**
7: END.

The procedure BELLMAN-FORD performs the algorithm of Bellman-Ford to determine the distance vector d' for longest path problem from node $1'$ to node n' in network G'. We can obtain a maximum $1 - n$ cut for minimum flow problem in the network G by determining a longest path from node $1'$ to node n' in network G'.

From paper [1] we present the following theorems.

Theorem 6 If the flow $\overset{\circ}{f}$ determined with the algorithm mFSPN satisfies the conditions $\overset{\circ}{f}(i, j) \leq u(i, j)$ for all $(i, j) \in A$, then the flow $\overset{\circ}{f}$ is a minimum feasible flow.

Theorem 7 The algorithm mFSPN has complexity $O(n^2)$.

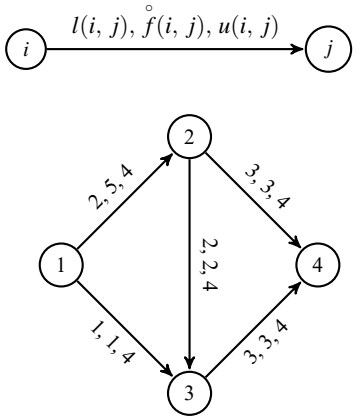

Figure 1. The minimum flow $\overset{\circ}{f}$ is infeasible

We remark that if the minimum flow $\overset{\circ}{f}$ obtained with the algorithm mFSPN is infeasible $(u(i, j) < \overset{\circ}{f}(i, j))$ for one or more arcs $((i, j) \in A)$, then there is no the feasible flow in the network G. For example, the minimum flow,

from Figure 1, $\overset{\circ}{f}$ obtained with the algorithm mFDPN is infeasible because $u(1, 2) = 4 < 5 = f(1, 2)$. If we consider $X = \{1, 3, 4\}, \bar{X} = \{2\}$ we have $u(X, \bar{X}) = 4 < 5 = l(\bar{X}, X)$ and from Theorem 1 we obtain that in network G there is not a feasible flow.

For maximum flow problem, if the arc (i, j) is a clockwise arc in the face x, we define $u^{'}(i^{'}, j^{'}) = u(i, j)$ and $l^{'}(j^{'}, i^{'}) = -l(i, j)$. We define $l^{'}$ and $u^{'}$ in the opposite manner if arc (i, j) is a counterclockwise arc in the face x.

The algorithm for maximum flow in a static $(1, n)$ planar network $G = (N, A, l, u)$ is presented below.

Algorithm 3 The algorithm for maximum flow in a static $(1, n)$ planar network $G = (N, A, l, u)$.

1: MFSPN
2: BEGIN
3: DIJKSTRA $(G^{'}, d^{'})$;
4: **for** $(i, j) \in A$ **do**
5: $\overset{*}{f}(i, j) := d^{'}(j^{'}) - d^{'}(i^{'})$;
6: **end for**
7: END.

The procedure DIJKSTRA performs the algorithm of Dijkstra to determine the distance vector $d^{'}$ for the shortest path problem from node $1^{'}$ to node $n^{'}$ in the network $G^{'}$. We can obtain a minimum $1 - n$ cut for maximum flow problem in the network G by determining a shortest path from node $1^{'}$ to node $n^{'}$ in the network $G^{'}$.

For the maximum flow problem in planar static network G, we have two theorems similarly with Theorem 6 and Theorem 7.

5. Optimum flows in planar dynamic networks. The dynamic approach

In this section we consider the optimum flows in $(1, n)$ stationary dynamic networks [2, 3]. There are two inconveniences for this problem. The first is that although in $(1, n)$ planar static network $D = (N, A, c = h, l, Q)$ exists a feasible flow, it is possible that in the static reduced expanded network $R_1 = (V_1, E_1, l_1, u_1)$ will not be any feasible flow. The second inconvenience consists in the fact that it is possible that the temporally repeated flow of a feasible and minimum time flow in network D, will not be feasible in network R_0 although in R_0 there is a feasible flow. These inconveniences are shown in the example presented in Section 6.

Firstly, we present an algorithm for the determination of a feasible flow in static network $R_0 = (V_0, E_0, l_0, u_0)$.

We suppose that the flow $f = \overset{\circ}{f}$, determined with the algorithm mFSPN presented in Section 4 in the static network $D = (N, A, c = h, l, q)$, is feasible. Let P_s be any path determined with procedure ADSFEF from the algorithm MDFSDN presented in algorithm 1 from Section 3 with $r(P_s)$ the flow, $h(P_s)$ the transit time and $\nu(P_s) = (T + 1) - h(P_s)$ the number of repetitions of path P_s, $s = \{1, \ldots, k\}$. We consider that $h(P_1) \leq \ldots \leq h(P_k)$. If the path P_s is $P_s = ((1, x), \ldots, (y, i), (i, j), \ldots, (z, n))$, then we define $h_i(P_s) = h(1, x) + \ldots + h(y, i)$ and $\bar{h}_i(P_s) = h(P_s) - h_i(P_s)$. Obviously, $h_n(P_s) = h(P_s)$. Let H_i, $i \in N$ be the sets defined in Section 3 with the specification that dynamic network D is stationary. We define the sets $H_i^s = \{t | t \in H_i, h_i(P_s) \leq t \leq T - \bar{h}_i(P_s)\} = \{h_i(P_s), h_i(P_s) + 1, \ldots, T - h_i(\bar{P}_s)\}$ with $|H_i^s| = \nu(P_s), i \in P_s$ and $H_i^s = \emptyset$ for $i \notin P_s$, $s = 1, \ldots, k$. The accordingly path in network R_0 are $P_s^t = (1_t, \ldots, i_{\gamma+t}, \ldots, n_{\eta+t}), t = 0, \ldots, k_s$, $k_s = \nu(P_s) - 1$, $\gamma = h_i(P_s)$, $\eta = h_n(P_s) = h(P_s)$, $s = 1, \ldots, k$. If the dynamic network is stationary, then we have $d(1, i; t) = d(1, i), d(i, n; t) = d(i, n)$ for all $i \in N$ and are performing with usual shortest path algorithm.

We consider that the arcs in A and in E_0 are arranged in some order. The $e = (e(i, j)), q = (q(i, j)), g = (g(i, j; t))$ and $l_0 = (l_0(i_t, j_\theta)), u_0 = (u_0(i_t, j_\theta)), f_0 = (f_0(i_t, j_\theta))$ denote the lower bound vector, upper bound vector and flow vector in which these components are ordered in the same order as arcs are in A and respectively in E_0. Our generation of g is keyed to the static flow in static network $D = (N, A, c = h, e, q)$ and we don't actually construct the static network R_0 but we frequently refer to its existence.

We define the list $E_0^{'} = (a_1^{'}, a_2^{'}, \ldots, a_\alpha^{'})$ with property that $a_i^{'} \in E_0$, $f_0(a_i^{'}) < l_0(a_i^{'})$, $i = 1, 2, \ldots, \alpha$, where f_0 is a temporally repeated flow in network R_0 generate of feasible and minimum time flow f^*. The flow f_0 is generated by algorithm MDFSDN presented in Section 3 in which the procedure AMVMCSF uses the algorithm mFSPN presented

in Section 4 which determines a feasible flow $f = \overset{\circ}{f}$ and the minimum mean cycle canceling algorithm for determines a feasible and minimum time flow f^* in static network $D = (N, A, c = h, e, q)$. Let $k_i^{'}$ the number of paths P_s^t which contain the arc $a_i^{'}$, $i = 1, 2, \ldots, \alpha$. If exist arc $a_j^{'} \in E_0^{'}$ and $a_j^{'} \notin P_s^t$, $s = 1, 2, \ldots, k$, $t = 0, \ldots, k_s$, then we can determine easily the path P_{k+1}^t which contains the arc $a_j^{'}$. We select the arcs $a_i^{'}$ from $E_0^{'}$ in breeder order of numbers $k_i^{'}$. Let $\mathscr{P}_0$ be $\mathscr{P}_0 = \{P_s^t | (i_t, j_\theta) \in P_s^t, (i_t, j_\theta) \in E_0^{'}\}$.

If $E_0^{'} = \emptyset$ the f_0 is a feasible flow in network R_0, else we determine a feasible flow $\overset{\circ}{f}_0$ in network R_0 with the procedure $AFFR_0$ presented in Algorithm 4.

Algorithm 4 The algorithm for a feasible flow in R_0.

1: $AFFR_0(l_0, u_0, f_0, E_0^{'}, \mathscr{P}_0, k_1^{'}, \ldots, k_\alpha^{'})$;
2: BEGIN
3: $\overset{\circ}{f}_0 := f_0$; $\beta := 1$;
4: **repeat**
5: select $a_i^{'}$ from $E_0^{'}$ with $k_i^{'}$ minim;
6: select P_s^t from $\mathscr{P}_0$ with $a_i^{'} \in P_s^t$;
7: let $\hat{P}_s^t := P_s^t - \{a_i^{'}\}$;
8: $r(\hat{P}_s^t) := min\{u_0(i_t, j_\theta) - f_0(i_t, j_\theta) | (i_t, j_\theta) \in \hat{P}_s^t\}$;
9: **if** $l_0(a_i^{'}) - f_0(a_i^{'}) \leq r(\hat{P}_s^t)$
10: **then** BEGIN
11: $r(P_s^t) := l_0(a_i^{'}) - f_0(a_i^{'})$;
12: augment $\overset{\circ}{f}_0$ with $r(P_s^t)$ along the path P_s^t;
13: eliminate from $E_0^{'}$ the arcs $a_i^{'}$ with $l_0(a_i^{'}) \leq \overset{\circ}{f}_0(a_i^{'})$;
14: END
15: **else** $\beta := 0$;
16: **if** $\beta = 0$
17: **then** Exit;
18: **until** $E_0^{'} = \emptyset$;
19: **if** $\beta = 1$
20: **then** $\overset{\circ}{f}_0$ is a feasible flow in R_0
21: **else** there is no feasible flow in R_0;
22: END

In paper [3] we prove the following two theorems.

Theorem 8 If $E_0^{'} \neq \emptyset$ and exist a feasible flow in $(1, n)$ directed planar dynamic network $D = (N, A, h, e, q)$ then the procedure $AFFR_0$ determines a feasible flow in D, else this procedure specifies that there is not feasible flow in D.

Theorem 9 The procedure $AFFR_0$ has the complexity $O(n^2T^2)$.

In paper [2] we prove the following theorem.

Theorem 10 The feasible flow determined with the procedure $AFFR_0$ is a minimum feasible flow in $(1, n)$ directed planar dynamic network $D = (N, A, h, e, q)$.

Below we present an algorithm for the maximum and minimum time feasible flow $\overset{*}{f}^*$ in the static network $G = (N, A, c = h, e, q)$ which generates a maximum feasible flow $\overset{*}{f}_0$ in the static network R_0. The algorithm for the maximum and minimum time feasible flow $\overset{*}{f}^*$ in the static network $D = (N, A, c = h, e, q)$ is basically the same as the algorithm of Ahuja-Orlin of layered networks for maximum flow problem [7]. We make the following modification to the algorithm of Ahuja-Orlin of layered networks: the exact distance labels $d(i)$ are the distances from node 1 concerning the cost $c = h$, which can be solved by a classical algorithm. We remark that this algorithm is a variant of successive shortest path algorithm [7] which has the complexity $O(n\bar{q} \cdot O(n, m))$ with $\bar{q} = max\{q(i, j) | ((i, j) \in A)\}$ and $O(n, m)$ denotes the complexity of algorithm used to solve the shortest path problem. The algorithm to generate a maximum feasible flow $\overset{*}{f}_0$

in the static network R_0 is basically the same as the algorithm of Wilkinson algorithm [13]. The variant of the Wilkinson algorithm (VWA) is presented in algorithm 5.

Algorithm 5 The variant of Wilkinson algorithm.

```
1: VWA;
2: BEGIN
3:    f*: = f*; f0*: = f0;
4:    construct the residual network G̃ concerning f*;
5:    obtain the exact distance labels d(i) in G̃;
6:    for i ∈ N do
7:       b(i): = true;
8:    i: = 1;
9:    while d(n) ≤ T do
10:      if b(1) = true
11:        then
12:          if exists an admissible arc(i, j)
13:            then BEGIN
14:                    ADVANCE (i);
15:                    if i = n
16:                      then BEGIN
17:                             AUGUMENT;
18:                             i: = 1;
19:                           END;
20:                  END
21:            else RETREAT(i)
22:        else BEGIN
23:               determine the exact distance labels d(i) in G̃;
24:               for i ∈ N do
25:                  b(i): = true;
26:               i: = 1;
27:             END;
28: END.
```

```
1: PROCEDURE ADVANCE(i);
2: BEGIN
3:    p̃(j): = i;
4:    i: = j;
5: END;
```

```
1: PROCEDURE RETREAT(i);
2: BEGIN
3:    b(i): = false;
4:    if i ≠ 1
5:       then i: = p̃(i);
6: END;
```

1: PROCEDURE AUGMENT;
2: BEGIN
3: identify an augmenting path $\tilde{P}$ using the predecessor vector $\tilde{p}$;
4: $r(\tilde{P}) := min\{r(i, j)|(i, j) \in \tilde{A}\}$;
5: update the residual network $\tilde{G}$;
6: identify the chains C_0^t in R_0 corresponding to path $\tilde{P}$;
7: determine $r(C_0^t)$ for all t;
8: augment the flow $\overset{*}{f}_0$ with $r(C_0^t)$ along chain C_0^t for all t;
9: END.

The network $\tilde{D}$ is the residual network of the static network $D = (N, A, c = h, e, q)$. We recall that an arc (i, j) in the residual network $\tilde{D}$ is admissible if it satisfies the conditions that $d(j) = d(i) + 1$ and node j is not blocked ($b(j) =$ true) with $d(i), d(j)$ the distances concerning the cost $c = h$. The chains C_0^t in static network R_0 are identified in the same way as the paths P_s^t corresponding to path P_s which is shown above.

In paper [3] we prove the following two theorems.

Theorem 11 The VWA determines a maximum feasible flow in a $(1, n)$ planar dynamic network $D = (N, A, h, e, q)$.

Theorem 12 The VWA has the complexity $O(n^2T^2\bar{q})$, where $\bar{q} = max\{q(i, j)|(i, j) \in A\}$.

Now we present, below, the complete algorithm for maximum feasible flow in the $(1, n)$ planar dynamic network (CAMFPDN) problem.

Algorithm 6 The complete algorithm for maximum feasible flow in $(1, n)$ planar dynamic network.

1: CAMFFPDN
2: BEGIN
3: MDFSDN($G, \overset{\circ}{f}, f^*, f_0$);
4: **if** f_0 is not feasible
5: **then** $AFFR_0(R_0, f_0, E_0', \mathscr{P}_0, k_1', \ldots, k_\alpha')$
6: VWA($G, f^*, R_0, f_0, \overset{*}{f}{}^*, \overset{*}{f}_0$);
7: END.

The procedure MDFSDN presented in algorithm 1 from Section 3 uses the algorithm mFSPN presented in algorithm 2 from Section 4 which determines a feasible flow $f = \overset{\circ}{f}$ and the minimum mean cycle canceling algorithm. This algorithm determines a feasible and minimum cost (time) flow f^* in the static network $D = (N, A, c = h, e, q)$. Also, the procedure MDFSDN generates the flow f_0 in the static network R_0 from feasible and minimum time f^*. If the flow f_0 is not feasible, then it is performed the procedure $AFFR_0$ which supplies a feasible flow f_0. The procedure VWA provides a maximum and minimum time feasible flow f^* in the static network $G = (N, A, h, e, q)$ and a maximum flow $\overset{*}{f}_0$ in the static network R_0. Obviously, if $f = \overset{\circ}{f}$ is not feasible then f_0 is not feasible as well.

In paper[3] we prove the following two theorems.

Theorem 13 The CAMFFPDN algorithm determines a maximum feasible flow in $(1, n)$ planar dynamic network $D = (N, A, h, e, q)$.

Theorem 14 The CAMFFPDN algorithm has the complexity $O(n^2T^2\bar{q})$.

6. Example

The support graph of $(1, 6)$ planar stationary dynamic network $D = (N, A, h, e, q)$ is presented in Figure 2 and the time horizon set to $T = 5$, therefore $H = \{0, 1, 2, 3, 4, 5\}$. The transit times $h(i, j)$, the lower bounds $e(i, j)$, and the upper bounds $q(i, j)$ for all arcs $(i, j) \in A$ are indicated in Table 1.

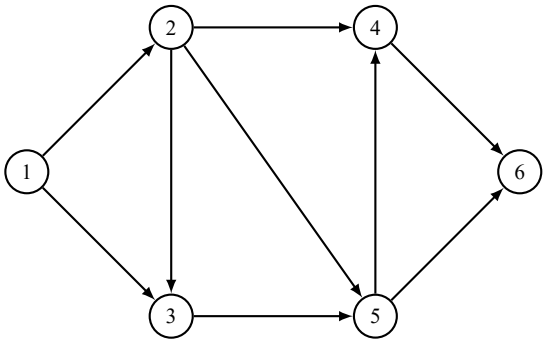

Figure 2. The support graph of $(1, 6)$ planar dynamic network

Table 1. The transit time h, the lower bound e, the upper bound q, the flow $f = \overset{\circ}{f}$ and $\overset{*}{f}$ for all $(i, j) \in A$

(i, j)	$h(i, j)$	$e(i, j)$	$q(i, j)$	$\overset{\circ}{f}(i, j)$	$\overset{*}{f}(i, j)$
$(1, 2)$	1	4	8	4	8
$(1, 3)$	1	1	6	2	6
$(2, 3)$	1	1	2	2	2
$(2, 4)$	2	0	4	1	4
$(2, 5)$	2	1	4	1	2
$(3, 5)$	1	4	8	4	8
$(4, 6)$	1	1	6	3	6
$(5, 4)$	1	1	6	2	2
$(5, 6)$	2	3	8	3	8

We obtain the vector $d' = (0, 2, 4, 5, 3, 6)$ and the minimum flow $\overset{\circ}{f}$ is presented in Table 1.

Because the minimum flow $\overset{\circ}{f}$ is a feasible flow we further determine with minimum mean cycle canceling algorithm, a feasible and minimum time flow $f^* = f = \overset{\circ}{f}$. The results of procedure ADSFEF are presented in Table 2.

Table 2. The results of procedure ADSFEF

P_s	$r(P_s)$	$h(P_s)$	$v(P_s)$
$P_1 = ((1, 2), (2, 4), (4, 6))$	1	4	2
$P_2 = ((1, 3), (3, 5), (5, 6))$	2	4	2
$P_3 = ((1, 2), (2, 5), (5, 6))$	1	5	1
$P_4 = ((1, 2), (2, 3), (3, 5), (5, 4), (4, 6))$	2	5	1

Figure 3 shows the support graph of $(1, 6)$ planar static network $G = (N, A, c = h, l = e, u = q)$ and the attache dual network $G' = (N', A', l', u')$.

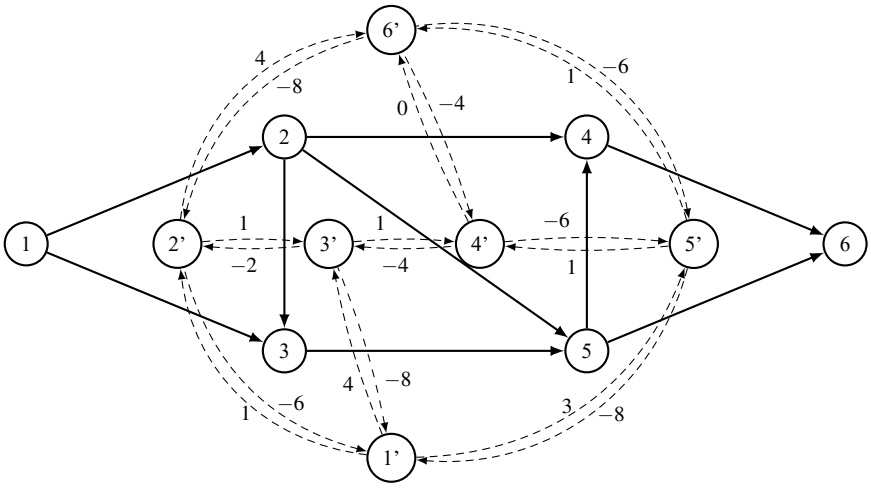

Figure 3. The support graph $G = (N, A)$ and the dual network G'

We obtain $d(1, 1) = 0$, $d(1, 6) = 4$, $d(1, 2) = 1$, $d(2, 6) = 3$, $d(1, 3) = 1$, $d(3, 6) = 3$, $d(1, 4) = 3$, $d(4, 6) = 1$, $d(1, 5) = 2$, $d(5, 6) = 2$, $d(6, 6) = 0$ and $H_1 = \{0, 1\}$, $H_2 = \{1, 2\}$, $H_3 = \{1, 2\}$, $H_4 = \{3, 4\}$, $H_5 = \{2, 3\}$, $H_6 = \{4, 5\}$, $H_{1,2} = \{0, 1\}$, $H_{1,3} = \{0, 1\}$, $H_{2,3} = \{1\}$, $H_{2,4} = \{1, 2\}$, $H_{2,5} = \{1\}$, $H_{3,5} = \{1, 2\}$, $H_{4,6} = \{3, 4\}$, $H_{5,4} = \{2, 3\}$, $H_{5,6} = \{2, 3\}$.

The support graph of static network $R_0 = (V_0, E_0, l_0, u_0)$ is presented in Figure 4.

For $q(2, 4) = 3$ the flow $\overset{\circ}{f}$ presented in Figure 4 is a feasible flow in static network $D = (N, A, c = h, l, q)$. If $q(2, 4) = 3$ then for $Y_0 = \{2_2\}$, $\bar{Y}_0 = V_0 - Y_0$ we have $u_0(Y_0, \bar{Y}_0) = u_0(2_2, 4_4) = q(2, 4) = 3 < 4 = e_0(1, 2) = l_0(1_1, 2_2) = l_0(\bar{Y}_0, Y_0)$. From Theorem 1 we obtain that the flow problem in static network R_0 (dynamic network D) is infeasible.

The procedure ARPF from MDFSDN generates the flow f_0 in static network R_0 and is given in Figure 4 in the form $\overset{\circ}{k}$. We have: $k_1 = 1$, $k_2 = 1$, $k_3 = 0$, $k_4 = 0$; $P_1^0 = ((1_0, 2_1), (2_1, 4_3), (4_3, 6_4))$, $P_1^1 = ((1_1, 2_2), (2_2, 4_4), (4_4, 6_5))$, $P_2^0 = ((1_0, 3_1), (3_1, 5_2), (5_2, 6_4))$, $P_2^1 = ((1_1, 3_2), (3_2, 5_3), (5_3, 6_5))$, $P_3^0 = ((1_0, 2_1), (2_1, 5_3), (5_3, 6_5))$, $P_4^0 = ((1_0, 2_1), (2_1, 3_2), (3_2, 5_3), (5_3, 4_4), (4_4, 6_5))$. The list E_0' is $E_0' = (a_1', a_2', a_3', a_4') = ((1_1, 2_2), (3_1, 5_2), (5_2, 4_3), (5_2, 6_4))$ with $(1_1, 2_2) \in P_1^1$, $(3_1, 5_2) \in P_2^0$, $P_5^0 = ((1_0, 3_1), (3_1, 5_2), (5_2, 4_3), (4_3, 6_4))$, $(5_2, 4_3) \in P_2^0$, $(5_2, 6_4) \in P_2^0$ and $k_1' = 1$, $k_2' = 2$, $k_3' = 1$, $k_4' = 1$. The results of procedure $AFFR_0$ are: $r(P_1^1) = 3$, $E_0' = (a_2', a_3', a_4')$, $r(P_2^0) = 1$, $E_0' = (a_2', a_4')$, $r(P_5^0) = 1$, $E_0' = \text{ø}$. The minimum feasible flow $f_0 = \bar{f}_0$ in network R_0 is given in Figure 4 in the form $\bar{k}$. In network R_0 for maximum cut we have $\overset{\circ}{Y}_0 = \{1_0, 1_1, 3_1, 3_2\}$, $[\overset{\circ}{Y}_0, \overset{\circ}{\bar{Y}}_0] = (\overset{\circ}{Y}_0, \overset{\circ}{\bar{Y}}_0) \cup (\overset{\circ}{\bar{Y}}_0, \overset{\circ}{Y}_0) = \{(1_0, 2_1), (1_1, 2_2), (3_1, 5_2), (3_2, 5_3)\} \cup \{(2_1, 3_2)\}$. The value of flow $\overset{\circ}{f}$ is $\overset{\circ}{w}_0 = l_0(\overset{\circ}{Y}_0, \overset{\circ}{\bar{Y}}_0) - u_0(\overset{\circ}{\bar{Y}}_0, \overset{\circ}{Y}_0) = l_0(1_0, 2_1) + l_0(1_1, 2_2) + l_0(3_1, 5_2) + l_0(3_2, 5_3) - u_0(2_1, 3_2) = \overset{\circ}{f}_0(1_0, 2_1) + \overset{\circ}{f}_0(1_1, 2_2) + \overset{\circ}{f}_0(3_1, 5_2) + \overset{\circ}{f}_0(3_2, 5_3) - \overset{\circ}{f}_0(2_1, 3_2) = 4 + 4 + 4 + 4 - 2 = 14$.

The execution of procedure VWA supplies the follows results: $r(P_1) = 3$, $r(P_1^0) = 3$, $r(P_1^1) = 0$, $r(P_5) = 4$, $r(P_5^0) = 2$, $r(P_5^1) = 4$, $r(P_3) = 1$, $r(P_3^0) = 1$. The flow $\overset{*}{f}{}^*$ is presented in Table 1 and the flow $\overset{*}{f}_0$ is presented in Figure 4 and is given in the form $\overset{*}{k}$. In the network G we have $\overset{*}{X} = \{1\}$, $\overset{*}{\bar{X}} = N - \overset{*}{X}$, $[\overset{*}{X}, \overset{*}{\bar{X}}] = (\overset{*}{X}, \overset{*}{\bar{X}}) \cup (\overset{*}{\bar{X}}, \overset{*}{X}) = (\overset{*}{X}, \overset{*}{\bar{X}}) = \{(1, 2), (1, 3)\}$ and $\overset{*}{v} = \overset{*}{f}{}^*(\overset{*}{X}, \overset{*}{\bar{X}}) - \overset{*}{f}{}^*(\overset{*}{\bar{X}}, \overset{*}{X}) = \overset{*}{f}(\overset{*}{X}, \overset{*}{\bar{X}}) = \overset{*}{f}{}^*(1, 2) + \overset{*}{f}{}^*(1, 3) = 8 + 6 = u(1, 2) + u(1, 3) = u(\overset{*}{X}, \overset{*}{\bar{X}}) - l(\overset{*}{\bar{X}}, \overset{*}{X}) = \kappa[\overset{*}{X}, \overset{*}{\bar{X}}]$. In network R_0 we have $[\overset{*}{Y}_0, \overset{*}{\bar{Y}}_0] = (\overset{*}{Y}_0, \overset{*}{\bar{Y}}_0) \cup (\overset{*}{\bar{Y}}_0, \overset{*}{Y}_0) = (\overset{*}{Y}_0, \overset{*}{\bar{Y}}_0) = \{(1_0, 2_1), (1_0, 3_1), (1_1, 3_2), (2_2, 4_4)\}$ and $\overset{*}{w}_0 = u_0(\overset{*}{Y}_0, \overset{*}{\bar{Y}}_0) - l_0(\overset{*}{\bar{Y}}_0, \overset{*}{Y}_0) = \overset{*}{f}_0(\overset{*}{Y}_0, \overset{*}{\bar{Y}}_0) - \overset{*}{f}_0(\overset{*}{\bar{Y}}_0, \overset{*}{Y}_0) = f(\overset{*}{Y}_0, \overset{*}{\bar{Y}}_0) = 24$. From (19) we obtain $\overset{*}{w}_0 = (5 + 1) \cdot 14 - (8 \cdot 1 + 6 \cdot 1 + 2 \cdot 1 + 2 \cdot 2 + 4 \cdot 2 + 8 \cdot 1 + 6 \cdot 1 + 2 \cdot 1 + 8 \cdot 2) = 24$.

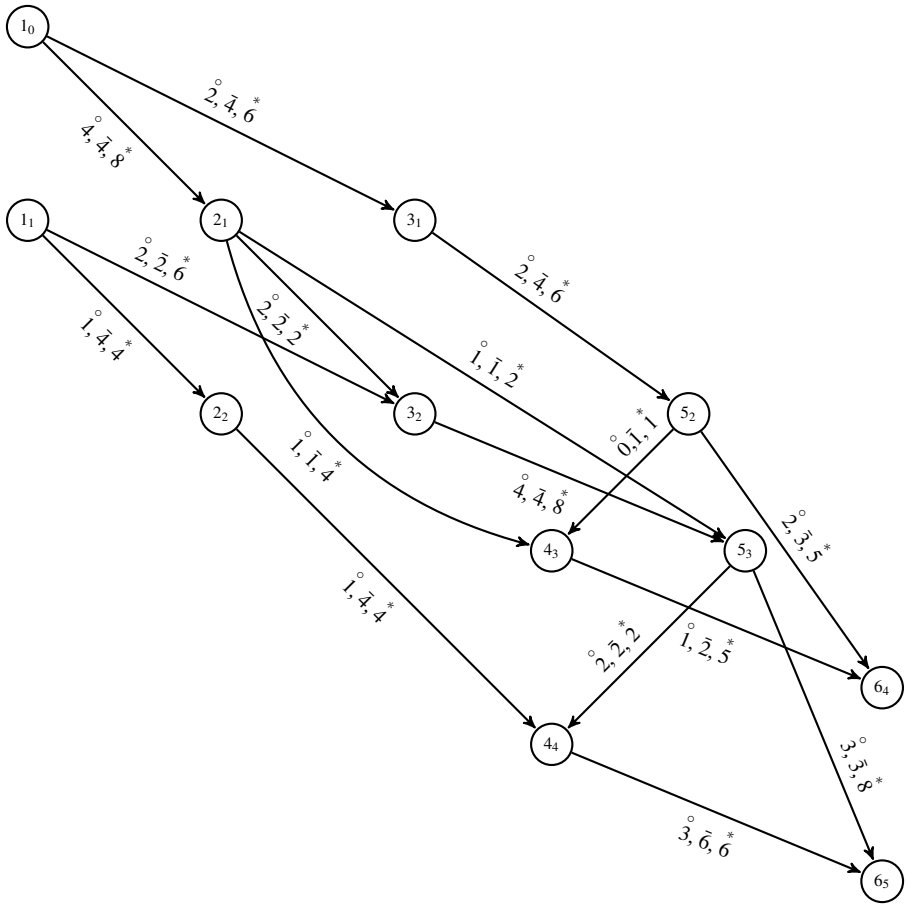

Figure 4. The support graph of static network R_0

7. Conclusions

In conclusion, the study of network flow problems, especially in the context of planar networks and the dynamic approach, reveals their critical importance in a diverse range of disciplines and practical applications. From optimizing transport routes and real-time traffic management to improving data transmission efficiency and market dynamics, the complexities of flow dynamics underscore the need for effective algorithmic solutions.

Flows in directed planar dynamic networks have multiple interdisciplinary practical applications due to their ability to model and optimize processes involving flows of resources, information, or objects between different nodes in a network. Some areas where these directed planar dynamic networks may have applications are: transportation and logistics, financial and economic systems, medicine and computational biology, telecommunications and data networks, applications in artificial intelligence and machine learning, and of course in other domains.

Transport route optimization can be used to optimize the flow of vehicles (cars, trains, trucks, etc.) in transport networks (highways, railways) to minimize travel times and optimize costs. Dynamic elements account for changes in traffic or network capabilities over time. In logistics, flows can be used to optimize deliveries and the distribution route through a network of warehouses and distribution centers so as to minimize costs and maximize delivery efficiency. Modeling capital flows in an interconnected economy can help optimize the allocation of funds between different financial institutions or capital markets. Directed planar networks can simulate how funds move over time and how market dynamics affect transactions.

In economics and management, flows of goods and services can be optimized in a dynamic network of suppliers and distributors, given changing demand over time and the need to maintain optimal inventory levels.

In the management of medical institutions, dynamic flows can model the movements of patients between different departments of hospitals, optimizing the waiting time and the allocation of resources (beds, doctors). In molecular biology studies, directed flows can dynamically model biological processes, such as chemical reactions in metabolic networks or genetic information flows in protein interaction networks.

With the help of directed dynamic planar networks we can model communication networks, optimizing data flows between routers and servers to minimize latency times and maximize transfer speed. Dynamic flows can also optimize communication between geographically distributed IoT devices, ensuring efficient data flows and minimizing power consumption and network congestion.

In artificial neural networks (such as those used in deep learning), dynamic flows can model the propagation of information through the nodes (neurons) of the network, optimizing learning processes and updating synapse weights.

This paper has highlighted the evolution and advances in addressing these challenges, using advanced algorithms and data structures to improve theoretical performance and practical results. By integrating theoretical insights with real-world applications, the research makes a meaningful contribution to the subject of network flow problems, paving the way for future innovations and advances in this field of study.

Many interesting flows problem in planar dynamic networks are still open: the minimum flows in directed planar dynamic networks with arcs and nodes capacities in dynamic approach, the maximum flows in directed planar dynamic networks with arcs and nodes capacities in static approach and in dynamic approach. Other research directions are possible, such as expanding the scope of the study by focusing on networks that include loops and parallel arcs. This extension would allow a deeper investigation of the behavior of these complex structures, providing opportunities for the development of new theoretical models and practical applications in various related fields. Other future research directions include problems like:

• Maximum flows in directed $(1, n)$ planar dynamic networks, where the transit times, the capacities of arcs are all time-varying.

• The maximum flow in directed $(1, n)$ planar dynamic networks with lower bounds in stationary case and in dynamic case.

Conflict of interest

The authors declare no competing financial interest.

References

[1] Ciurea E, Georgescu O. Minimum flows in directed s-t planar networks. *Bulletin Mathématique de la Société des Sciences Mathématiques de Roumanie*. 2010; 101(4): 305-313.

[2] Ciurea E, Schiopu C. Minimum flows in directed planar dynamic networks. In: *Proceedings of the 6th International Conference on Control, Decision and Information Technologies*. Paris, France: IEEE; 2019. p.1746-1751.

[3] Schiopu C, Ciurea E. Maximum flows in planar dynamic networks with lower bounds. *Fundamenta Informaticae*. 2018; 163: 189-204.

[4] Ford L, Fulkerson D. Maximal flow through a network. *Canadian Journal of Mathematics*. 1956; 8: 399-404.

[5] Lipton RJ, Tarjan RE. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*. 1979; 36(2): 177-189.

[6] Cooke KL, Halsey E. The shortest route through a network with time-dependent internodal transit times. *Journal of Mathematical Analysis and Applications*. 1966; 14(3): 493-498.

[7] Ahuja R, Magnanti T, Orlin J. *Network Flows. Theory, Algorithms and Applications*. NJ: Prentice hall; 1993.

[8] Ciurea E, Ciupală L. Sequential and parallel algorithms for minimum flows. *Journal of Applied Mathematics and Computing*. 2004; 15(1-2): 53-75.

[9] Ford L, Fulkerson D. *Flows in Networks*. NJ: Princeton; 1962.
[10] Cai X, Sha D, Wong C. *Time-Varying Network Optimization*. New York: Springer; 2007.
[11] Ciurea E. An algorithm for minimal dynamic flow. *Korean Journal of Computational and Applied Mathematics*. 2000; 7: 259-270.
[12] Salehi HF, Khadayifar S, Raayatpanoh M. Minimum flow problem on network flows with time-varying bounds. *Applied Mathematical Modelling*. 2012; 36: 4414-4421.
[13] Wilkinson W. An algorithm for universal maximal dynamic flows in a network. *Operations Research*. 1971; 19(7): 1602-1612.
[14] Skutella M. An introduction to network flows over time. In: *Research Trends in Combinatorial Optimization*. Berlin, Heidelberg: Springer; 2009. p.451-482. Available from: https://doi.org/10.1007/978-3-540-76796-1_21.
[15] Borradaile G, Klein P. An $O(n\log n)$ algorithm for maximum st-flow in a directed planar graph. *Journal of the ACM*. 2009; 56: 1-30.
[16] Hassin R. Maximum flows in $(s-t)$ planar networks. *Information Processing Letters*. 1981; 13: 107.
[17] Hassin R, Johnson D. An $O(n\log^2 n)$ algorithm for maximum flow in undirected planar networks. *SIAM Journal on Computing*. 1985; 14: 612-624.
[18] Itai A, Shiloach Y. Maximum flows in planar networks. *SIAM Journal on Computing*. 1979; 8: 135-150.
[19] Khuller S, Naor J. Flows in planar graphs with vertex capacities. *Algorithmica*. 1994; 11: 200-225.