GOCE DELCEV UNIVERSITY - STIP
FACULTY OF COMPUTER SCIENCE

ISSN 2545-4803 on line

BALKAN JOURNAL OF APPLIED MATHEMATICS AND INFORMATICS (BJAMI)

BALKAN JOURNAL
OF APPLIED MATHEMATICS AND INFORMATICS

YEAR 2019

VOLUME II, Number 2

GOCE DELCEV UNIVERSITY - STIP, REPUBLIC OF NORTH MACEDONIA
FACULTY OF COMPUTER SCIENCE

ISSN 2545-4803 on line

BALKAN JOURNAL OF APPLIED MATHEMATICS AND INFORMATICS

(BJAMI)

YEAR 2019 VOLUME II, Number 2

AIMS AND SCOPE:
BJAMI publishes original research articles in the areas of applied mathematics and informatics.

Topics:
1. Computer science;
2. Computer and software engineering;
3. Information technology;
4. Computer security;
5. Electrical engineering;
6. Telecommunication;
7. Mathematics and its applications;
8. Articles of interdisciplinary of computer and information sciences with education, economics, environmental, health, and engineering.

Address of the editorial office
Goce Delcev University – Stip
Faculty of philology
Krste Misirkov 10-A
PO box 201, 2000 Štip,
Republic of North Macedonia

BALKAN JOURNAL
OF APPLIED MATHEMATICS AND INFORMATICS (BJAMI), Vol 2

ISSN 2545-4803 on line
Vol. 2, No. 2, Year 2019

EDITORIAL BOARD

CONTENT

UDC 004.7.021

VISUALIZATION OF FORD-FULKERSON ALGORITHM

**Natasha Stojkovikj, Mirjana Kocaleva, Aleksandra Stojanova,
Isidora Janeva and Biljana Zlatanovska**

Abstract. This paper examines the Ford-Fulkerson algorithm for finding the maximum flow in the flow network. For this purpose, we first give the basic definitions of the flow, the residual network and the augmenting path. Also, a program for visualizing the Ford Fulkerson algorithm has been made (in Java) in order for students to understand the algorithm more easily.

1. Introduction

In the mid-fifties of the last century, scientists T. E. Harris and F. S. Ross from the Air Force, ıblished a confidential article in which they researched the connection of the railway network :tween the Soviet Union and its satellite countries in Eastern Europe. The network was modelled as graph with 44 nodes, which represent geographic regions and 105 links which represent the ɔnnection of those regions with the railway networks. Every link has a given weight, which presents the amount of material that can be transported from one region to another. In fact, through al and error, they determined the maximal amount of materials that can be transported from Russia Europe, as well as the cheapest way to interrupt the network with the erasing (removing) of some ıks, which they called "bottleneck". Their results were published in 1999. These results also cluded the map of the Warsaw pact railway network which is shown in Figure 1 [3].

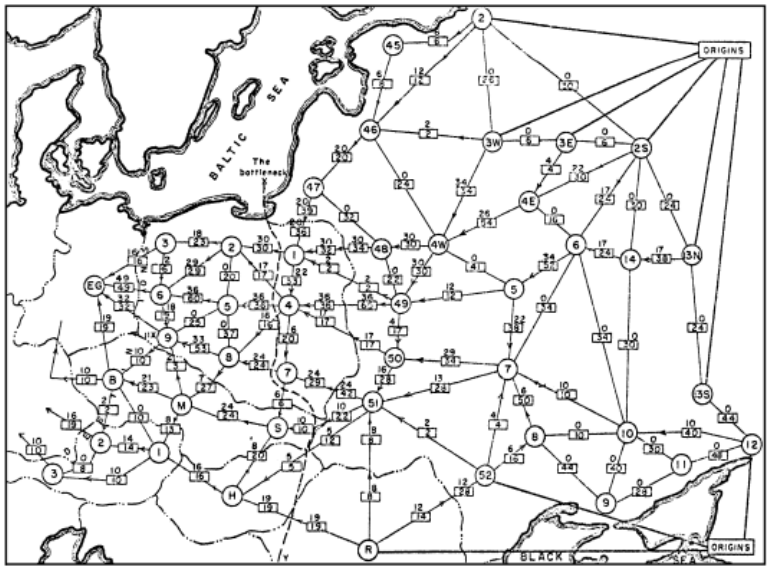


Figure 1 *The map of Harris and Ross of the Warsaw Pact Railway Network*

These are among the first observed applications of maximum flow problems and a minimum cut oblem. For both problems, the transport system can be represented by a directed graph with two ɨecial nodes, a source, and a sink. Through the network you need to transfer a certain amount of aterial from the source (where the material is produced) to the sink (where the material is ɔnsumed). A variety of problems can be modelled with the help of flow systems: fluid flow through pes, parts through a movable tape, electricity through electrical networks, and information through ɔmmunication systems.

Every directed link can be imagined as a channel (pipe) through which material passes. Every channel has an initial capacity, which represents the maximum capacity that can be transported through that channel. The nodes can be imagined as crossroads (a fusion) of channels, and for all nodes except the source and the basin, the material only passes through them, without remaining in them. In other words, the amount of material that enters the node, must be equal to the amount of material exiting the node. This trait is called "flow conservation" and is equivalent to Kirchhoff's law.

In a general case, when we observe a two-terminal flow network *G(V,E),* we observe a directed graph with two special nodes - a source s and a sink *(s ≠ t),* in which every link $(u, v) \in E$ has a nonnegative capacity $c(u, v) \geq 0$. The function *c* is called capacity function. Shortly, we denote capacity network with *G (V, E, c).*

Flow in *G (V, E, c)* is a function $f: E \to R^{+} \cup \{0\}$ *that* satisfied the following two constraints:

1. *Capacity constrain:* $0 \leq f(u, v) \leq c(u, v)$, for every $(u, v) \in E$, or the flow of the link cannot exceed the capacity of the link.
2. *Flow conservation:* For all $v \in V/\{s, t\}$

$$\sum_{u \in V} f(u,v) = \sum_{w \in V} f(v,w), \tag{1}$$

In other words, the total flow that enters *v* is equal to the total flow that exits *v*, for all $v \in V/\{s, t\}$.

We assume that, if the link *(u, v)* doesn't exist, or $(u, v) \notin E$, then $f(u, v) = 0$.

The value of the flow $|f|$ is defined in the following way:

$$|f| = \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s), \tag{2}$$

or the value of the flow is equal to the value received when from the total flow that exits the source, we deduct the total flow that enters the source. Usually, there are no links that enter the source in flow network, so the flow that enters the source is going to be zero.

$$\sum_{v \in V} f(v,s) = 0,$$

On the other hand, it can be shown that the value of the flow is equal to the value that is received when we deduct the total flow that exits the sink from the total flow that enters the source.

Because flow conservation is valid for all nodes, we have:

$$\sum_{v \in V} \left(\sum_{w \in V} f(v,w) - \sum_{u \in V} f(u,v) \right) = \sum_{v \in V} \sum_{w \in V} f(v,w) - \sum_{v \in V} \sum_{u \in V} f(u,v) = 0,$$

On the other hand we have

$$\sum_{v \in V} \left(\sum_{w \in V} f(v,w) - \sum_{u \in V} f(u,v) \right) = \left(\sum_{w \in V} f(t,w) - \sum_{u \in V} f(u,s) \right) + \left(\sum_{w \in V} f(s,w) - \sum_{u \in V} f(u,t) \right)$$

$$= |f| + \left(\sum_{w \in V} f(t,w) - \sum_{u \in V} f(u,t) \right).$$

From which follows:

$$|f| = \left(\sum_{u \in V} f(u,t) - \sum_{w \in V} f(t,w) \right).$$

A flow f is **feasible** if f $(u, v) \leq c\ (u, v)$, for every $(u, v) \in E$. Most commonly, we will observe only flows that are possible with regard to some fixed function with a capacity c. It is said that the flow f **saturates** the link (u, v) if f $(u, v) = c\ (u, v)$ and destroys (annuls) the link (u, v) if f $(u, v) = 0$.

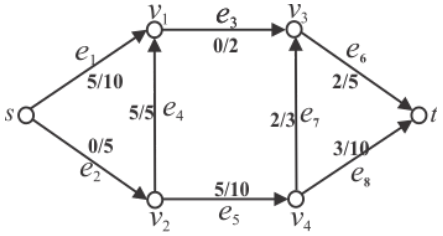


Figure 2 *Flow with a value of 5. Every link is denoted with its flow/capacity*

The flow is **maximal** if there is a maximum possible value between all flows from the source s to the sink t in a given flow network with a given capacity function.

The problem of finding the maximum flow in a given flow network G with a source s and a sink t is to find the flow with the highest possible value, or to find the maximum flow.

2. Ford – Fulkerson Algorithm

The Ford-Fulkerson algorithm is an algorithm with which the maximum flow in a flow network is calculated. In literature, it is frequently referred to as the Ford-Fulkerson method because multiple implementations of this algorithm with a different execution time exist. Before we give the Ford-Fulkerson algorithm, we have to initiate the terms for residual networks, augmenting path and minimum cut [2].

The value of the flow increases iteratively with the Ford-Fulkerson algorithm. The initial value of the flow is taken to be equal to 0, or $f\ (u, v) = 0$, for all $(u, v) \in V$. In every iteration, the value of the flow is increased with finding the augmenting path in the residual network for the flow f, G_f. We need to take into account if the flow of any link in G is going to change (increase or decrease), because with every iteration of the algorithm, the value of the flow of the network is increased. The decrease of flow for some link can be necessary in order to enable the algorithm to send a bigger flow from the source s to the sink t. This action is repeated while augmenting paths exist in the corresponding residual network [2].

Now, let us give the Ford-Fulkerson algorithm.

Algorithm 1 – Ford – Fulkerson Algorithm ($G\ (V, E, c)$)

Step 1. Initializing flow f to 0, f = 0,

Step 2. While an augmenting **path** p exists in the residual G_f, find a flow f through p,

Step 3**.** Return f.

3. Residual network

For a given flow network G and flow f, the residual network consists of links whose capacities show how much the flow of the links in G can change. Additional flow is assigned to the links, which is equal to the value received then the flow that is released in that link is deducted from the capacity of the link. If this value is positive, then it is called residual capacity $c_f(u, v) = c(u, v) - f(u, v)$. In G_f only links from G through which we can release additional flow are found. Otherwise, if the capacity of the link is equal to the flow that has already been released through the link $c_f(u, v) = 0$, these links are also not in G_f. It should be emphasized that there are links in G_f that are not in G. In order to represent the possible decrease of flow, we add links (v, u) in G_f whose residual capacity is $c_f(u, v) = f(u, v)$.

Let's assume that we have a flow network $G(V, E)$, with a source s and a sink t. Let f be a flow in the flow network. The residual capacity is defined in the following way:

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v), & \text{if } (u,v) \in E \\ f(u,v), & \text{if } (u,v) \in E \\ 0, & \text{else} \end{cases}. \tag{3}$$

The residual flow network for the flow network $G(V, E)$ induced by the flow f is $G_f(V, E_f)$ where

$$E_f = \{(u, v) \in E: c_f(u, v) > 0\}. \tag{4}$$

In Figure 3 a) a flow network with a flow f is represented, that has a value of $|f| = 3$. For every link it is denoted how much the flow is, that passes through the link and what the possible capacity of the link (a/b) is. The corresponding residual flow network is represented in Figure 3 b).

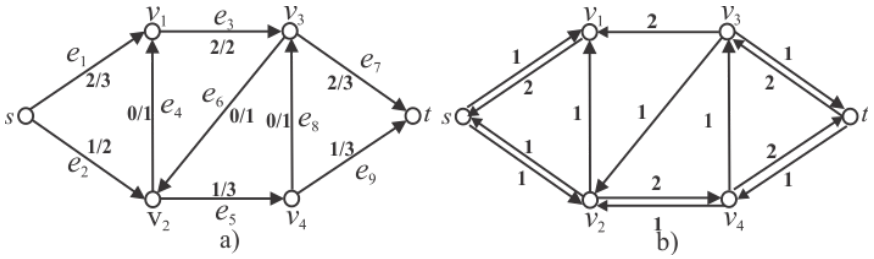


Figure 3 *a) Flow network G (V, E) with flow f, |f| = 3. b) Residual flow network G_f (V, E).*

4. Augmenting path

For a given flow network $G(V, E)$ and flow f, an augmenting path p is a path from s to t in the residual flow network $G_f(V, E)$. According to the definition of a residual flow network, we can increase the flow of the link (u, v) in the augmented path p for $c_f(u, v)$, without damaging the capacity constraint of the link (u, v) and of the link (v, u) in the flow network $G(V, E)$. In Figure 4 the links of the augmented path $p = \langle s, v_2, v_4, t \rangle$ are marked with a dashed line.

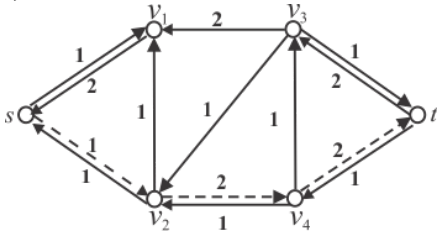


Figure 4 *Residual flow network G_f (V, E) from Figure 3.3 with an augmented path $p = \langle s, v_2, v_4, t \rangle$.*

In order not to violate the capacity constraint, we can increase the flow through every link of the path $p = <s, v_2, v_4, t>$ for 1, because the smallest residual capacity of this path is $c_f(s, v_2) = 1$. The smallest capacity for which we can increase the flow for every link of the path p is called residual capacity of p:

$$c_f(p) = \min\{ c_f(u, v) : (u, v) \text{ is a link in } p\}. \tag{5}$$

4.1. Cut in flow network

With the Ford – Fulkerson Algorithm we do an increase of the flow multiple times, until we find the maximum flow. The question of how we know the algorithm ends arises. The theorem for maximum flow – minimum cut says that the flow is maximal if and only if its residual flow network does not contain additional augmented paths. In order to prove this theorem, it is necessary to introduce the term cut in a flow network.

A cut (V_1, V_2) of a flow network $G(V, E)$ is a division of the set of nodes V into sets V_1 and $V_2 = V / V_1, s \in V_1$ and $t \in V_2$. If f is a flow, then the flow through the cut $f(V_1, V_2)$ is defined in the following way:

$$f(V_1,V_2) = \sum_{u\in V_1}\sum_{v\in V_2} f(u,v) - \sum_{u\in V_1}\sum_{v\in V_2} f(v,u). \tag{6}$$

The capacity of the cut (V_1, V_2) is

$$c(V_1,V_2) = \sum_{u\in V_1}\sum_{v\in V_2} c(u,v). \tag{7}$$

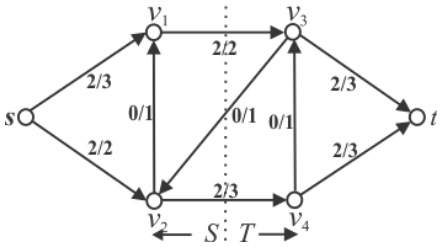


Figure 5 *Cut in the flow network from Figure 3.3 a) where $V_1 = \{s, v_1, v_2\}$ and $V_2 = \{v_3, v_4, t\}$. The flow $f(V_1, V_2) = 4$ and the capacity $c(V_1, V_2) = 4$.*

The minimum cross section in a flow network is a cut whose capacity is smaller than the capacity of all the other cuts in the network.

Lemma 31 Let f be a flow in a flow network G with a source s and a sink t, and let (V_1, V_2) be any cut in the flow system. Then, the flow through the cut $f(V_1, V_2) = |f|$.

The following corollary emanates of Lemma 1

Corollary 1 The value of each flow f in the flow network G is limited from above with the capacity of any cut of G. Hence, the flow in the flow network cannot be larger than the capacity of any cut in the flow network.

The following theorem gives us a way of calculating the maximum flow in a flow network. We have already said that the residual network $G_f(V, E)$ does not contain an augmented path when the flow of the flow network is maximal. From the following theorem, we can conclude that if the value of some flow is equal to the capacity of some cut, then that flow is maximal.

Theorem 1 (Theorem of maximum flow – minimum cut)

If f is a flow in the flow network $G(V, E)$ with a source s and a sink t, then the following conditions are equivalent:

1. f is a maximum flow in G,
2. The residual network G_f doesn't contain an augmented path,
3. $|f| = c(V_1, V_2)$ for any cut (V_1, V_2) of G.

After introducing the terms for residual network, augmented path and residual capacity, we will give a simple expansion of the pseudo-code for the algorithm given above:

Algorithm 2 - Ford – Fulkerson Algorithm ($G(V, E, c)$)

Step 1. For every link $(u, v) \in G, f(u, v) = 0$,
Step 2. While there is an augmented path p from s to t in the residual network G_f,
Step 3. $c_f(p) = \min\{ c_f(u, v) \mid (u, v) \in p \}$
Step 4. For every link $(u, v) \in p$
Step 5. If $(u, v) \in E$

$$f(u, v) = f(u, v) + c_f(p),$$

else

$$f(v, u) = f(v, u) - c_f(p),$$

The complexity of this algorithm is $O(|f^*||E|)$ where $|f^*|$ is the value of the maximum flow in the flow network.

Example 1 Let us find the maximum flow of the flow network given in Figure 3.3 a). Given in Figure 3.3 b) is the residual network for the flow with a value of 3. According to step 3 of the algorithm, the flow of all links of the path p = <s, v_2, v_4, t > is going to be increased by 1 because the links (s, v_2), (v_2, v_4) and (v_4, t) are elements of E.

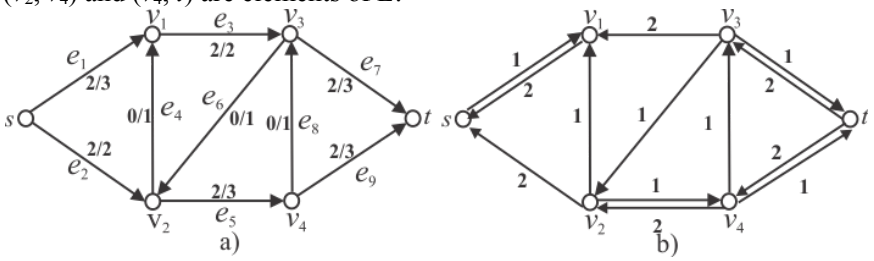


Figure 6 *a) Flow network G (V, E) with flow f, |f| = 4. b) Residual network G_f (V, E)*

We can observe that the residual network shown in Figure 6 b) does not contain an augmented path, from which, according to the theorem for maximum flow – minimum cut, we can conclude that the maximum value of the flow from the flow network in Figure 3 a) is 4.

5. The visualization program of the Ford – Fulkerson algorithm

This program uses the Ford – Fulkerson algorithm on a graph that you are going to construct on the drawing surface.

You draw a node with a left click:

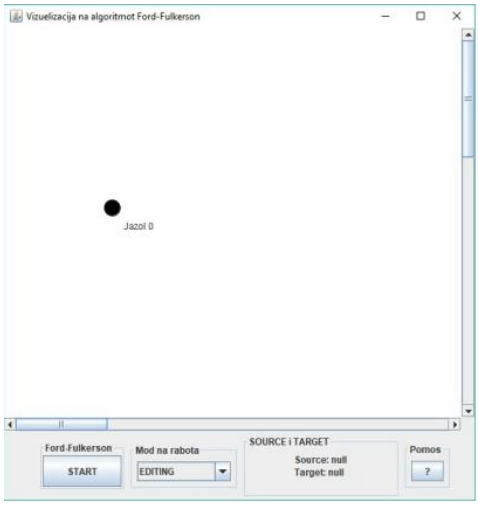


Figure 7 *Drawing of a node*

With a click on the starting node and a pull to the destination node, you draw a link between the two nodes.

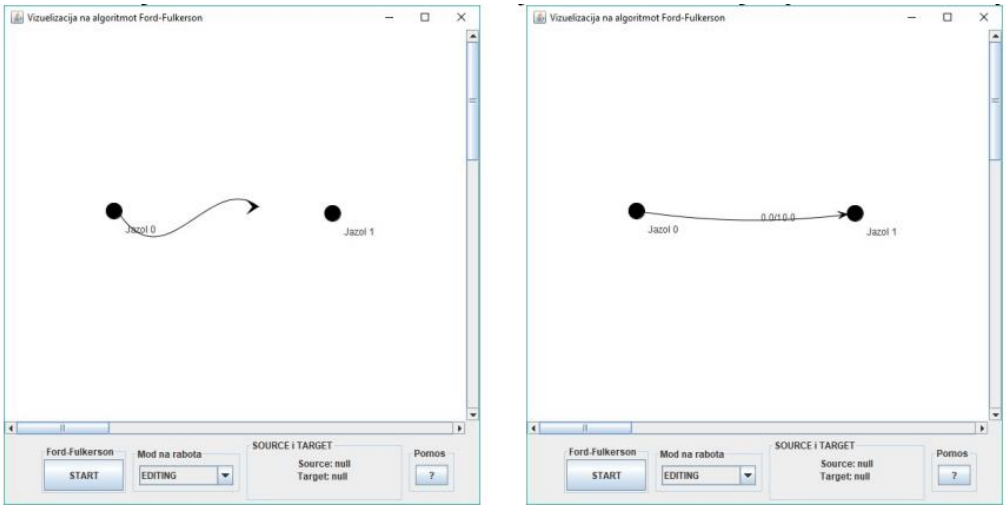


Figure 8 *Adding a link*

A menu appears with pressing the right mouse click:
Through the menu you can:

- delete a node
- assign a node as a source – the node that will be assigned as a source will be colored in green. Only one node can be a source because if the graph already contains a source and you assign a new node to be the source, the previous node will be an ordinary node. You can read which node is assigned as source in the bottom part of the window.
- assign a node as a target (sink) – the node that will be assigned as a target (sink) will be colored yellow. Only one node can be a target because if the graph already contains a target and you assign a new node to be the target, the previous node will be an ordinary node. You can read which node is assigned as target in the bottom part of the window

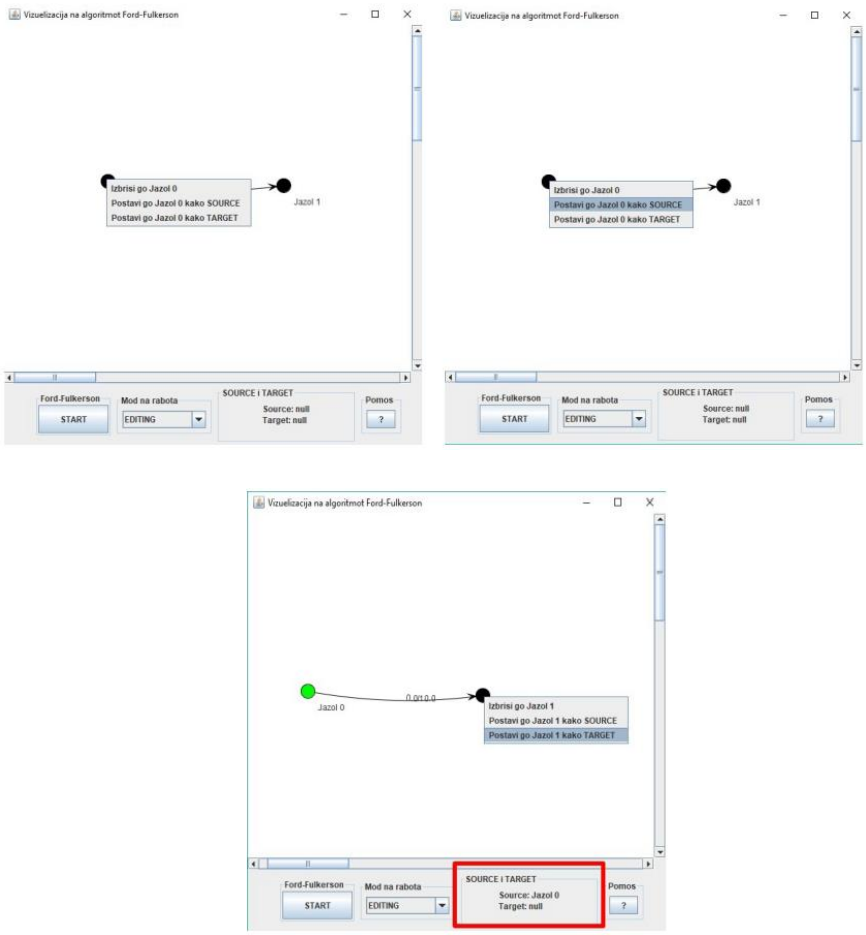


Figure 9 *Menu on right click on node, assigning node as source or target (sink)*

A menu appears with pressing the right mouse click through which you can:

- delete a link

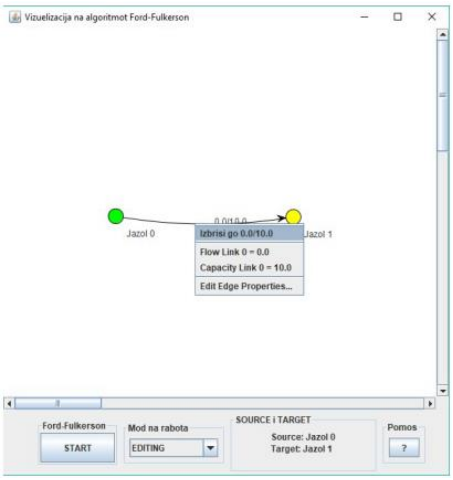


Figure 10 *Menu or right mouse click, deleting link*

- change the flow and capacity of a link

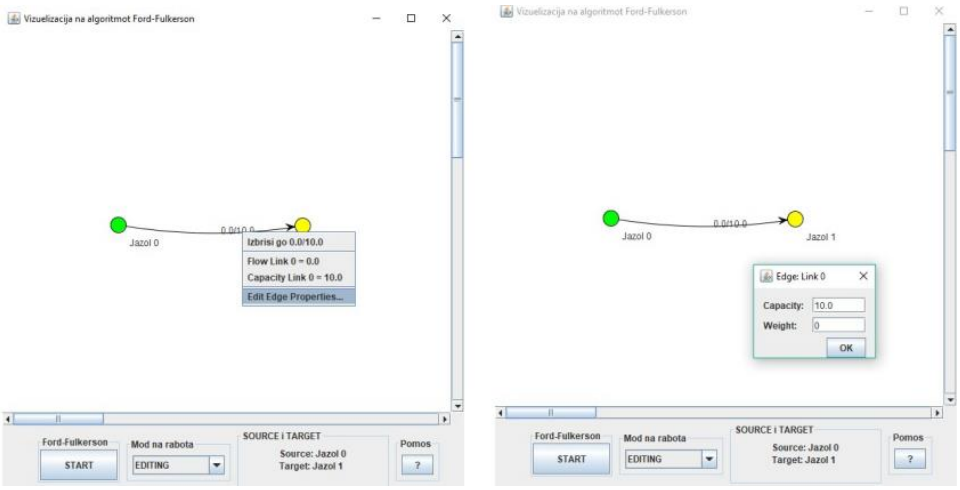


Figure 11 *Changing the capacity and the flow of a link*

You can position the nodes, zoom and move them with a change in work mode. The addition of nodes and links and their editing which is described up to this point is done when “EDITING| is chosen in the dropdown menu. That is the basic work mode in which the program opens when it is started.

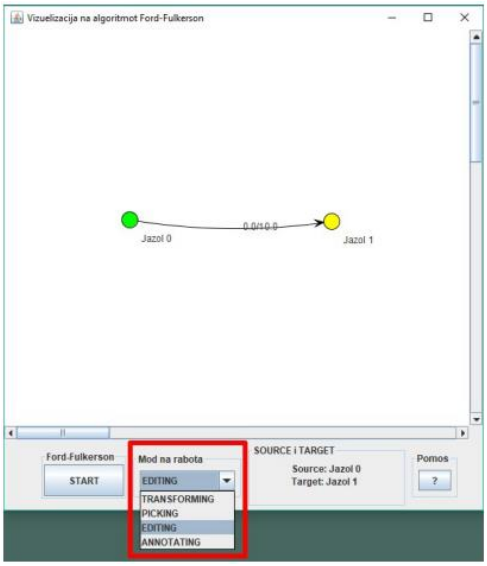
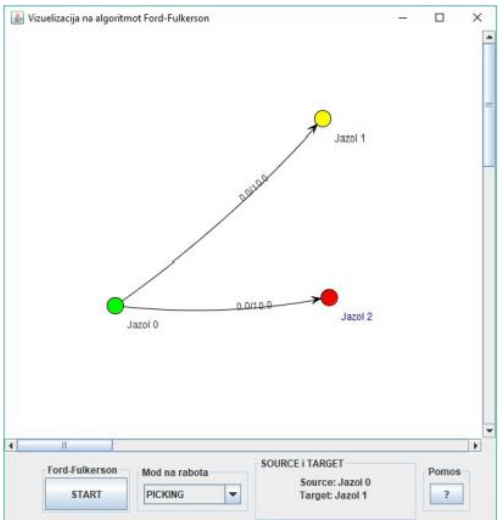


Figure 12 *Changing work mode*

If you select “PICKING” from the dropdown menu, you can move the nodes and links. The selected nodes or links are colored red.



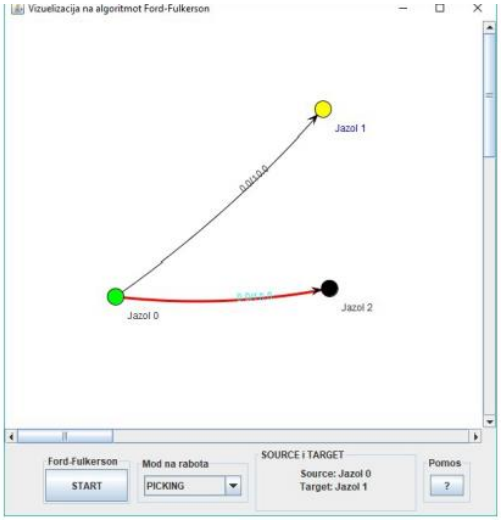


Figure 13 *a) Selected node* *b) Selected link*

You can navigate through the work surface with the sliders on the window. If you select “TRANSFORMING” from the dropdown menu, you can navigate through the work surface with the mouse as well.
You can zoom with the middle button of the mouse in any work mode: you move away by scrolling forward, you move closer to the graph by scrolling backwards.

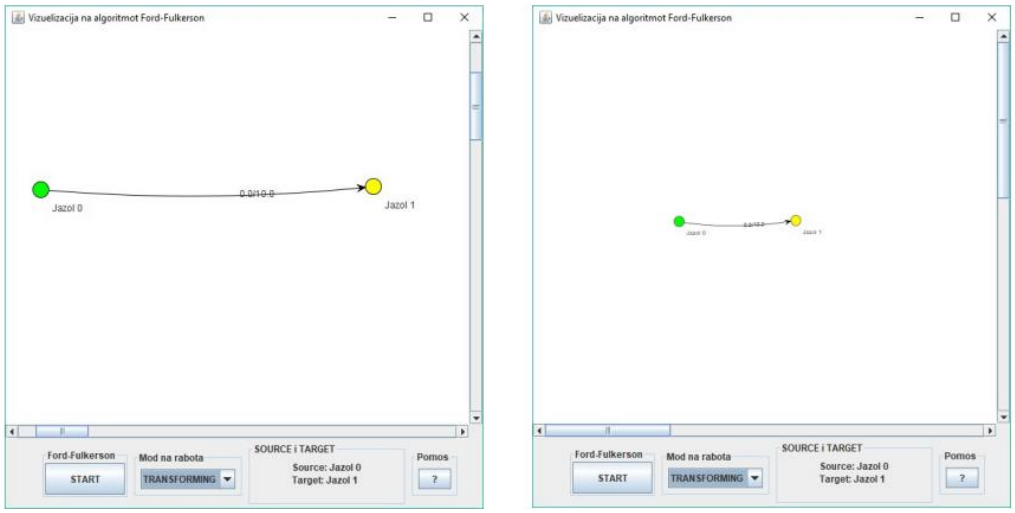
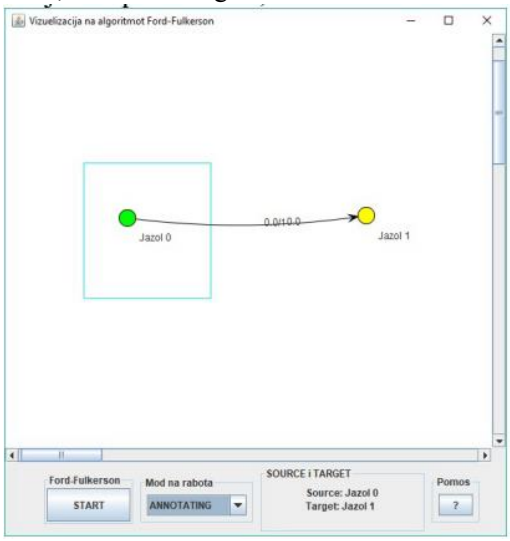


Figure 14 *Zoom*

If you select "ANNOTATING" from the dropdown menu, you can draw rectangles which you can use for marking of some surface. But, these markings cannot be erased.



Click on start to apply the algorithm. When the links are passed through in order to find an augmented path from the source to the target, the links are colored blue. When the algorithm returns back and calculates the residual capacity, the links are colored red and the values of the flow are being updated. After finding the maximum flow, the nodes that are part of the cut of the source, in the minimum cut, are colored white.

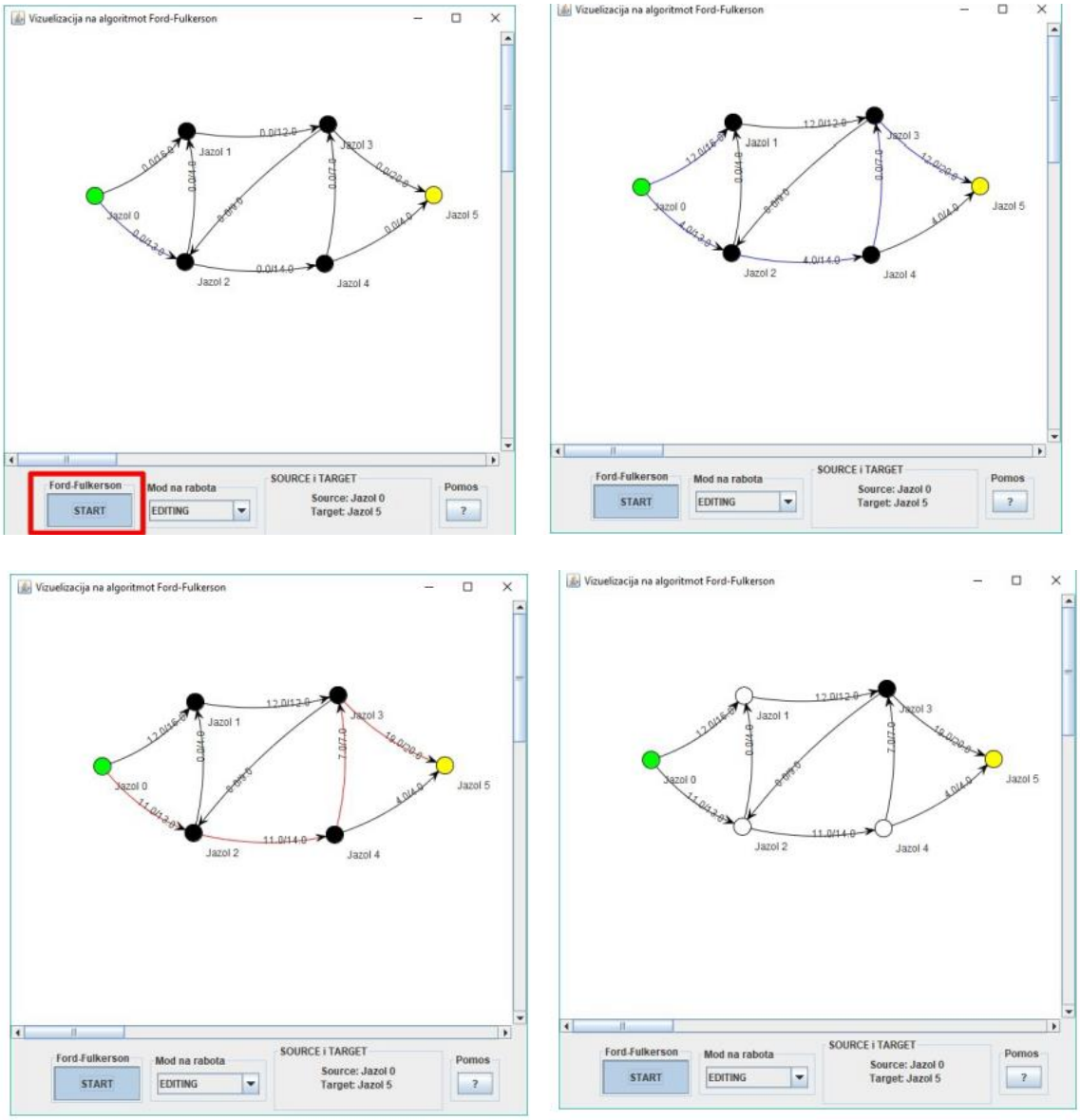


Figure 15 *Applying the algorithm*

With a click on the help button, you receive short instructions on how to use the program.

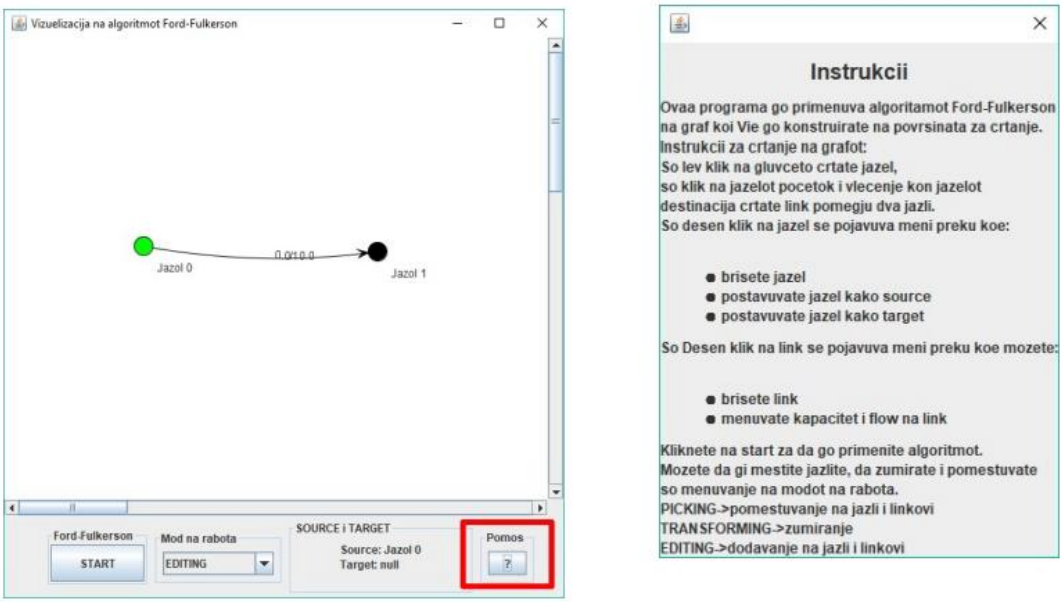


Figure 16 *Instructions for use*

6. Conclusion

The paper i gives a visualization of the algorithm for finding the maximum flow in the flow network. Instructions for drawing, deleting nodes and creating a link between nodes are given. In the program, we can see the changes in the flows in the networks and finding of augmenting paths in the residual network. This is all done for one purpose – for the students to better understand the Ford-Fulkerson algorithm.

References

[1] *A. Kamil*. (2003)" Graph Algorithms,"CS61B, University of California, Berkeley.

[2] *H.T. Cormen, E.C. Leiserson, R.L. Rivest, C. Stein*. (2009) "Introduction to Algorithms", 3rd edition, The MIT Press.

[3] *J. Erickson*. (2009) "Algorithms", University of Illinois at Urbana-Champaign.

[4] J. Schroeder, P.A. Guedes, P. D. (2004)"Elias Computing the Minimum Cut and Maximum Flow of Undirected Graphs,"Relatório Técnico RT-DINF 003/2004, Curitiba, PR.

[5] *L.R. JR. Ford, D.E. Fulkerson*. (1962) "Flows in Networks," Princeton University Press.

[6] *Max-Flow Algorithms*, Read more: https://courses.engr.illinois.edu/cs473/fa2012/notes/22-maxflowalgs.pdf

[7] *JUNG 2.0 Tutorial,* Read more: http://www.grotto-networking.com/JUNG/JUNG2-Tutorial.pdf

[8] jung2 2.0.1 API, Read more: http://jung.sourceforge.net/site/apidocs/overview-summary.html

[9] FordFulkerson.java, Read more: https://algs4.cs.princeton.edu/code/edu/princeton/cs/algs4/FordFulkerson.java.html

[10] [10] *Ford–Fulkerson Algorithm*, Read more: https://www.coursera.org/lecture/algorithms-part2/ford-fulkerson-algorithm-0pI7R

[11] *Network Flow*, Read more: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-design-and-analysis-of-algorithms-spring-2012/lecture-notes/MIT6_046JS12_lec13.pdf

[12] *V. Hadzilacos*, (2017) "Choosing augmenting paths in the Ford-Fulkerson algorithm", Read more: http://www.cs.toronto.edu/~vassos/teaching/c73/handouts/FF-augmenting-path-choice.pdf

Natasha Stojkovikj
Goce Delcev University,
Faculty of computer science
Bul.Goce Delcev 89, 2000 Stip Macedonia
E-mail address: natasa.maksimova@ugd.edu.mk

Mirjana Kocaleva
Goce Delcev University,
Faculty of computer science
Bul.Goce Delcev 89, 2000 Stip Macedonia
E-mail address: mirjana.kocaleva@ugd.edu.mk

Aleksandra Stojanova
Goce Delcev University,
Faculty of computer science
Bul.Goce Delcev 89, 2000 Stip Macedonia
E-mail address: aleksandra.stojanova@ugd.edu.mk

Isidora Janeva
Goce Delcev University,
Faculty of computer science
Bul.Goce Delcev 89, 2000 Stip Macedonia
E-mail address: isidora.102362@student.ugd.edu.mk

Biljana Zlatanovska
Goce Delcev University,
Faculty of computer science
Bul.Goce Delcev 89, 2000 Stip Macedonia
E-mail address: biljana.zlatanovska@ugd.edu.mk