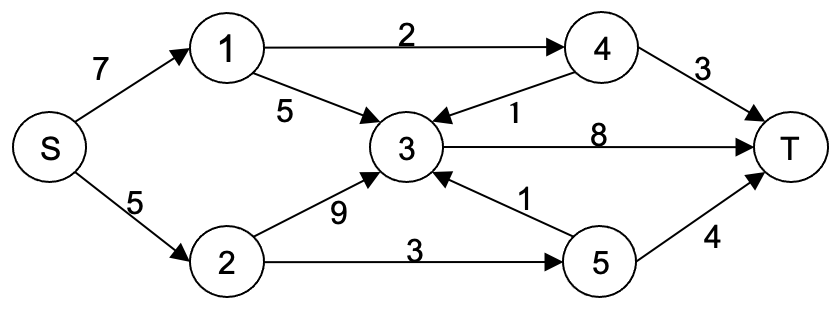
Lab Work 09 Report:  
Graph Algorithms in Computer Networks.

|  |  |  |
| --- | --- | --- |
| **Student Name Surname** | **Student ID** | **Date** |
| Li Yurij Janovich 2 |  |  |

4.1. Create your variant of the tasks

**4.1.1. Determine your topology variant:**

Li **Y**urij the first letter in the name is numbered 25; (25 MOD 10) +1 = 5 + 1 = 6th topology variant.

**4.1.2. Router names**.

For (LI YURIJ JANOVICH 2)🡺(LIYURJANOVCH) we get the following names for routers: L(s), I(1), Y(2), U(3), R(4), J(5), A(t).

**4.1.3. The source s-router** for the given network topology will be the L=s router. It will also be the root router for building the spanning tree and the source for calculating the maximum flow, the sink t-router will be router A=t.

**4.1.4. The weight of edges** (links between routers) corresponds to the modulus of the difference between the alphabetical numbers of the letters of the corresponding vertices (routers).

Сommunication line (edges) weight for LIYURJA router network topology:

L🡪I = |12-9| = 3

L🡪Y = |12-25| = 13

I🡪R = |9-18| = 9

I🡪U = |9-21| = 12

Y🡪U = |25-21| = 4

Y🡪J = |25-10| = 15

U🡪A = |21-1| = 20

R🡪U = |18-21| = 3

R🡪A = |18-1| = 17

J🡪U = |10-21| = 11

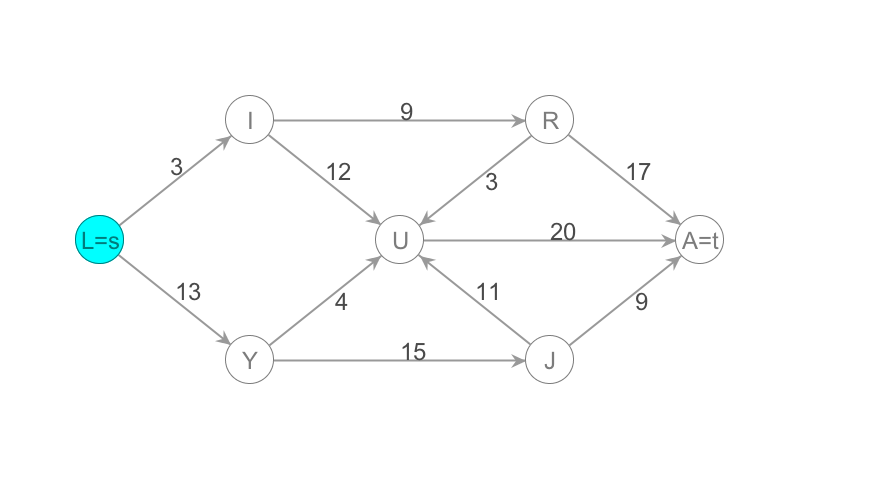
J🡪A = |10-1| = 9

4.2. Adjacency matrix of Final Graph.

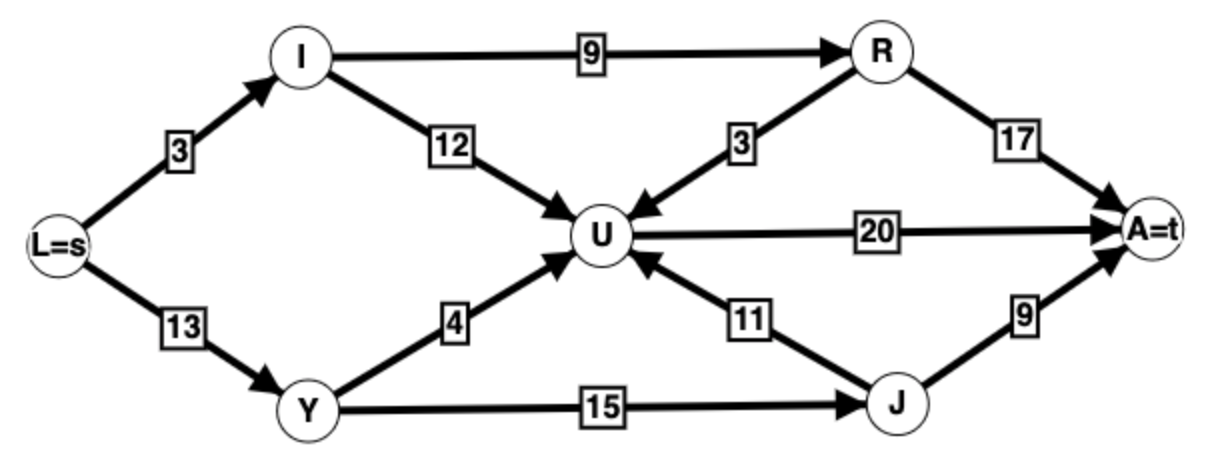
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Vertices** | **L=s** | **I** | **Y** | **U** | **R** | **J** | **A=t** |
| **L=s** | 0 | 3 | 13 |  |  |  |  |
| **I** |  | 0 |  | 12 | 9 |  |  |
| **Y** |  |  | 0 | 4 |  | 15 |  |
| **U** |  |  |  | 0 |  |  | 20 |
| **R** |  |  |  | 3 | 0 |  | 17 |
| **J** |  |  |  | 11 |  | 0 | 9 |
| **A=t** |  |  |  |  |  |  | 0 |

4.3. Final Graph in GraphTea/GraphOnline.

**4.3.1. GraphTea.**

****

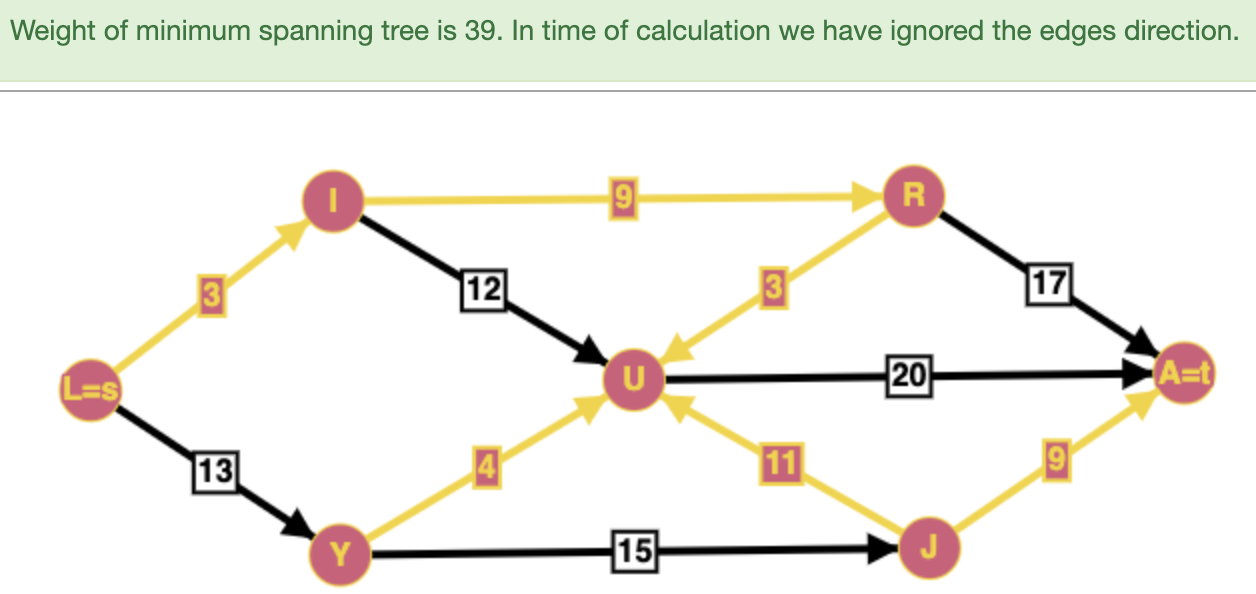
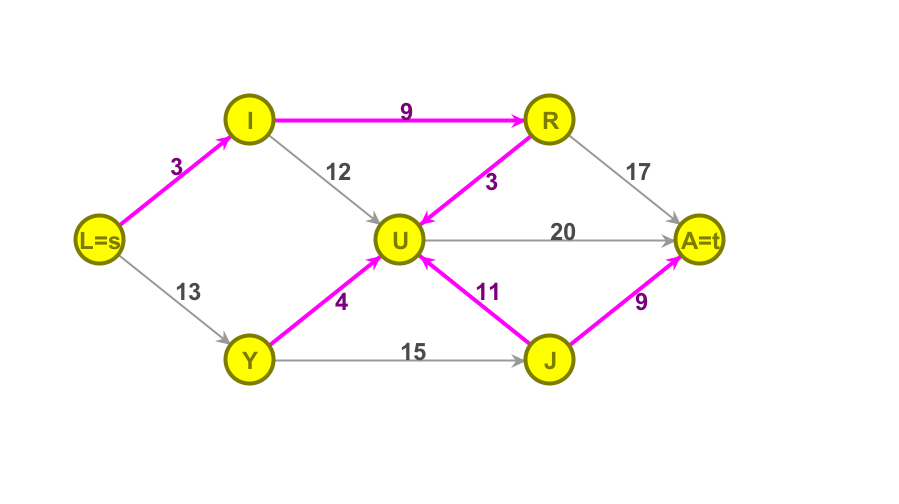
**4.3.1. GraphOnline.**



**GrapOnline** link: <https://graphonline.ru/en/home?graph=fekRTJntTGMZRcpBZZcst>

4.4. Building Min Spanning Tree in GraphTea/GraphOnline.

MinST(L) = L-I-R-U-Y,U-J-A; MinST(L) Cost = 3+9+3+4+11+9 = 39.

****

4.5. Manually Building Min Spanning Tree.

**4.5.1. Prima Algortihm.**

Select any vertex and sequentially search for the next vertex with min weight without cycles.

**MinST(A)** = A-J-U-R,U-Y,R-I-L; **MinST(A) Cost** = 9 + 11 + 3 + 4 + 9 + 3 = 39

**4.5.2. Kruskal Algorithm.**

Select the min edge and sequentially add the next edge with min weight without cycles.

**MinST** = I-L, R-U, U-Y, A-J, I-R, J-U; **MinST Cost** = 3 + 3 + 4 + 9 + 9 + 11 = 39

**Comparison of results.** Differences between MinST in 4.5.1., 4.5.2. and 4.4. is absent.

4.6. Building SPF Tree by the Dijkstra algorithm manually.

Dijkstra algorithm step-by-step table.

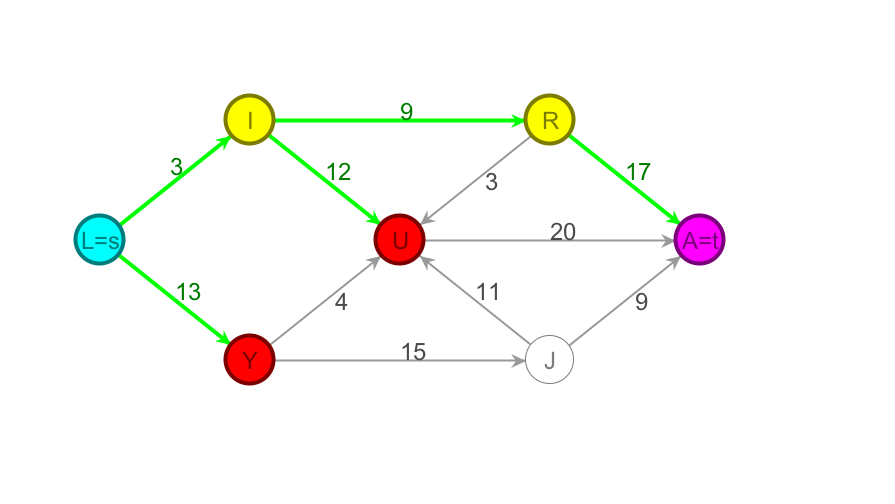
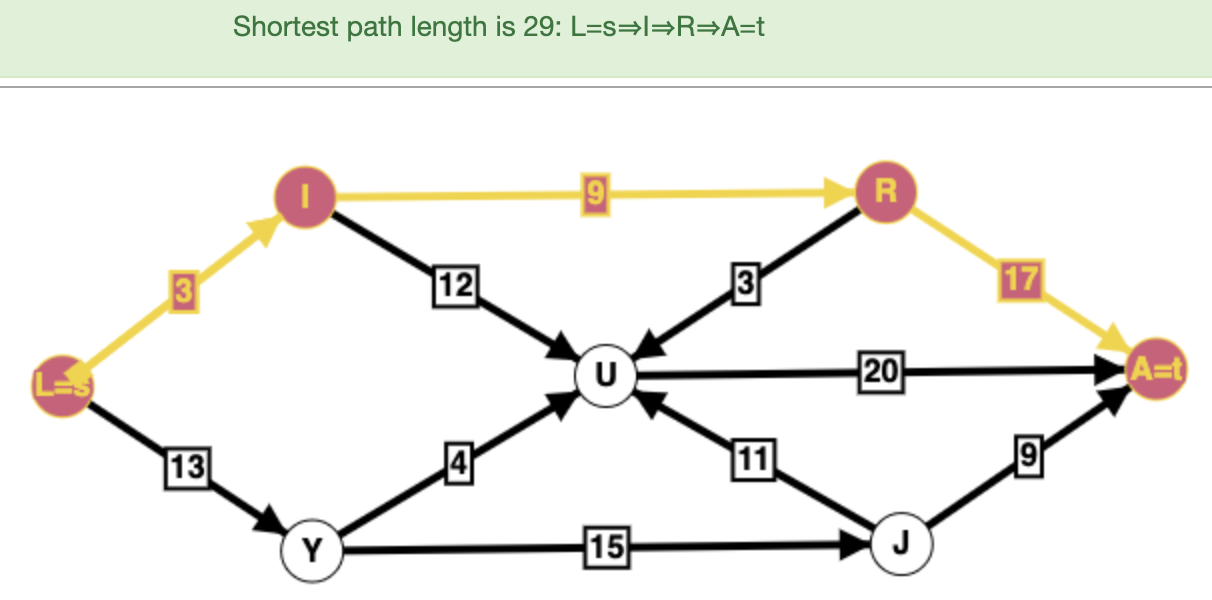
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Step** | **The set of visited vertices** | **Cost labels and the path of connection of s-vertex with other vertices** | | | | | | |
| **L=s** | I | **Y** | **U** | **R** | **J** | **A=t** |
| 0 | {L} | (0-LL) | 3-LI | 13-LY | ∞ | ∞ | ∞ | ∞ |
| 1 | {L,I} |  | (3-LI) | 13-LY | 15-LIU | 12-LIR | ∞ | ∞ |
| 2 | {L,I,R} |  | 3-LI | 13-LY | 15-LIU | (12-LIR) | ∞ | 29-LIRA |
| 3 | {L,I,R,Y} |  | 3-LI | (13-LY) | 15-LIU | 12-LIR | 28-LYJ | 29-LIRA |
| 4 | {L,I,R,Y,U} |  | 3-LI | 13-LY | (15-LIU) | 12-LIR | 28-LYJ | 29-LIRA |
| 5 | {L,I,R,Y,U,J} |  | 3-LI | 13-LY | 15-LIU | 12-LIR | (28-LYJ) | 29-LIRA |
| **6** | **{L,I,R,Y,U,J,A}** |  | **3-LI** | **13-LY** | **15-LIU** | **12-LIR** | **28-LYJ** | **(29-LIRA)** |

Note (The vertex is finally included in the set of visited and the min path to it is determined).

The shortest paths to the vertices are given in the last line of the table.

4.7. Building SPF Dijkstra in GraphTea/GraphOnline.

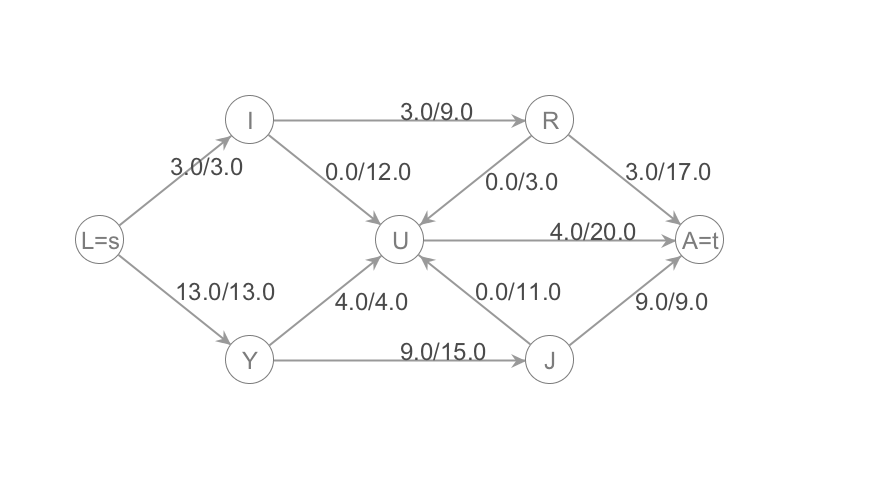
SPF(L🡪A) Cost =LIRA=3+9+17=29.

** **

**Comparison of results.** Differences between SPF Trees in 4.6. and 4.7. is absent.

4.8. Building MaxFlow in GraphTea/GraphOnline.

MaxFlow(L🡪A) = 16.

**** 