INPRESSCO
International Press Corporation

International Journal of Current Engineering and Technology
ISSN 2277 - 4106
© 2013 INPRESSCO. All Rights Reserved.
Available at http://inpressco.com/category/ijcet

Research Article

Application of Graph Theory to find Optimal Paths for the Transportation Problem

Ramë Likaj[a], Ahmet Shala[a*] and Mirlind Bruqi[a]

[a]Faculty of Mechanical Engineering, University of Prishtina, Prishtina, Kosovo

Accepted 10 August 2013, Available online 20August 2013 **Vol.3, No.3 (August 2013)**

Abstract

Graph theory is used for finding communities in networks. Graphs are used as device for modeling and description of real world network systems such are: transport, water, electricity, internet, work operations schemes in the process of production, construction, etc. Although the content of these schemes differ among themselves, but they have also common features and reflect certain items that are in the relation between each other. So in the scheme of transport network might be considered manufacturing centers, and roads and rail links connected directly to those centers. In this paper is designed the solution for an practical problem to find a Minimum Spanning Tree by using Kruskal algorithm and graph search Dijkstra's Algorithm to find the shortest path between two points, Also, for this case was developed a network model of the transportation problem which is analyzed in detail to minimize shipment costs.

Keywords: *Graph, Transport, Algorithm, Minimum Spanning Tree, Node, Arcs*

1. Introduction to Graph Theory

Graph theory provides many useful applications in Operations Research. A graph is defined as a finite number of points (known as nodes or vertices) connected by lines (known as edges or arcs). In this paper for a given graph find a minimum cost to find the shortest path between two points.

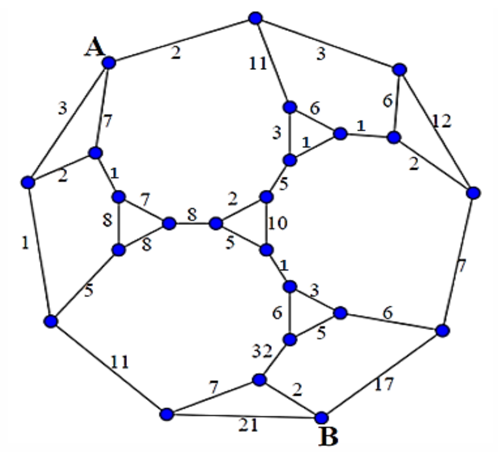

Figure 1 Connected Graph

There are different path options to reach from node A to node B, but our aim is to find the shortest path with a minimum transportation costs, this requires a lot efforts.

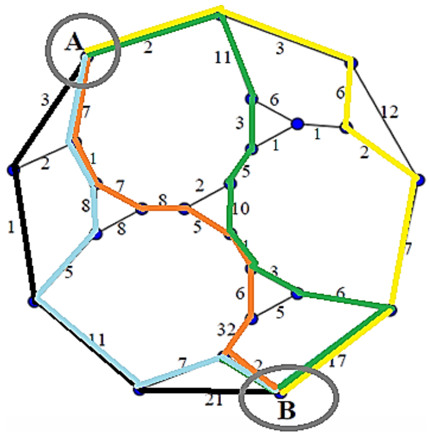

Figure 2 Some of the path options

2. Minimum spanning tree by using Kruskal Algorithm

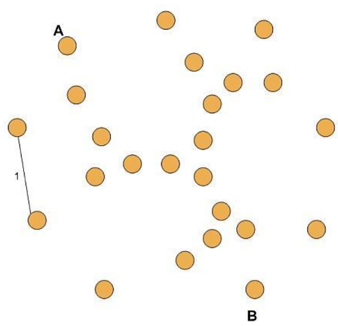

Figure 3

*Coresponding author: Ahmet Shala

In this paper the Minimum Spanning tree for the given case is described by several figures given in the following. Firstly are used all nodes of the given graph without arches, then we will start to put arcs in their place starting from the lowest cost (arc length 1) to the one with higher costs, but having in mind not to create cycles (Figure 3). This process continues by placing the second arc of length 2 (Figure 4).

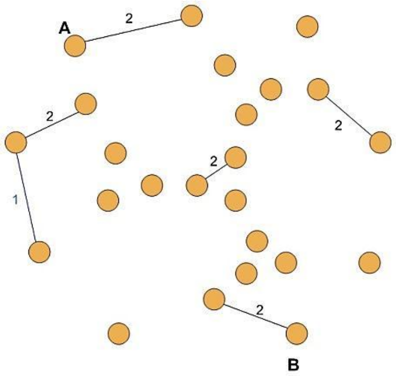

Figure 4

Arch of lower cost that comes after him with units 1 and 2 is the arc of length 3. Again we have processed in the same way having in mind that we should not create cycles (Figure 5).

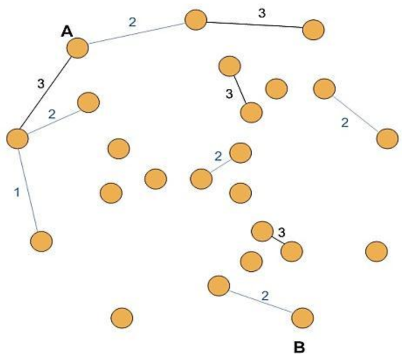

Figure 5

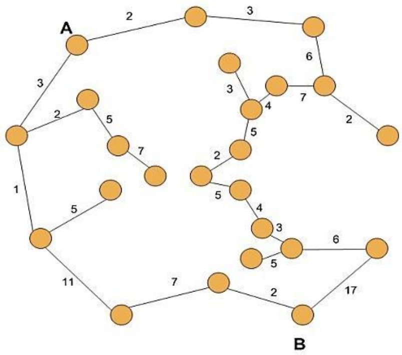

Figure 6

Applying this rule to all arches of the given Graph given, we have gained a minimum Spanning tree which is given in Figure 7. Arches which are removed from the graph are denoted by red color, this happened because, because their deployment create cycles Figure 7.

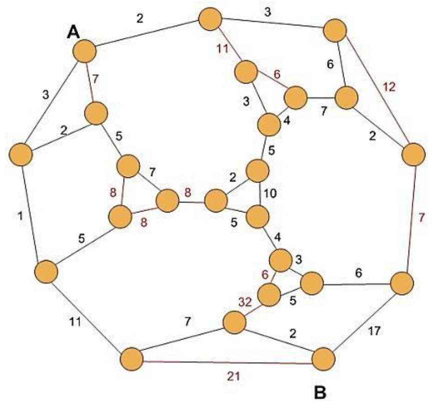

Figure 7

3. Minimum cost path

From the Minimum Spanning Tree shown in Figure 6 we are able to find the minimum cost path (trajectory) from node A to node B. As we can see from the Figure 7, there are two alternative ways to reach from node A to node B, which are distinguished by dash line.

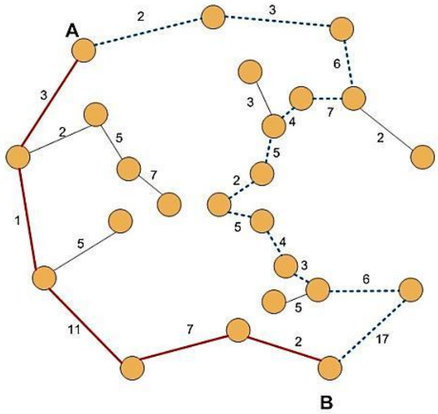

Figure 8

Let's start with first option to calculate the distance from node A to node B (dash line), the result is as follows:
$\mu = 2+3+6+7+4+5+2+5+4+3+6+17 = 64$ units (this is the most expensive path)
For the second option (full line):
$\mu = 3+1+11+7+2 = 24$ units
This means that the second option represents the minimum cost path from node A to node B.

4. DIJKSTRA's Algorithm

By using Dijkstra's Algorithm we are able to find the shortest distances (length of arc) from a node to all other nodes. Firstly, we start from the node A, which is chosen as permanent node. Analyzing the distances of the neighborhoods nodes of the node A, we are able to find the shortest path to node 2 (its distance is equal with 2). Afterwards node 2 is chosen as permanent node, and we have to check after the distances from node 2 to the neighbor nodes. To the each neighbor node is added the length of the permanent node

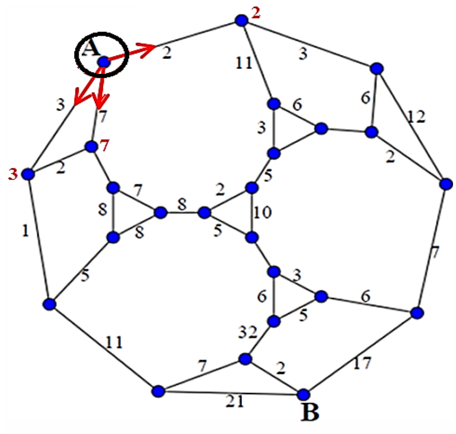

Figure 9 Minimum path

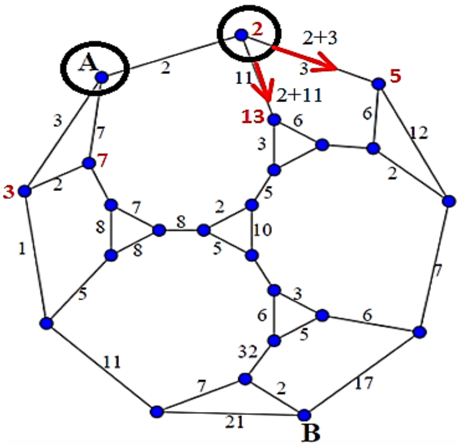

Figure 10 Minimum path

Now is chosen the minimum distance from node A. Minimum distance is chosen as permanent node, since the 3+2 distance is shorter than 7, this means that distance 7 is not going to be considered anymore and we have to use the distance 5.

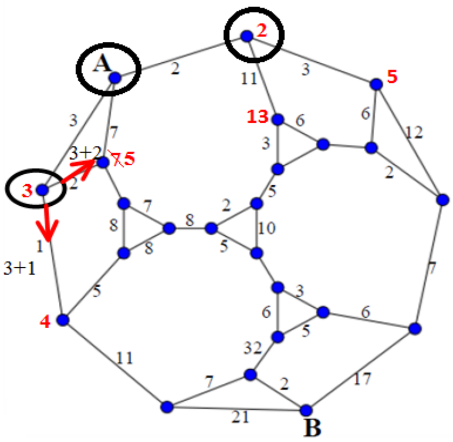

Figure 11 Minimum path

Now is chosen the minimum distance from node A. For this case the permanent node is chosen the minimum distance 4, this means that to all neighbor node is added the distance of permanent node.

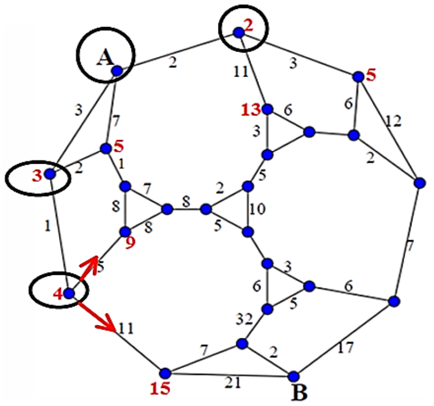

Figure 12 Minimum path

This process is repeated for each node respectively.

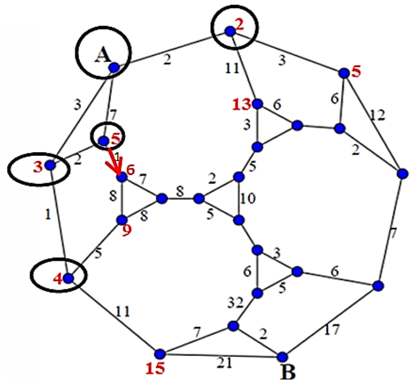

Figure 13

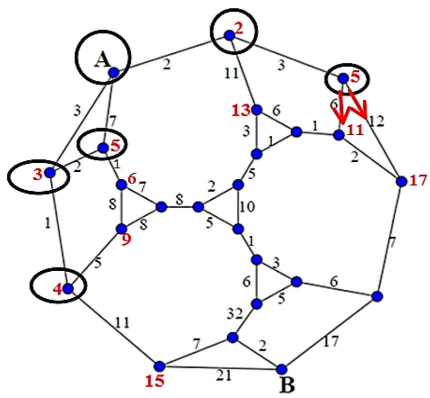

Figure 14

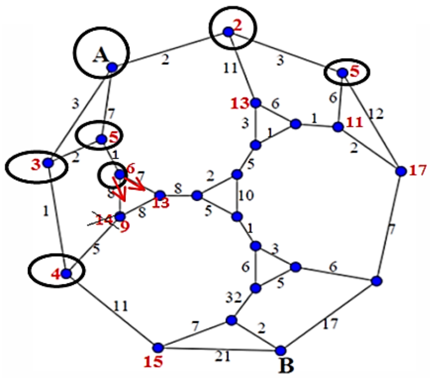

Figure 15

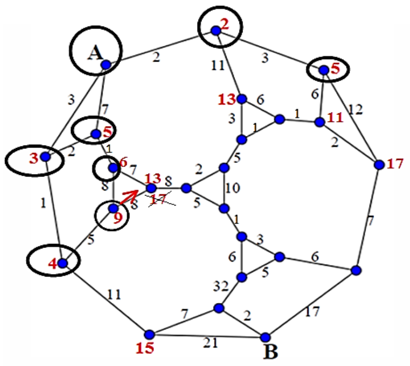

Figure 16

For example, for the permanent node 6 by adding distances 6+8=14, is shown that 14 > 9, this means that the previous distance 9 remains, while the distance 14 is not considered anymore. This means that node 9 is chosen as permanent node and the procedure is similar to the previous cases.

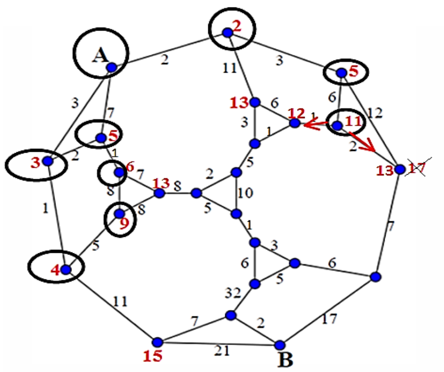

Figure 17

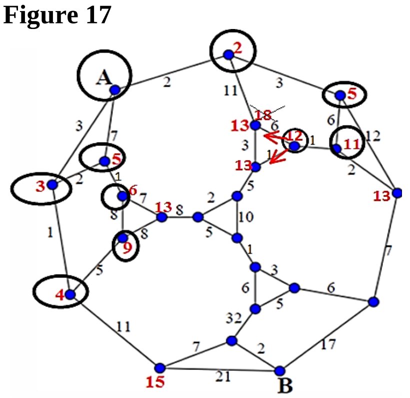

Figure 18

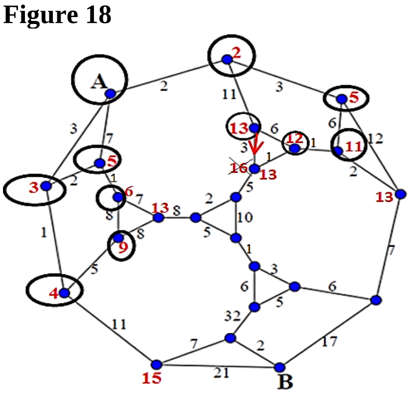

Figure 19

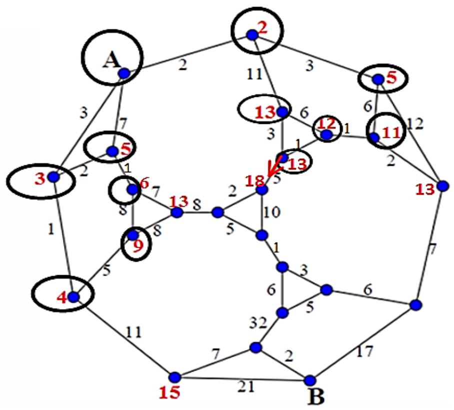

Figure 20

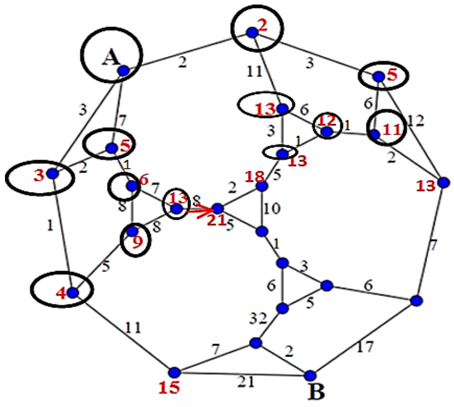

Figure 21

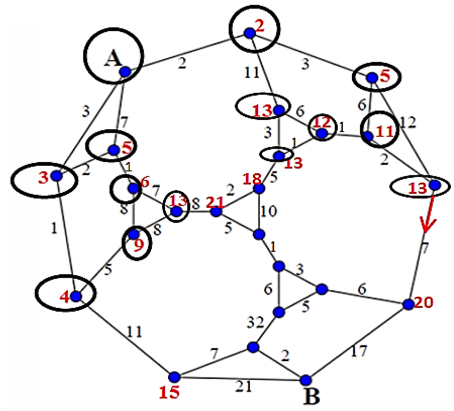

Figure 22

5. Minimum path between nodes A and B

Let's find the shortest path from node A to node B; this is done starting from the node B, by substituting from this node the distance for each neighbor node.

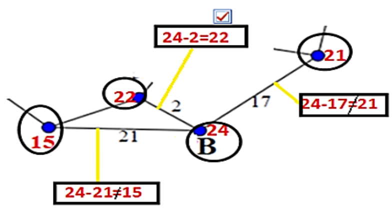

Figure 23

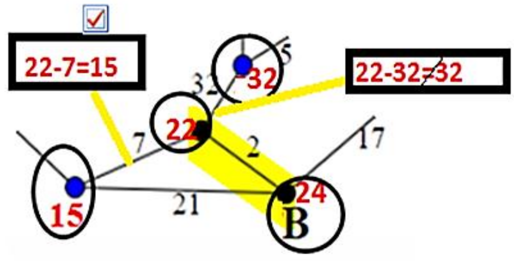

Figure 24

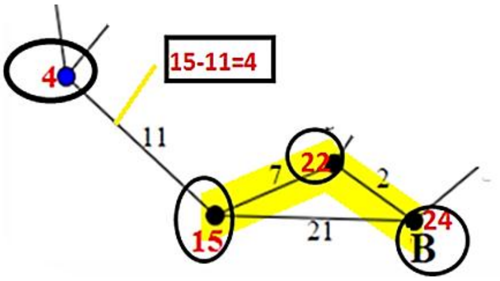

Figure 25

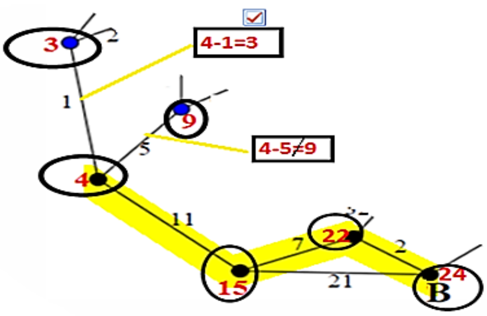

Figure 26

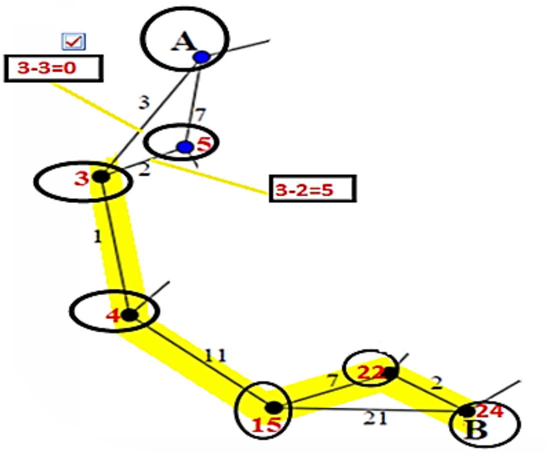

Figure 27

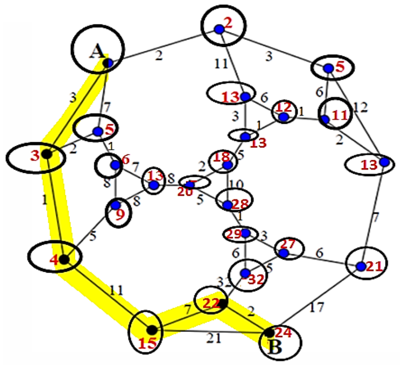

Figure 28 Minimum cost path from node A to node B

Conclusions

Dijkstra's Algorithm will find the shortest path between two nodes/vertices. Kruskal's algorithm will find the minimum spanning tree connecting all the given vertices. Basically, Dijkstra's will find a connection between two vertices, while Kruskal's will find a connection between and number of vertices.

The results which are obtained for the given example shows that Dijkstra’s Algorithm is very effective tool to find the path with lowest cost from node A to node B. Same results have been obtained also for Minimum Spanning Tree by using Kruskal algorithm, but this case the procedure is much simpler with a minimum spanning tree to reach node B from node A with the lowest total cost. We have tried also to find the worst scenario to reach node B from node A which is approximately 63% more expensive from the first case.

References

Likaj, R. (2009), The problem of transport in designing the production systems, pages 175-180, *MITIP*, Bergamo, Italy.

Likaj, R. (2010), Sensitivity analysis in designing the production systems and economic interpretation, MOTSTP, *International Scientific Conference Management of Technology*, Zagreb Croatia.

Chamero, Juan. (2006), Dijkstra’s Algorithm As a Dynamic Programming strategy, *www.intag.org*

Rutter, Sh., (2009), Dijkstra Algorithm final project EDUC 528 from http://shawnrutter.com/pdf/Dijkstra_Algo.pdf

Lee, D.C. (2006), Proof of a modified Dijkstra's algorithm for computing shortest bundle delay in networks with deterministically time-varying links, pages 734 – 736, *Communications Letters, IEEE*

Wen-Chih Chang, Yan-Da Chiu, Mao-Fan Li, (2008), Learning Kruskal's Algorithm, Prim's Algorithm and Dijkstra's Algorithm by Board Game, Pages 275 – 284, *Springer-Verlag* Berlin, Heidelberg