The Use of Weighted Adjacency Matrix for Searching Optimal Ship Transportation Routes

Uporaba ponderirane matrice susjedstva za traženje optimalnih brodskih prometnih ruta

RETRACTED

Karel Antoš
Institute of Technology and Business
Department of Informatics and Natural Sciences
České Budějovice , Czech Republic
e-mail: antos.vste@seznam.cz

DOI 10.17818/NM/2018/[illegible]
UDK 519.17:65[illegible]1
Preliminary communication / *Prethodno priopćenje*
Paper accepted / *Rukopis primljen:* 24. 4. 2017.

Summary

This article provides a new approach to searching solutions of water transport optimization problems. It brings a new tool of the graph theory which [illegible] the Weighted Adjacency Matrix. This Weighted Adjacency Matrix is suitable for searching for the Minimum Spanning Tree (MST) of the graph. It describes the Weighted Adjacency Matrix as a new element, and shows how it could be used in cases where weighted edges of the graph are given. This creates a new procedure of searching the MST of the graph and completes previously known algorithms of searching for the MST. [illegible] the field of ship transportation it could be succesfully [illegible] for solutions [illegible]ing transportation routes where lowest transport costs are needed. Proposed Weighted Adjacency Matrix could be used in similar issues in the field of [illegible] theory, where graphs with weighted edges are given. The procedure is shown in the given example. The paper discusses the application of [illegible] route optimization technique forthe maritime sector.

KEY WORDS
graph theory
minimum spanning tree
Joseph Kruskal
reverse algorithm

Sažetak

U radu se prikazuje novi pristup traženju rješenja optimizacije brodskog transporta. Predstavljen je novi alat teorije grafa, osnosno ponderirana matrica susjedstva. Matrica je prikladna za traženje minimalnog razgranatog stabla (MST) na grafu. Ponderirana matrica susjedstva prikazana je kao novi element i opisano je kako se njomemože koristiti u situacijama kada su zadani ponderirani [illegible]ubovi grafa. Ovim se stvara novi postupak pretraživanja MST grafa i d[illegible]šavaju se prethodni algoritmi pretraživanja MST-a. Ovim postupkom bi se moglo uspješno koristiti u brodskom prijevozu za optimizaciju isplativih transport[illegible]ta. Pre[illegible]ženom ponderiranom matricom susjedstva može se koristiti i za druge slučajeve teorije grafa kada su zadani ponderirani bridovi. Postupak je prikazan na [illegible]loženom primjeru. U [illegible] se raspravlja o primjeni takve tehnike optimizacije u pomo[illegible]om s[illegible]

KLJUČNE RIJEČI
teorija grafa
minimalno razgranato stablo
Joseph Kruskal
reverzni algoritam

1. INTRODUCTION / *Uvod*

One of the important aims in the field of shipping traffic is to find the ideal combination of shipping traffic routes so as to ensure the serviceability of all places, and to reduce the costs of transport connections to the lowest level. It is necessary to reach each hub and to reduce transport costs to the minimum. Hubs are ports and transport routes are shipping lanes.

Graph theory offers useful tools for solving problems in this area. To model this situation we created a connected weighted graph where vertices represent sea ports and the edges represent the routes between the ports through which ships transport goods. The weight of an edge between two vertices represents the energy consumed to conduct the boat between these ports.

At the beginning there was a situation where ships transported goods between hubs over many different routes and in different ways, but the transport links were inefficient and expensive as a whole.

The task of our algorithm is to optimize the connections between hubs, so that the cost of transport links between all ports would be minimal,given that every port is reachable through traffic routes.In this approach, transport cost functions are linear, depending only on distances and they are not in relation to the amount of cargo on the ship, capacity limitation of the ship, loading/unloading expenses in ports etc.

To search for optimal transport connection we can use the tool spanning tree from the graph theory. This tool is useful to optimalize the connections between all hubs to be as simple as possible. Another tool is the minimum spanning tree, which ensures that this unique connection will be the least expensive. To search the minimum spanning tree we offer a new algorithm, which complements the previously known algorithms and demonstrates new and original approach.

2. DESCRIPTION OF THE MST ISSUES / *Opis minimalno razgranatog stabla (MST)*

All graphs in this article are finite, simple and connected. We can transform the system of shipping traffic routes into the the graph where vertices represent sea ports, edges represent transport routes and weights of edges represent the energy consumed to drive the boat between two ports. To model this situation we create a connected graph $G = (V, E)$ with weighted edges. The optimal traffic connection of the system is represented by the spanning tree of the graph. And the problem of the cheapest traffic system means that we must find the minimum spanning tree.

The spanning tree of a connected graph G is a subgraph $G´$ which connects all vertices and which does not contain any cycles [3]. The minimum spanning tree we denote $T = (V, E´)$, where $V´= V$ and $E´$ is the set of $n - 1$ edges of the minimum spanning tree, and it applies that $E´\subseteq E$. In the subsequent text we use the abbreviation MST (short for the Minimum Spanning Tree) [6]. The sum of the weights of edges of MST is minimal.

For searching for the minimum spanning tree there are several obviously known algorithms which search for the MST in different ways. For example The Kruskal's algorithm, Prim's algorithm or Borůvka's algorithm are the generally known. In the article we use some principles of Prim´s algorithm for searching the MST [4]. But this article presents a new procedure for searching for the MST, which is Weighted Adjacency Matrix.

Let $G = (V, E)$ be a connected, finite and non-oriented graph with positively weighted edges, where V is a set of n vertices and E is the set of m edges. The set of vertices V we denote $V = \{v_1, v_2,\ldots,v_n\}$ and the set of edges E denote E where e_{ij} denotes the edge between vertices v_i and v_j, then it is $e_{ij} = \{v_i, v_j\} \in E$. $W(e_{ij})$ denotes the weight of the edge connecting vertices v_i and v_j, where $e_{ij} = \{v_i, v_j\} \in E$.

The spanning tree of a connected graph G is a subgraph $G´$ which connects all vertices and which does not contain any cycles [5]. For this subgraph $G´$ it holds that $G´= (V´, E´)$, where $V´= V$ and $E´\subseteq E$. Note that the set $E´$ contains $n-1$ edges [2].

For subgraph $G´= (V´, E´)$ of the graph G we put $w(G´) = \Sigma_{e\in E´} w(e)$. Because of the spanning tree is a tree we denote it

The spanning tree $T = (V, E_1)$ of the graph G we call the minimum spanning tree if for each spanning tree $T_2 = (V, E_2)$ of the graph G it holds that

$$w(T_1) \leq w(T_2)$$

The minimum spanning tree we denote $T = (V, E_1)$, where $V´ = V$ and $E´$ is the set of $n - 1$ edges of the minimum spanning tree, and it applies that $E´\subseteq E$. In the subsequent text we use the abbreviation MST (short for the Minimum Spanning Tree).

If we define the function $w: E \rightarrow R$ (ie. evaluation of edges), then the minimum spanning tree is such a spanning tree {\displaystyle ({\mathit {V}},{\mathit {E}}´)}for which it holds that the sum of the weights of edges of MST is minimal, i.e. $w(E´) = \Sigma_{e\in E´} w(e)$ is minimal.

In the following capture a new algorithm is displayed which uses some new elements for searching the MST and adapts them to one of the previously mentioned, to the Prim's algorithm.

3. WEIGHTED ADJACENCY MATRIX / *Ponderirana matrica susjedstva*

At first, in the proposed algorithm we create a modified adjacency matrix, which we call " Weighted Adjacency Matrix". This matrix is similar to the Adjacency Matrix where in positions of elements of the matrix are either 1 or 0 if there is an edge between vertices v_i and v_j or not. In this modified Weighted Adjacency Matrix the positive number w_{ij} on the position of the element v_i and v_j indicates the weight of the edge connecting vertices v_i and v_j, if the edge between vertices v_i and v_j exists. A value of 0 indicates that there is no edge between vertices v_i and v_j [6].

Weighted Adjacency Matrix (Fig. 1) is thus a square matrix $W = n \times n$, where n denotes the number of vertices and the value of the element at the position w_{ij} corresponds to the weight of the edge between vertices v_i and v_j.

$$w_{ij} = \begin{cases} w(e_{ij}) & \text{if } e_{ij} \in E \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{array}{c|ccccccc} & v_1 & & v_i & \cdots & v_j & \cdots & v_n \\ \hline v_1 & 0 & \cdots & w_{1i} & \cdots & w_{1j} & \cdots & w_{1n} \\ \cdots & & & \cdots & \cdots & \cdots & \cdots & \cdots \\ v_i & & & 0 & \cdots & w_{ij} & \cdots & w_{in} \\ \cdots & & & & & \cdots & \cdots & \cdots \\ v_j & & & & & 0 & \cdots & w_{jn} \\ \cdots & & & & & & & \cdots \\ v_n & & & & & & & 0 \end{array}$$

Figure 1 Weighted Adjacency Matrix
Slika 1. Ponderirana matrica susjedstva

Weighted Adjacency Matrix is symmetric with respect to the main diagonal, the diagonal elements have a value of 0, the algorithm will only use the elements of the triangle above the main diagonal. The algorithm of searching for the MST works in the Weighted Adjacency Matrix and works with elements in the triangle above the main diagonal.

4. ALGORITHM PROCEDURE / *Izrada algoritma*

Search through the elements of the matrix and find the one with the smallest positive value w_{ij}. Denote chosen matrix element in bold and underlined, then mark the rows v_i, v_j and columns v_i, v_j (Denote the columns and rows with arrows at the top of the table). If there is more than one element with the same smallest positive value, it is possible to choose arbitrary one of these. Then more than one MST exists.

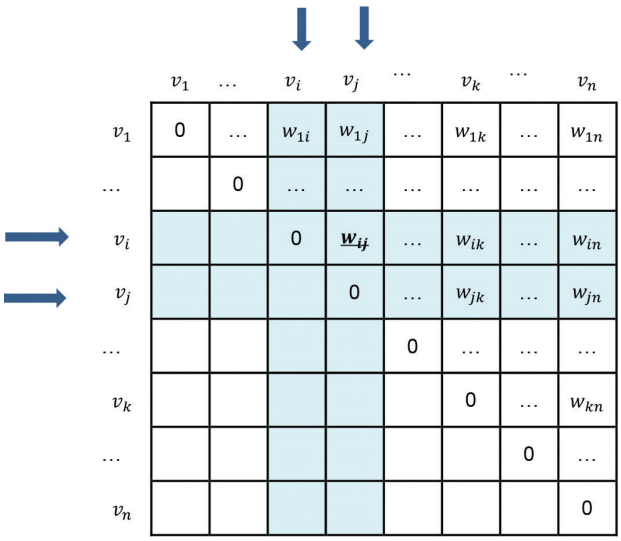

Figure 2
Slika 2.

Search again through the elements of the matrix and find another smallest positive element, search between elements in the marked rows and columns (Fig. 2). Denote the chosen element in the matrix in bold and underlined. Let the new element be w_{jk}. According to the index position of the element mark the row v_k and column v_k. Rows and columns marked in the previous steps remain marked.

This step ensures the connection of the generated MSTbecause the connecting edge has one of the index the same as the previous selected element, so this element connects to any of previously connected vertices.

Furthermore delete (i.e. replace by the cross) the elements in positions where newly marked row and column intersect with rows and columns previously marked. Here delete the element w_{ik}. This step prevents creating cycles.

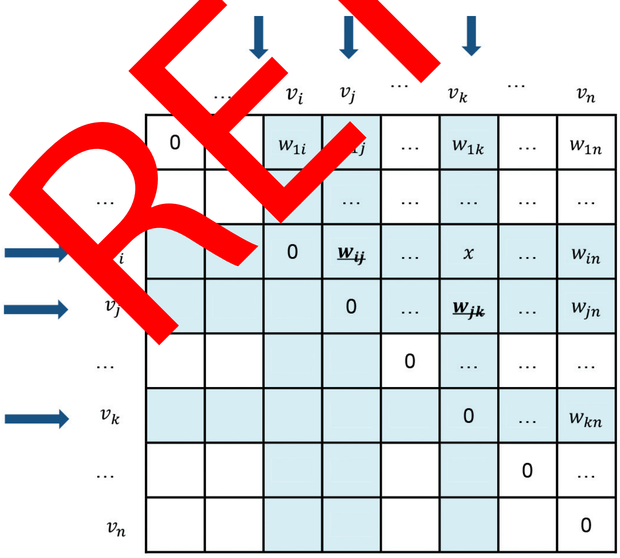

Figure 3
Slika 3.

Search again through the elements of the matrix and find another smallest positive element, search between elements in the marked rows and columns. Denote the chosen element in the matrix in bold and underlined (Fig. 3). Let the new element be w_{1i}.

According to the index position of the element mark the row v_1 and column v_1. Rows and columns marked in the previous steps remain marked. Furthermore delete all the elements in positions where newly marked row and column intersect with rows and columns previously marked. Here delete the elements w_{1j} and w_{1k} (Fig. 4).

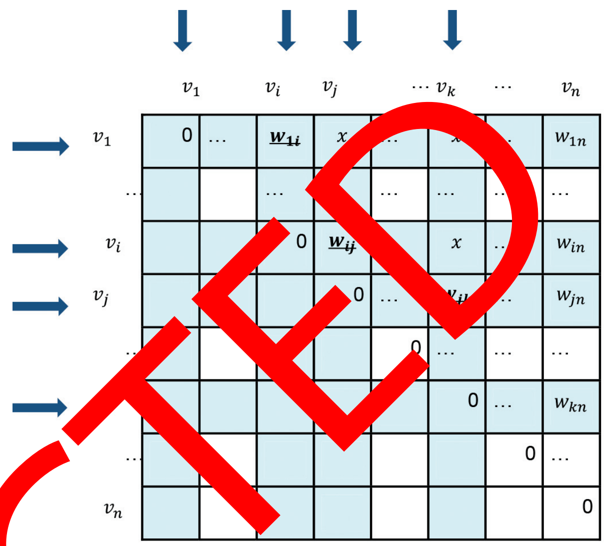

Figure 4
Slika 4.

Suppose that our algorithm made *k* steps.

- If $k = n - 1$, algorithm stops, we have made all steps
- If $k < n - 1$, we make $(k + 1)$-th step analogously

After we make the $(n - 1)$-th step in our Weighted Adjacency Matrix $(n - 1)$ chosen elements are labeled (in bold and underlined), the other elements (which were not chosen) are replaced by a cross. At the same time all rows and columns in our matrix are labeled (Fig. 5). Elements denoted in the matrix in bold and underlined are values of weights of edges of the MST. Labeling the rows and columns of the selected element indicates the vertices v_i and v_j that the edge on this position connects. The sum of the values of all chosen elements give the total weight of the MST.

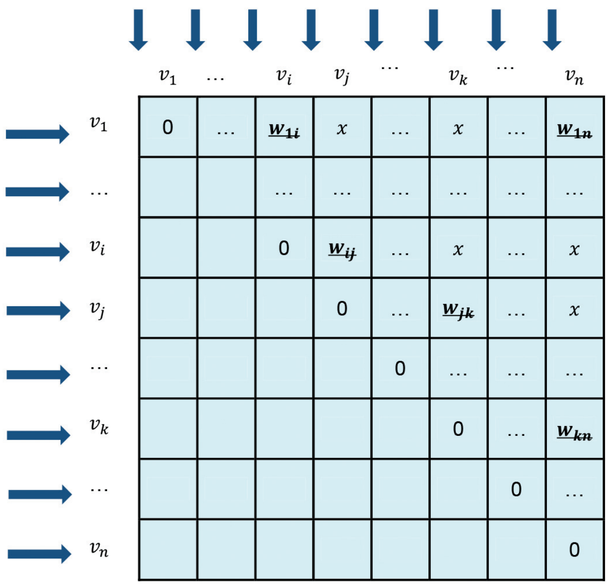

Figure 5 Final Matrix
Slika 5. Konačna matrica

5. VERIFICATION OF THE ALGORITHM / *Verifikacija algoritma*

1. Continuity of generated MST is guaranteed by the fact that newly connected edge has one of the indices the same as the indices of previously selected elements [4]. Therefore, it connects to one of the previously connected vertices.

2. The cycles are avoided by deleting all the elements in positions where newly marked row and column intersect with rows and columns previously marked [5].

3. The algorithm is a variant of the Prim´s algorithm, with the difference that in the first step we do not begin by selecting the arbitrary initial vertex, but in our Weighted Adjacency Matrix we begin by selecting the edge with the smallest weight. From the second step our algorithm works analogously as in the Prim´s algorithm (which has been proven, see [1]). This guarantees selection of the minimum spanning tree.

6. DEMONSTRATION OF SOLVED EXAMPLE/ *Prikaz riješenog primjera*

Imagine the system of the sea transport between many ports and their distances. Transport is going in many directions, conecting two or more ports. So the route can consist of one or more ports. At first we transform the system of transport directions into the weighted graph (Fig. 6). There are 6 ports represented by 6 vertices of the graph $v_1, v_2, \ldots, v_6$, connections between the ports are represented by the edges in the [illegible]h and numbers belonging to the edges represent the cos[illegible] o[illegible] energy consumed to conduct the ship (boat) between [illegible]o ports. As we said before, costs are in relation to distances, b[illegible] other expenses could be incorporated, too[illegible]ere w[illegible] have ful[illegible] mesh network structure but in reality it [illegible]ely occurs[illegible] Because of geografic topology it is not ration[illegible]l to [illegible]vel i[illegible]lar direction and skip some ports on t[illegible] path. In [illegible]ense MS[illegible] an acceptable solution for man[illegible]sport proble[illegible]

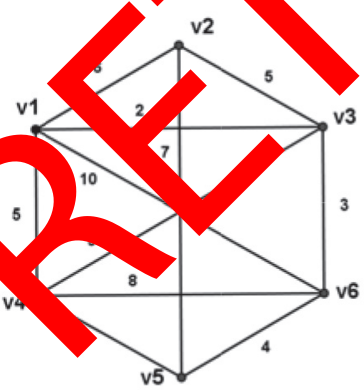

Figure 6
Slika 6.

Corresponding Weighted Adjacency Matrix is (Fig. 7):

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	6	2	5	0	10
v_2		0	5	0	7	0
v_3			0	9	0	3
v_4				0	7	8
v_5					0	4
v_6						0

Figu[illegible] Given Matrix
Slika [illegible]dana [illegible]trica

7. STEPS OF A[illegible]ORITHM / [illegible]*raci al*[illegible]*ritma*

1. Search [illegible]rou[illegible] the elements [illegible] th[illegible] matrix and find the one with the smallest [illegible]sitive value w_{13} = 2. Denote chosen matrix [illegible]ement in bold a[illegible]nderlined, then mark the rows v_1, v_3 a[illegible] columns v_1, v_3 (Fig. 8).

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	6	**<u>2</u>**	5	0	0
v_2		0	5	0	7	0
v_3			0	9	0	3
v_4				0	7	8
v_5					0	4
v_6						0

Figure 8
Slika 8.

2. Search again through the elements of the matrix and find another smallest positive element w_{36} = 3, search between elements in the marked rows and columns. Denote the chosen element in the matrix in bold and underlined. According to the index position of the element mark the row v_6 and column v_6. Rows and columns marked in the previous steps remain marked (Fig. 9). Furthermore we delete (ie. replace by the cross) all the elements in positions where newly marked row and column intersect with rows and columns previously marked. Here we delete the element w_{16}.

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	6	**<u>2</u>**	5	0	x
v_2		0	5	0	7	0
v_3			0	9	0	**<u>3</u>**
v_4				0	7	8
v_5					0	4
v_6						0

Figure 9
Slika 9.

3. Search again through the elements of the matrix and find another smallest positive element w_{56} = 4, search between elements in the marked rows and columns. Denote the chosen element in the matrix in bold and underlined. Accord[illegible] the index position of the element mark the row v_5 and [illegible]lum[illegible] v_5. Rows and columns marked in the previous steps [illegible]main marked. Furthermore delete all the element[illegible]ositions w[illegible] newly marked row and column intersect[illegible]th row[illegible]nd colu[illegible]s previously marked. Here delete the [illegible]ments, $w_{[illegible]}$ = x, w_{35} = [illegible] (Fig. 10).

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	[illegible]	6	**<u>2</u>**	5	x	x
v_2		[illegible]	5	0	7	0
v_3			0	9	x	**<u>3</u>**
v_4				0	7	8
v_5					0	**<u>4</u>**
v_6						0

Figure 10
Slika 10.

4. Search again through the elements of the matrix and find another smallest positive element w_{14} = 5, search between elements in the marked rows and columns. Denote the chosen element in the matrix in bold and underlined. According to the index position of the element mark the row v_4 and column v_4. Rows and columns marked in the previous steps remain marked. Furthermore delete all the elements in positions where newly marked row and column intersect with rows and columns previously marked. Here delete the elements w_{34} = x, w_{45} = x, w_{46} = x (Fig. 11).

	v_1	v_2	[illegible]	v_4	v_5	v_6
v_1	0	6	[illegible]	**<u>5</u>**	x	x
v_2		0	5	[illegible]	[illegible]	0
v_3			0	[illegible]	x	**<u>3</u>**
v_4				0	x	x
v_5					0	**<u>4</u>**
v_6						0

Figure 11
Slika 11.

5. Search again through the elements of the matrix and find another smallest positive element w_{23} = 5, search between elements in the marked rows and columns. Denote the chosen element in the matrix in bold and underlined. According to the index position of the element mark the row v_2 and column v_2. Rows and columns marked in the previous steps remain marked. Furthermore delete all the elements in positions where newly marked row and column intersect with rows and columns previously marked. Here delete the elements w_{12} = x, w_{24} = x, w_{25} = x, w_{26} = x (Fig. 12).

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	x	**<u>2</u>**	**<u>5</u>**	x	x
v_2		0	**<u>5</u>**	x	x	x
v_3			0	x	x	**<u>3</u>**
v_4				0	x	x
v_5					0	**<u>4</u>**
v_6						0

Figure 12
Slika 12.

8. TERMINATION OF THE ALGORITHM / *Kraj algoritma*

When the graph *G* has *n* vertices then the MST has *n* – 1 edges [3]. At each step of the algorithm we add to the gradually rising MST one edge, then algorithm makes *n* - 1 steps. In our example, the graph has 6 vertices, then the MST has 5 edges. That is the reason why algorithm makes 5 steps.

After the final step, all the elements in the Weighted Adjacency Matrix went through processing, ie. the edges chosen for the MST are denoted in the agreed way, i.e. here in bold and underlined, deleted edges are marked by symbol x.

At the same time, after the last step all the rows and columns of the matrix are marked with the arrows next to the rows and above columns.

9. FINAL GRAPH OF THE MST IS HERE / *Konačni graf MST- a*

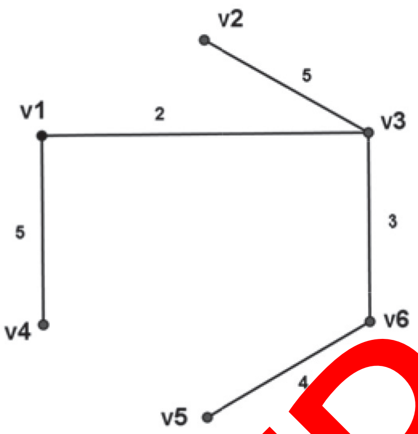

Figure 13 Final Graph of the MST
Slika 13. Konačni graf MST-a

10. VALUE OF THE FINAL MST IS / *Vrijednost konačnog MST-a je:*

$$\sum_{e\in E'} w(e) = (2+3+4+5+5) = 19$$

11. RESULT DISCUSSION / *Rasprava*

From results in fig. 13 we have spanning tree that generaly offers minimal expenses of transportation. But at glance it is not obvious how can we apply that route in practise. Firstly, any transportation means (ship, train, airplain etc.) has to be backed to home port, which results in extra costs. So, in practice we use loop structure for ship routing very often. In that way we can significantly reduce costs. But if we say that home port has to be in port v3 (in fig. 13.), the result makes sense. It is obvious that ship has three directions (branches) for shipping, toward port v2, to port v4 over port v1, and to port v5 over port v6. It means that we can optimally organize our fleet, by using periodically the same ship for each of the three directions, or to use more ships, each for one direction. In suchrouting solution we don't consider the problem of the ship's own capacity and loading/unloading strategy for each port. Also, we do nottake into accountthe cargo contingents and their amounts, transporting them from the predefined starting to the predefined ending port. If laod (cargo) influences the route definition then we have to use algoriths for multi-commodity flow problem [7] and [8].

We can say that such minimum spanning tree (MST) solution provides solidgrounds for strategic planning of shipmentin the maritime sector. This approach is more effective when cargo amount does not influece the transport costs, for example in passenger line transport. Especially in case of cruise ships where passengers mostlyembark the ship in hub ports with big airports, making trips in different directions from home port. Similar strategy could be efficient for the fishing fleet. Alternative route plannings are based on the shortest path algorthms and Transport Salesman Problem which is another technique based on graph theory, see [9] and [10].

12. CONCLUSION / *Zaključak*

This article describes the proposal of the algorithm for searching the minimum spanning tree. The proposed algorithm is similar to the Prim´s algorithm [1], which creates the minimum spanning tree as a gradually growing set of edges of the MST. In this regard there is a compliance with Prim´s algorithm. The Prim´s algorithm starts with the arbitrary vertice. Here, however, the first element the algorithm starts with the lowest weight edge.

The novelty here is the "Weighted Adjacency Matrix (abr. WAM)" It follows the principle of adjacency matrix known in the graph theory, but in the positions of matrix elements there are values of edges weights connecting the vertices. The vertices denote the rows and columns of the matrix.

The whole process of searching MST begins with choosing the smallest element of the matrix, representing the edge with the lowest weight. Gradually, we add elements so that another new element has one index the same as some of the elements that have been chosen in previous steps. This step guarantees the continuity of MST.

Elements which are not selected in the denoted rows and columns , must be removed because these edges would create cycles. The entire process takes place in WAM, the original graph is not needed.

Benefits of the proposed algorithm arethe efficient and fast searching of the MST by using WAM.. According to my knowledge the search of the MST by using WAM is a new tool and it can be assumed that the WAM could be used for solving other similar problems in the graphs, where wighted edges are given. The algorithm procedure is shown on the solved example.

The proposed algorithm is suitable for optimising the ship transport besause the distances onthe ship traffic routes can be easily transformed into the WAM which is clear represenation of the graph with weighted edges. Solving the problem of searching for the minimum spanning tree goes in this matrix quickly and is illustratively presented in the solved example. Generaly, WAM is used fordefinition of optimal ship transportation routes but for the routes based on cycles we have to apply quite a different approach.

REFERENCES / *Literatura*

[1] Harris J. M., Hirst J. L., Hossinghofer M. J. (2000). Combinatorics and Graph Theory, Springer, New York, ISBN978-0-387-79711-3

[2] Matoušek J., Nešetřil, J. (2009). Kapitoly z diskrétní matematiky, 4th edition, Prague, Charles University in Prague, ISBN 978-80-246-1740-4

[3] Kleinberg, J., Tardos, E. (2006). Algorithm Design, New York: Pearson Education, Inc.

[4] Kruska, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem, PAm Math Soc 7, 48–50. https://doi.org/10.1090/S0002-9939-1956-0078686-7

[5] Cormen, T. H., Leiserson, C. E., Rivest, R. L, and Stein, C. (2001). Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill, ISBN:0262033844 9780262033848

[6] Jackson, T. S. and Read, N. Theory of minimum spanning trees, Phys Rev E 81 (2010) 021130 https://doi.org/10.1103/PhysRevE.81.021131

[7] Ouorou, A., Mahey, P., Vial, J. Ph., (2000). A Survey of Algorithms for Convex Multi-commodity Flow Problems, Markup Languages, Vol. 46, No. 1, pp. 126-147.

[8] Castro, J., Nabona N., (1996). An Implementation of Linear and Nonlinear Multi-commodity Network Flows, European Journal of Operational Research, Vol. 92, No.1, pp. 37-53. https://doi.org/10.1016/0377-2217(95)00137-9

[9] Case Study: "Shortest-Path Algorithms" http://www.mcs.anl.gov/~itf/dbpp/text/node35.html#algdij1

[10] Krile, S., (2013). Efficient Heuristic for Non-linear Transportation Problem on the Route with Multiple Ports, Polish Maritime Research, Gdansk, Poland, Vol. 20, No 4, pp. 80-86, https://doi.org/10.2478/pomr-2013-0044