Graph methods for estimation of railway capacity

S.A. Branishtov[1], Y.A. Vershinin[2], D.A. Tumchenok[1], A.M. Shirvanyan[1]

[1]*Institute of Control Sciences, Russian Academy of Sciences; pochta-na@mail.ru*
[2]*Coventry University, Coventry, UK, o292j(a)gmx. com*

Abstract: Railway capacity is the important characteristic that shows traffic potentialities of railways. Analytical methods of the calculation of the capacity of a single track are well known. Methods to estimate the capacity of a railway network and the railway direction are proposed in this paper. The railway network is represented in the form of a graph. Mathematical methods of the graph theory are applied in order to find the maximum flow. Parameters of tracks and stations are used in order to calculate the network capacity.

Keywords: capacity of railway network, graph theory, the maximum flow.

1. INTRODUCTION

The availability capacity assessment is the technical certification of railway directions and it characterizes the railway freight traffic. The calculation results are used in the planning of the infrastructure development and traffic control. There are different methods to calculate the railway capacity. These methods can be divided into four categories: analytical methods, parametric models, optimization methods and methods based on simulation (Krueger, 2000; Abril et al., 2008).

The available capacity of a railway section is the maximum number of freight trains (couples trains) with the defined weight and length, which can be moved on this section per day depending on the technical equipment and the accepted method of the train traffic. The capacity of a railway station is the most probable number of freight trains (separately without processing and reprocessing) and a predetermined number of passenger trains that can be moved through a station per day in all directions under conditions of the work, ensuring the full use of the available infrastructure. The capacity calculation of each element does not estimate the overall network. It should be taken into account the mutual influence of elements.

This work answers the question of how to calculate the capacity of the railway network and directions based on the knowledge about the capacity of separate sections and stations. This will help one to utilise the infrastructure for the traffic control in the more efficient way, to "avoid" network bottlenecks and to develop methods for the increasing of the capability of the transportation infrastructure.

2. THE PROBLEM OF THE MAXIMUM FLOW

The capacity of some railway networks (subnet and direction) determines the maximum size of the railway traffic. It is sufficient to solve the problem of the finding the maximum flow in the graph in order to determinate the capacity. Then the value of the maximum flow capacity will comply with the capacity. For this purpose the railway network is represented as a graph whose vertices are the stations, and the edges represent tracks between stations. The capacity value of tracks must comply with parameters of edges. Then, solving the problem of finding the maximum flow in the graph, the total amount of the capacity between two given subnet vertices can be obtained.

There are various methods and algorithms to achieve this goal. For example, the following algorithms can be used: the Ford-Fulkerson algorithm, the Edmonds-Karp algorithm, the general push-relabel maximum flow algorithm, the push-relabel algorithm with dynamic trees and others (Cormen et al, 2009). The algorithm proposed by Ford and Fulkerson (Ford and Fulkerson, 2010) is based on the following theorem:

Theorem 1. Theorem of the maximum flow and minimum cut (Dantzig and Fulkerson, 1956; Lawler, 2001; Papadimitriou and Steiglitz, 1998) *The value of the maximum flow from s to t equal to the minimal cut* $(X_m \longrightarrow \widetilde{X_m})$*, separating s from t. Cut* $X_o \longrightarrow \widetilde{X_o}$ *separates s from t, if* $s \in X_o$ *and* $t \in X_o$*. Value of this cut (or capacity) is the sum of capacities of all the edges from* G*, whose initial vertices lie in* X_o*, and end in* $\widetilde{X_o}$*, i.e.*

$$v(X_o \longrightarrow \widetilde{X_o}) = \sum_{(x_i, x_j) \in (X_o \longrightarrow \widetilde{X_o})} q_{ij}.$$

Minimal cut$(X_m \longrightarrow \widetilde{X_m})$ *– is the cut with lowest value.*

Consider an example. Figure 1 shows a graph corresponding to the scheme of the subnet.

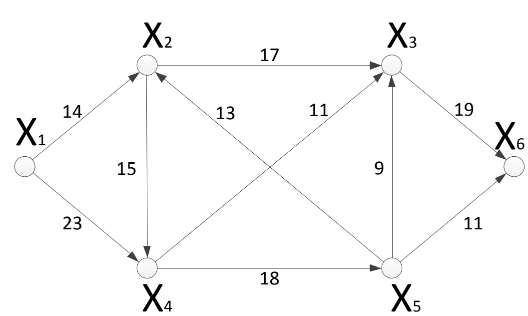

Fig. 1. A railway subnet with the section capacity

Vertices of this graph represent the railway station, and the edges represent tracks. "The number" on the edge of the graph represents the capacity of the corresponding track. The task is to find the subnet capacity. It is proposed in this paper to use the Ford-Fulkerson algorithm and to find the maximum flow between vertices x_1 and x_6 in order to solve the above task. The solution can be found in the form of the distribution of flows over edges of the graph (Figure 2).

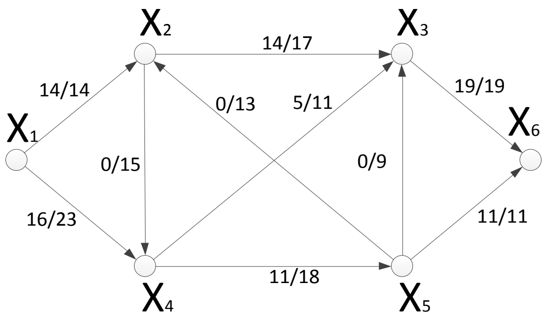

Fig. 2. The flow distribution.

Values on the edges of the graph correspond to the passing the flow/edge capacity. The maximum flow between vertices x_1 and x_6 is $f_{x_1 x_6} = 30$.

This method uses the capacity of direct tracks only, without branches and ignores stations properties. The method finds the subnet capacity for exactly defined entry and exit points of trains. This method is applicable for graphs with directed edges only.

3. GRAPHS WITH CAPACITIES OF EDGES AND VERTICES

Railway stations use their own measure of the capacity. The application of methods to find the maximum flow for the capacity subnet estimation will give doubtful results in the Also, each edge (x_i, x_j) in G (which incidence x_j) corresponds to edge (x_i^-, x_j^+) from G_o (incidences x_j^+) and each edge (x_j, x_k) from G, coming from x_j, corresponds to (x_j^-, x_k^+) from G_o (coming from x_j^-). Additionally, the edge between x_j^+ and x_i^- with capacity w_j is introduced, i.e. equal vertex capacity x_j.

Figure 3 shows an example of a graph with the capacity of edges and vertices, and Figure 4 shows the graph G_o, which is built according to the above description. Since the total flow entering the vertex x_j^+, should proceed along the edge (x_j^+, x_j^-) with the capacity w_j, then the maximum flow in the graph G with the capacity of edges and vertices equal to the maximum flow in the graph G_o (which has edges capacities only). If the minimal cut in G_o does not contain edges of the form (x_j^+, x_j^-), then the capacities of vertices in G unnecessary and it should not be considered. If the minimal cut in G_o contains such the edge, then the corresponding vertices are located in G. If the minimal cut in G_o contains such an edge, then the corresponding vertices in G are saturated by the received maximum flow.

case if this measure has not been considered. In other words, a station can limit the subnet capacity. Therefore, in order to obtain the more accurate assess of the subnet capacity, sections capacities and stations capacities must be taken into the account.

The station capacity is determined by the following technical resources:

- receiving parks and dispatch parks;
- connecting tracks (between parks);
- passenger platform lines;
- technical tracks of passenger stations;
- leads of all types of stations.

Mathematical methods are employed in this paper in order to take into the account the station capacity and to solve the problem of the finding the maximum flow in the graph of the railway network.

Let the edges have capacities q_{ij}, and let the vertices of the graph have capacities $w_j (j = 1,2, \dots, n)$ such that the total flow entering the vertex x_j, must have a value less than w_j, i.e.

$$\sum_{x_i \in T^{-1}(x_j)} \xi_{ij} \leq w_j \; for \; all \; x_j.$$

It is required to find the maximum flow between vertices *s* and *t* on the graph.

The graph G_o defines the graph in such way that for each vertex x_j of the graph G two vertices x_j^+ and x_j^- in graph G_o are corresponded.

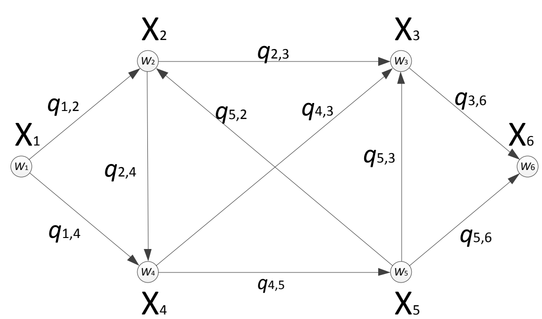

Fig. 3. Graph whose vertices and edges with attributed capacity.

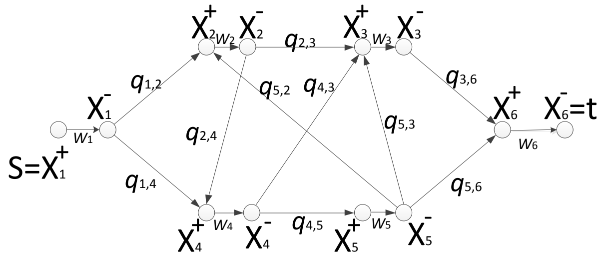

Fig. 4. Equivalent graph in which capacities have an edge only.

The example of how to change the maximum flow at the subnet with the given capacities of nodes is demonstrated below. To do this, we add to the railway subnet (Fig. 1) stations capacity. Figure 5 shows a subnet with tracks and stations capacities.

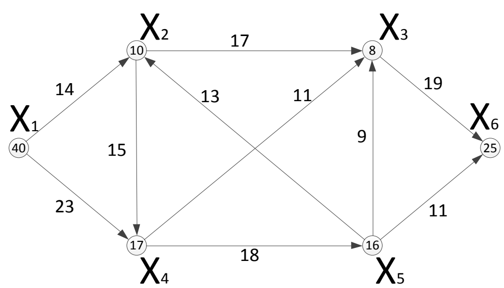

Fig. 5. Railway subnet with tracks and stations capacities.

We transform the graph into an equivalent so that an only edge has the capacity (Fig. 6):

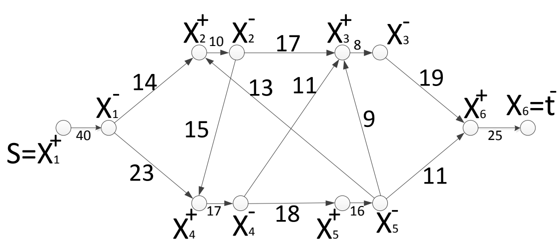

Fig. 6. Equivalent graph in which an edge has the capacity only.

We calculate the maximum flow of the graph shown in Figure 6 by one of the methods. Flows on the last step of the algorithm as follows:

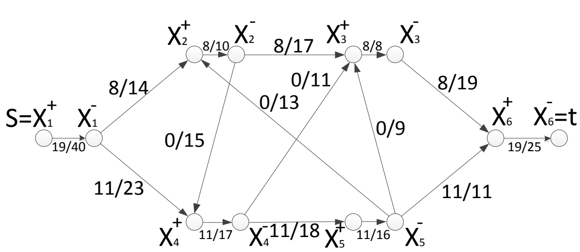

Fig. 7. Flows on the graph at the last step of the algorithm

On Figure 7 "numbers" on the edges of the graph correspond to the passing flow / capacity. Maximum flow between vertices x_1 and x_6 is $f_{x_1 x_6} = 19$. Flow of edges (x_5^-, x_6^+) and (x_3^-, x_3^+) reached maximum capacity. The edge (x_3^-, x_3^+) on the graph corresponds to the capacity of the station X_3.

4. GRAPHS WITH MULTIPLE SOURCES AND SINKS

The Ford-Fulkerson method allows one to find the maximum flow between two vertices of the graph. This parameter corresponds to the subnet capacity for a couple of specific vertices of the railway network, and does not represent the overall network capacity. Next, this parameter will be called direction capacity.

When we calculate the bandwidth direction, we take into account several sources of flow of trains and several sinks (destination stations). Problem of several sources can be solved by introducing an artificial source vertex, which is common to all vertices sources. You can also make the sinks.

We consider a graph with s_n sources and t_m sinks and assume that the flow can come from any source to any sink. The problem to estimate the maximum flow from all sources to all sinks can be converted into a simple problem of maximum flow (from s to t) by adding a new artificial source s and a new artificial sink t by adding edges leading from s to each original source and from each original sink to t.

Figure 8 shows how a set of sources and sinks can be reduced to a single source and a single sink. Edges capacities leading from s_n sources can be chosen equal to infinity. Similarly, the capacities of edges leading from sinks to t are assumed equal to infinity.

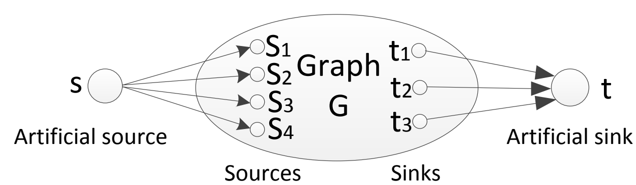

Fig. 8. Graf with artificial source and sink

5. MAXIMUM FLOW BETWEEN EACH PAIR OF VERTICES

How to estimate the maximum capacity of subnet between any two stations *s* and *t*? If you calculate the maximum flow for each pair of vertices in a large graph it is very time-consuming work. Or you can use the Gomory and Hu algorithm (Gomory and Hu, 1964) that is presented below. The algorithm is more efficient in the case of undirected graphs. This algorithm uses two important concepts – equivalence flow and packing vertices.

5.1 Flow equivalence

Theorem 2. *Let f_{ij} is the maximum flow from the vertex x_i to vertex x_j of graph G. (Since G is undirected graph, then $f_{ij} = f_{ji}$). These flows satisfy $f_{ij} \geq min[f_{ik}, f_{kj}]$ for each i, j, k.*

If any graphs have equal maximum flows between a certain set of vertices then these graphs are called “flow-equivalent” or simply “equivalent” with respect to this set of vertices. There is a flow-equivalent graph for any other graph anytime.

5.2. Compressing the vertices

The main idea is that several vertices in the graph will be presented by one vertex. The edges between the compressed vertices will have the infinite capacity. The edges from compressed vertices will transformed to single edge with common flow. Let assume the maximum flow problem for a graph *G* is solved for two randomly selected vertices *s* and *t*. Let $(X_o, \widetilde{X_o})$ is minimal cut corresponding to the maximum flow, then consider the two vertices x_i and x_j, both are lying in X_o. If we want to estimate the maximum flow f_{ij} from x_i in x_j, then all the vertices of $\widetilde{X_o}$ can be "compressed" in one vertex $\widetilde{x_o}$. This compression is such that the edges (x_a, x_b), x_a and $x_b \in X_o$ are replaced by edges $(x_a, \widetilde{x_o})$ and any parallel edges between the same pair of vertices are replaced by a single edge, whose capacity equal to the sum of capacities of parallel edges. Figures 9, 10 illustrate the compression process. Gomory and Hu (Gomory and Hu, 1964; Hu, 1970) have approved this compression set $\widetilde{X_o}$ is possible.

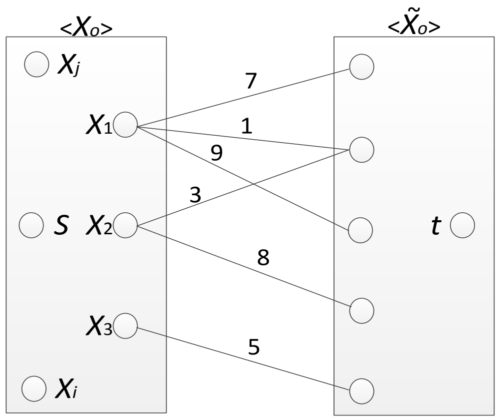

Fig. 9. Minimum cut from s to t

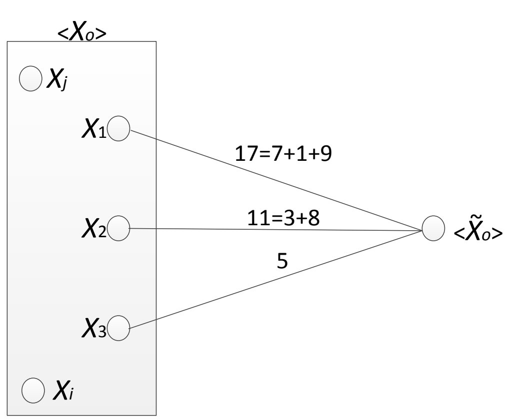

Fig. 10. Graph after compression

6. AN ALGORITHM FOR CONSTRUCTING THE MAXIMUM FLOW BETWEEN ALL PAIRS OF VERTICES

The Gomory and Hu algorithm generates a tree T^*, which flow-equivalent for undirected graph G. The maximum flow f_{ij} between two vertices x_i and x_j for graph G can be found as follows in this tree:

$$f_{ij} = \min\left[q'_{ik_1}, q'_{k_1k_2}, q'_{k_2k_3}, \dots, q'_{k_pj}\right], \qquad (3)$$

where $(x'_i, x'_{k_1}, x'_{k_2}, \dots, x'_j)$ — the single chain that coming on the edges of the tree T^* and leading from x'_i to x'_j. Each vertex x'_k from T^* corresponds to the vertex x_k from G and q'_{k_1} is the capacity of edge (x'_k, x'_i) from T^*.

The idea of the algorithm is as follows: there exists a tree T^*, which flow-equivalent for graph G, and T^* contains only $n-1$ edges, so it is sufficient to calculate the capacities of $n-1$ edges of T^*. You compress a vertices on each step, get new graph and calculate the flow.

Description of the algorithm. Let the vertices from G are called G - vertices, and the vertices from T^* are T^*-vertices. (Kristofides, 1978)

Step 1. Let $S_1 = X, N = 1$.

T^* is a graph at any stage that has N vertices $S_1, S_2, \dots, S_n$, and each of them corresponds to a certain set from G-vertices. Initially, the graph T^* consists of a single vertex.

Step 2. Find the set $S^* \in \{S_1, S_2, \dots, S_n\}$, that is containing more than one vertex. If such does not exist, then go to step 6, otherwise go to step 3.

Step 3. If S^* was removed from T^*, then the tree would disintegrate into several subtrees (connected components). Compress T^*-vertices in each subtree in one vertex and form a graph with *S* vertices. Take any two vertices x_i and $x_j \in S^*$ and find a minimal cut $(X_o, \widetilde{X_o})$ in G, separating x_i from x_j, , by calculating the maximum flow (from x_i to x_j).

Step 4. Remove from T^* vertex S^* together with the connected edges, and replace it with two T^*-vertices, composed of sets of G-vertices $S^* \cap X_o$ and $S^* \cap \widetilde{X_o}$, and an edge between them with the capacity of $v(X_o, \widetilde{X_o})$. So, let consider each T^*-vertices S_i, which were incident S^* (with a capacity of the edge equal c_i^o), then add to T^* the edge $(S_i, S^* \cap X_o)$, if $S_i \subset X_o$, or for T^* add edge $(S_i, S^* \cap \widetilde{X_o})$, if $S_i \subset \widetilde{X_o}$. Capacities of edges in this and in another case are c_i^o.

Remark. As previously noted, Gomori and Hu (Gomory and Hu,1964) showed that S_i lies in X_o or $\widetilde{X_o}$ entirely.

Step 5. Let $N = N + 1$. Vertices of T^* are now sets of vertices $S_1, S_2, \dots, S^* \cap X_o, S^* \cap \widetilde{X_o}, \dots, S_n$, where S^* is replaced by two T^*-vertices $S^* \cap X_o$ and $S^* \cap \widetilde{X_o}$, as explained above. Go to step 2.

Step 6. Stop. T^* is flow-equivalent graph to G now. Its T^*-vertices are the only G-vertices. Capacities of edges in T^* correspond to $(n-1)$ independent cuts in G. Equation (3) can be used to calculate f_{ij} (for each $x_i, x_j \in X$) now directly from T^*.

The algorithm gives the best result that follows directly from the properties of minimal cuts and properties of flow-equivalent graph G tree above. A formal proof can be found in Hu (Hu, 1970).

Example. Let's consider an undirected graph G (see fig. 11), representing the railway subnet. Capacities are shown as numbers on the edges. We need to find the maximum flow between each pair of vertices of G. The above algorithm will applied.

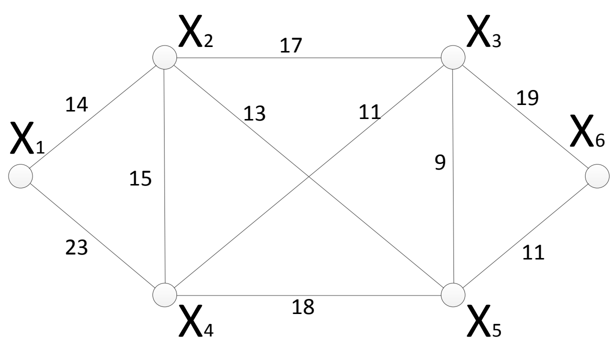

Fig. 11. Undirected graph

Step 1. $S_2 = \{x_1, x_2, x_3, x_4, x_5, x_6\}$; $N = 1$.

Step 2. $S^* = S_2$.

Step 3. Graph cannot be compressed. Let's take $x_i = x_2$ and $x_j = x_3$ randomly, then calculate the maximum flow (from x_2 to x_3), thus we find that the minimum cut is $(X_o, \widetilde{X_o})$, where $X_o = \{x_1, x_2, x_4, x_5\}$ and $\widetilde{X_o} = \{x_3, x_6\}$, and the value of cut equal 48.

Step 4. Tree T^* and capacities of its edges are shown in Figure 12.

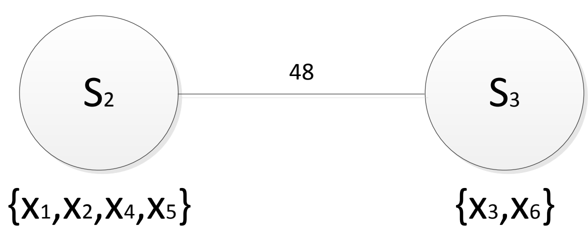

Fig. 12. T^* after first stage

Step 5. $N = 2$ (i.e. T^* is now have two vertices as shown in Figure 12: S_1 and S_2).

Step 2. We take $S^* = S_3$.

Step 3. We choose $x_i = x_3$ and $x_j = x_6$. Result compressed graph is shown in Figure 13. When we are calculating the maximum flow of the graph, the minimal cut $(X_o, \widetilde{X_o})$ will be found, where $X_o = \{x_6\}$ and $\widetilde{X_o} = \{\underbrace{x_1, x_2, x_4, x_5}_{S_1}, x_3\}$. The value of this cut equals 30.

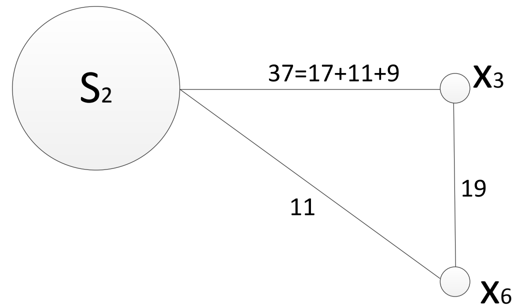

Fig. 13. Compressed graph after 2nd step

Step 4. T^*-vertex $S_3 = \{x_3, x_6\}$ is replaced by two new T^*-vertex $\{x_3\}$ and $\{x_6\}$ now. The new tree is shown in Figure 14.

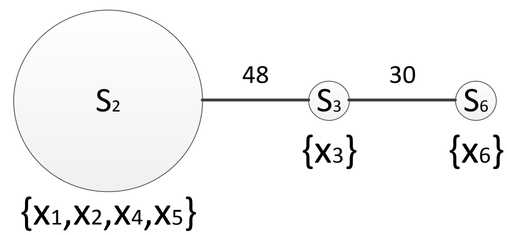

Fig. 14. T^* after 2nd step

Step 5. $N = 3$ (i.e. T^* has three vertices as shown in Figure 14).

Step 2. Let's take $S^* = S_2$.

Step 3. Take $x_i = x_1$ and $x_j = x_2$. Compressed graph shown in Figure 15. Minimal cut with a value 37 is equal to $(X_o, \widetilde{X_o})$, where $X_o = \{x_1\}$ and $\widetilde{X_o} = \{x_2, x_3, x_4, x_5, x_6\}$.

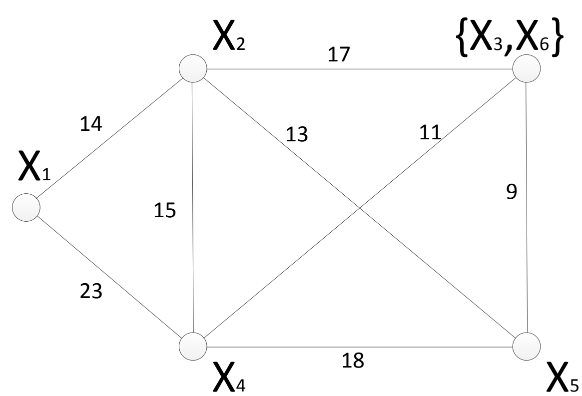

Fig. 15. Compressed graph after the third stage

Step 4. T^*-vertex $S_2 = \{x_1, x_2, x_4, x_5\}$ is now replaced by two new T^*-vertex $\{s_2\}$ and $\{x_1\}$ and a new tree is shown in Figure 16.

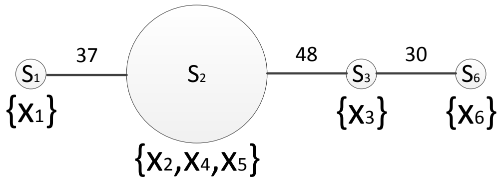

Fig. 16. T^* after 3-rd stage

Continuing in this way and taking $S^* = S_2$, consistently get flow-equivalent trees. The resulting flow-equivalent tree shown in Figure 17.

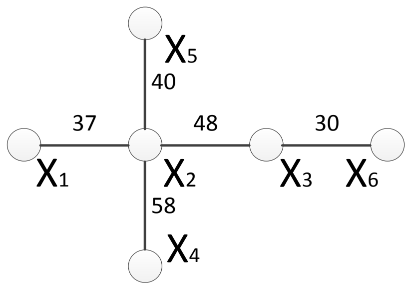

Fig. 17. The final flow-equivalent tree

Now we can calculate the matrix of maximum flows of original graph directly from (3) using the Figure 17. Matrix is presented in Table 1.

Table 1.Matrix of maximum flows

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	–	37	37	37	37	30
x_2	37	–	48	58	40	30
x_3	37	48	–	48	40	30
x_4	37	58	48	–	40	30
x_5	37	40	40	40	–	30
x_6	30	30	30	30	30	–

7. CONCLUSION

The article shows a simple way to solve the problems of estimating the capacity of railway subnets and directions. For this purpose railway subnets are represented as a graph whose vertices are the station, and the edges are tracks connecting them. The Ford-Fulkerson method and the Gomory-Hu method are applied in order to finding the maximum flow in the graph. The example shows how to take into account the capacity of a station in the evaluation of the capacity of a subnet.

REFERENCES

Abril M., Barber F., Ingolotti L., Salido M. A., Tormos P., Lova A., An assessment of railway capacity. *Transportation Research Part E: Logistics and Transportation Review*, 44, 2008. – pp.774-806.

Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford. Introduction to Algorithms. — 3rd. — *MIT Press*, 2009. — ISBN 0-262-03384-4

Dantzig, G. B., and D. R. Fulkerson, On the Max-Flow Min-Cut Theorem of Networks, Linear Inequalities and Related Systems, Annals of Mathematics Study No. 38, *Princeton University Press*, Princeton, 1956. – pp. 215-221

Ford, L.R., Jr.; Fulkerson, D.R., Flows in Networks, *Princeton University Press*, 2010. – 194p.

Gomory R.E, Hu T.C., Synthesis of a communication network, *J. Soc. Indust. Appl. Math.* Vol. 12, No. 2, June, 1964. – p.348

Hu T.C., Integer programming and network flows, *Addison Wesley*, *Reading*, Massachusetts, 1970. – p.520

Kristofides N., Teorija grafov. Algoritmicheskij podhod. – *M.: Mir* (In Russian), 1978. – 432p.

Krueger H., Parametric modelling in rail capacity planning. *1999 winter simulation conference*, 1999. – pp.1194-2000.

Lawler E. "4.5. Combinatorial Implications of Max-Flow Min-Cut Theorem, 4.6. Linear Programming Interpretation of Max-Flow Min-Cut Theorem". *Combinatorial Optimization: Networks and Matroids.* Dover, 2001, pp. 117–120. – ISBN 0-486-41453-1.

Papadimitriou C.H., Steiglitz K.. "6.1 The Max-Flow, Min-Cut Theorem". *Combinatorial Optimization: Algorithms and Complexity*. Dove, 1998, pp. 120–128.– ISBN 0-486-40258-4.