ISSN 1648-4142 TRANSPORT
http:/www.vtu.lt/english/editions

TRANSPORT – 2002, Vol XVII, No 2, 60– 65

SEARCH FOR THE OPTIMAL WAY IN THE NETWORK OF PUBLIC COMMUNICATION

Artūras Keršys[1], **Algirdas Jurkauskas**[2]

[1] *Kaunas University of Technology, Kęstučio g. 27, LT-3004 Kaunas, Lithuania,*
[2] *Kaunas University of Technology, Panevėžys branch, Klaipėdos g. 1, LT-5300 Panevėžys, Lithuania.*

Received 2002 02 05; accepted 2002 03 04

Abstract. The problem of searching the optimal way in the network of public communication is investigated in the article. The modified Minieka algorithm and designed combined search into depth and Dijkstra algorithm method, operating by the basis of graphs are presented. In order to evaluate practically the efficiency and applicability of algorithms experimental calculating of route search in the model of public transport communication network has been made. Public transport network changed in to graph is a real public transport network segment in Kaunas. Having compared their efficiency (memory size and calculating time) it has been proved that both methods might be applied to solve the problem as well as be integrated into the prediction models of communication processes. The combined depth-first search and Dijkstra algorithm method are better.

Keywords: public communication, the model of the network, search of the way, function of the resistance, graph.

1. Introduction

The paper is concerned with the problem of searching for the optimal route, which is an old and classic problem in the theory of the transport service system. There are many algorithms for solving the problem based on different ideas [1, 2]. More precise rout esearch in the theory of graph algorithms analyzing historical development in the Lithuanian and foreign scientific publications revealed that the above mentioned algorithms are constantly perfected reducing the time of calculation [3–6]. Bearing in mind the requirements of modern computation, the problem seems to be not very tipical, but in the case of the big network calculation time first of all it depends not on the speed of computing, but on the method of computing.

In practice, when solving the problem of specific optimal route in public transport service network, it is a much time consuming task to adjust already known graph theory algorithms for route search resulting in negative influence on efficiency [7, 8].

The essence of this paper is to design the efficient algorithm for optimal route in public communication network modifying already known graph theory algorithms.

[1] E-mail: artker@centras.lt
[2] E-mail: algirdas.jurkauskas@mf.ktu.lt

2. The Problem of Searching for the Optimal Way in the Network of Public Communication

Forecasting transport services with the help of computers it is necessary to use modelling of typical changes in the specific transport service network [9]. While modelling communication route choice, firstly the routes of communication that will be chosen by a participant to reach the destination point should be established. Knowing these ways it is possible to estimate transport flow in all the parts of public communication network and to foresee if future demand exceeds the supply at specific points of the path. Thus the measures can be taken to expand or shift the supply. Namely therefore the results of modelling the public transport service route are significant in transport planning.

The pattern of public transport communication is the basis of problem solving in the way search (Fig 1). Model structure of public communication network is nominally described in the following way:

$$VS = (TR, S, L, P, \tau). \qquad (1)$$

Where *TR* is finite set of transport regions. *S* is the finite set of stops, $S \cap TR\{\ \}$, when there is at least one path (line) at every stop: $\forall s \in S : \exists l \in L : s \in l$. $L \subseteq S^2 \times S$ - finite path set. $P \subseteq TR \times S$ - finite paths on foot set between transport regions and stops. $\tau : L \rightarrow IR^+$ - traffic schedule of the path (line) – time interval between two sequencing carriages in a path.

All the route search methods are based on the graph theory. Accordingly, in order to adjust and upgrade already known algorithms of graph theory dealing with the problem, the graph must be substituted for the network model.

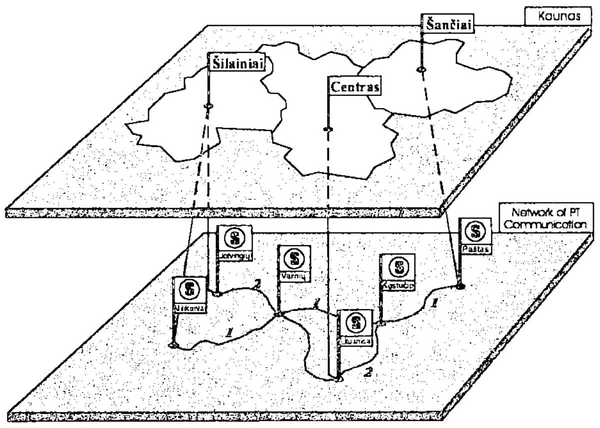

Fig 1. Public transport communication network model after experimental computation

Equivalent orientated finite graph for network model is as follows:

$$G = (V, E, \gamma). \tag{2}$$

Where $V = \{1,2,...,N\}$ - finite peak set $(N \in IN)$. $E \subseteq V^2$ - orientated arc set. $\gamma : E \to IR_0^+$ - arc evaluation function (arc length).

Substituting graph for the model of communication network, peaks are derived instead of transport regions and stops and arcs and their length instead of footways, paths (line) interval, transfer relations and resistance.

One of the main tasks of the route search algorithm problems is the selection of right one, as a rule, from infinite set of the model in all the routes of public communication network. Transport service connections must be sorted out in the way as to be able to compare one route with the other. For this purpose every road must be given the resistance that corresponds tangible and intangible passenger costs [10]. Only those criteria are studied that are quantifiable and have essential impact on the choice of the route. These specific influential factors are journey time, carriage cost and comfort.

Establishing all kinds of resistance function ω is defined and its value corresponds leverage time:

$$\omega = \omega(p) + \omega(a) + \omega(u). \tag{3}$$

In this equation $\omega(p)$ – footway $p = (tr, s) \in P$ resistance is defined as leverage time that is needed to reach destination on foot p and is derived from road length and pedestrian speed dependence. Resistance of $\omega(a)$– path segment $a = (s, \bar{s}, l) \in A$ is defined as leverage time, that is used to go from stop s to stop $\bar{s}$ by line l. Resistance of $\omega(u)$ transfer relations $u = (l_1, l_2, s) \in U$ comprises leverage waiting time during transfer and additional time for loss of comfort.

The modelling of transport needs is based on the assumption, that passengers choose the way with minimal resistance, thus the way with minimal tangible and intangible journey costs. In graph G optimal route $k_0 \in K_{i,j}$ between two peaks $i, j \in V$ is a path of minimal length: $\forall k \in K_{i,j} : \gamma(k) \geq \gamma(k_0)$. Thus $d_{i,j} = \gamma(k_0)$ is the length of optimal route between peaks i and j.

In route choice modelling only the optimal route is analyzed, thus estimated transport need in line segments, having no optimal route, equals to zero, although in most ways there is an equivalent alternative of the shortest path. To represent communication needs in a more realistic way, k-optimal routes must be included $(k \in IN)$. Critical factor is the resistance of the alternate route emerging in fixed limits. Lower limit is the length of optimal route

$d_{i,j}$ and the upper limit is calculated in this way:

$$D_{i,j} = d_{i,j} + \lambda_2. \tag{4}$$

In this equation

$d_{i,j}$ is the length between two objects i and j (peaks, transport regions, stops) of optimal way. In route choice all roads are important, that have the resistance higher than the length of optimal route

$d_{i,j}$ in constant time addition. If the distance between the start and the end of the journey is small, this constant time addition is unrealistically big. Then a relative time addition is used instead of absolute addition. With function of limited values Γ the upper limit of optimal route

$d_{i,j}$ is calculated:

$$\Gamma(d) = Min(\lambda_1 * d_{i,j}, d_{i,j} + \lambda_2). \tag{5}$$

Standard parameter values are: $\lambda_1 = 1{,}2$ and $\lambda_1 = 15$ [min]. Thus, in the route choice modelling all the routes with the resistance up to 20% higher than the optimal route length are analyzed, but not longer than 15 minutes. In route search algorithms there are always more than needed routes calculated, because in case of small distances upper limit $D_{i,j}$ is too high. At the end of the route search the upper limit must be adjusted and redundant routes should be excluded (Fig 2).

3. The Algorithms of Search of the Optimal Way in Network of Public Communication

3.1. The Modified Minieka Algorithm

The Minieka algorithm is used to solve the task of finding the shortest way k in the graph. The method was created after modifying the standard Floyd algorithm of finding the shortest way in the graph. For every peak pair $(i, j) \in V^2$ k shortest ways are defined (k is constant) while inserting an intermediate m peak. In this way all the

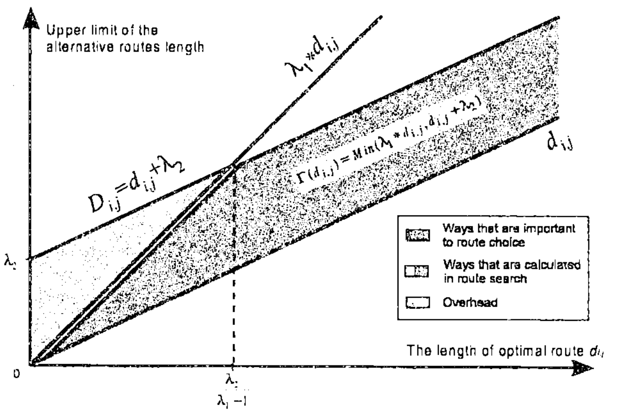

Fig 2. The dependence of the alternative route upper limit Γ from optimal route length $d_{i.j}$

route combinations between i and m, as well as between m and j are obtained and finally the new way of the pair (i, j) is obtained. The optimum ways k are selected, all the rest is deleted. The way combination from and to m for each peak $m \in V$ is calculated only once. As all the defined ways are as a rule non-cyclical the set for each apex is fixed. As only the non-cyclical ways are being defined, the calculations attributed to them fall away.
The Minieka algorithm defines any constant number of ways k for each pair of peaks (the lengths of ways k in the database entries). In route search of a way communication network the upper limit $D_{i,j}$ is only for the way length, not for the number of alternative routes. That is why for flexibility reasons a set of k-entries is introduced instead of the above-mentioned limit. The upper limit $D_{i,j}$, is to be known before starting the calculation. In this way the optimum length of the route is defined in its preparation phase. The Dijkstra algorithm is used for this purpose.

In the Minieka algorithm we suppose that all the lengths defined by the peak pairs are different which is not common for general graphs. All homogenous length roads are rejected. The problem is solved substituting the road lengths with sequences of peaks (road lists). A list of roads between these two peaks is made for each peak pair in the matrix $N \times N$.

The way list $P_{i,j}$ of the peak pair $(i, j) \in V^2$, $i \neq j$ the set of ways belonging to the solution set $X_{i,j}$:

$$P_{i,j} = \{k = (v_1, v_2, \ldots, v_n) \in K_{i,j} \quad \forall 1 \leq x < y \leq n : \\ v_x \neq v_y, \gamma(k) \leq D_{i,j}, \mu(k) \leq u_{\max}\} \tag{6}$$

The connection of two way lists $P_{a,b}, P_{b,c}$, $a, b, c \in V$ is the set of several combinations from $P_{a,b}$ and $P_{b,c}$ with the following limitations:

$$P_{a,b} \circ P_{b,c} := \{k_1 \circ k_2 \mid k_1 \in P_{a,b}, k_2 \in P_{b,c} \\ \gamma(k_1) + \gamma(k_2) \leq D_{i,j}, \ \mu(k_1) + \mu(k_2) \leq u_{max}\} \tag{7}$$

$k_1 \bullet k_2$ is the way connection. The condition of the cycle $k_1 \cap k_2 = \{b\}$ assures that no peaks appear until the connecting peak b, therefore only the ways belonging to the solution set are obtained. When $P_{a,b} = \{\ \}$ or $P_{b,c} = \{\ \}$, then $P_{a,b} \circ P_{b,c} = \{\ \}$. The following sequence in the algorithm is obtained: $P_{i,j}^{(0)}, P_{i,j}^{(1)}, \ldots, P_{i,j}^{(N)}$.

Procedure.

Given data:

Graph $G = (V, E, \gamma)$.

To be searched:

The optimal non-cyclical way from i to j $(i, j \in V, i \neq j)$ with the time limit $D_{i,j}$ and the largest frequencies of bus changes $u_{\max}$. The ways obtained are entered into $P_{i,j}$.

Preparation:

- $\forall i, j \in V, i \neq j$: Dijkstra algorithm defines the optimal way legth $d_{i,j}$ from I to j. As the algorithm is the 1:N type search method, the calculation for each peak is executed N number of times.
- $\forall i, j \in V, i \neq j := d_{i,j} + \lambda_2$ (temporary upper limit).

Formation:

First of all there are only the arc connections which lengths appear within the framework of the calculated limit:

$$\forall i, j \in V, i \neq j : P_{i,j}^{(0)} := \{(i, j)\}, \text{ when } (i,j) \in E \\ \text{and } \gamma((i,j)) \leq D_{i,j}, \quad P_{i,j}^{(0)} := \{\ \} \tag{8}$$

Basic part:

```
FOR m: = 1 TO N DO
  FOR i: = 1 TO N DO
    FOR j: = 1 TO N DO
      IF $i \neq m$ AND $j \neq m$ AND $i \neq j$ THEN
        $P_{i,j}^{(m)} := P_{i,j}^{(m-1)} \cup (P_{i,m}^{(m-1)} \circ P_{m,j}^{(m-1)})$;
      END;
    END;
  END;
END.
```

The correction of the upper limit:

$\forall i, j \in V, i \neq j : P_{i,j} := \{k \in P_{i,j}^{(N)} \mid \gamma(k) \leq \Gamma(d_{i,j})\}$ with Γ

Estimation of expenses:

In order to evaluate the costs it is important to fulfill the condition:

$$\exists k \in IN : \forall i, j \in V : \ |X_{i,j}| \leq k, \tag{9}$$

i.e. the size of the solution set is limited by figure k.. Without the limitation it would be impossible to compare the road search algorithms (all method $O(2^N)$). From $|P_{i,j}| \leq |L_{i,j}|$ the final conclusion follows, that $|P_{i,j}| \leq k$, i.e. the size of the list is limited by figure k.

Calculating procedure parameters:

Preparation: optimum way search according to Dijkstra algorithm requests $O(N \cdot \log(N))$. As 1:N method, is repeated N times for every peak.

- Optimal way search N:N: $\in O(N^2 \cdot \log(N))$;
- Superior limit $D_{i,j}$ calculation: N^2 pitch;

Formation: connecting two lists of ways $P_{i,m}$, $P_{m,j}$:

- cycle check ($k_1 \cap k_2$) : $\in O(N \cdot \log(N))$;
- connection $k_1 \circ k_2$: N pitch (way being copied);
- the number of way combinations $P_{i,m}$ and $P_{m,j}$:

  k^2 (because of $|P_{i,j}| \le k$);
- connection total: $k^2 \cdot (N \cdot \log(N) + N)$.

  Total number of checked cycles:

  $N \cdot (N-1) \cdot (N-2)$, because $m \ne i \ne j$.

Superior limit correction: N^2 pitch.
Elimination of redundant ways: $N^2 \cdot k$ pitch.
Total expenses:

$$N^2 \cdot \log(N) + 2 \cdot N^2 + N \cdot (N-1) \cdot (N-2) \cdot k^2 \cdot (N \cdot \log(N) + N) + \\ + N^2 + N^2 \cdot k \in O(N^4 \cdot \log(N) \cdot k^2)$$

Comparison with an original algorithm:

Original Minieka algorithm takes only $O(N^3)$ of calculation time. The essential cause of this difference is the absence of cycle check, constituting $N \cdot \log_2(N)$ of the pitch, which is introduced by the author of this presentation.

Occupied memory size:

The algorithm occupies a big part of memory because of the $P_{i,j}$ list, which takes a lot of space - $O(N^3 \cdot k)$. This is because k ways having N peaks defined for each N^2 connection at the same time.

3.2. Combined Depth-first Searches and Dijkstra Algorithm Method

Dijkstra algorithm which helps to find the shortest way in the graph is one of the best known in the graph theory and, because of its efficiency, is considered to be one of the standard algorithms used for searching the way in the graph. This method is not completely suited for the optimum way search. The modification of this method, i.e. when the time limit $D_{i,j}$ is introduced for calculating the way, reduces its efficiency. Therefore the method can be applied only for the calculation of the distance between two peaks.

The so-called depth-first search method is better suited for the estimation of the way, because this is a standard method of finding all the possible ways in the graph. Since it is a limited time method, cycles (contours) are avoided and this makes influence on the expenses. In addition, the superior resistance limit (limited depth-first search) is estimated which makes this method the optimum "limited-time search" solution method.

Starting from the initial point $a \in V$ arcs are checked in all directions, while the shortest route of the search dictates the search direction and peaks are found in turn on the basis of their distance from a. This is how each peak reached to the same peak is rejected.

List $P_{a,v}$, is made for every peak $v \in V$, and the ways found between initial peak $a \in V$ and v are stored in it. Lists are made according to a specified form.

Procedure:

Given data:

Graph $G = (V, E, \gamma)$ with initial peak $a \in V$ $\psi : V \to \{\text{vacant, crossed}\}$- conditional peak marking.

To be searched:

Optimal non-cyclic way from a to v ($v \in V \setminus \{a\}$) with a time limit $D_{a,v}$ and the highest frequency of changing $u_{\max}$. All the ways that are found are included into $P_{a,v}$.

Preparation:

Dijkstra algorithm with the initial peak $a \Rightarrow \forall v \in V \setminus \{a\}$: $d_{a,v}$ is realised;

$\forall v \in V \setminus \{a\}$:

$D_{a,v} := d_{a,v} + \lambda_2$ - interim superior limit;

$\psi(v) :=$ vacant all the peaks given at random.

Basic part:

Depth-first search $(a, 0, 0, 1)$;

Correction of the superior limit:

$\forall v \in V \setminus \{a\}$: $P_{a,v} := \{ k \in P_{a,v} \mid \gamma(k) \le \Gamma(d_{a,v}) \}$

Procedure:

Depth-first search ($i \in V$; $h \in IR^+$; $u, t \in IN$):

i – actual peak;

h – distance between i and a;

u – frequency of changing;

t – number of peaks in the search route.

When $\psi(i) =$ vacant and $h < D_{a,i}$ and $u \le u_{\max}$:

Cycle and limit check. Repetition to be rejected:

$\psi(i) =$ crossed.

Actual peaks to be registered:

$v_t := i$.

The way which is found is the route of the search:

$P_{a,i} := P_{a,i} \cup \{ (v_1, v_2, \ldots, v_t) \}$

Depth-first search :

$\forall e \in E$ with $e = (i, j)$: $(j, h + \gamma(e), u + \mu(e), t + 1)$.

Peaks to be cleared:

$\psi(i) =$ vacant.

Since all the arcs coming out of one peak $v \in V$ are checked, only the ways belonging to the solution set $X^*_{a,V}$, are included into the list $P_{a,V}$:

$$X^*_{a,v} = \{k = (v_1, v_2, ..., v_n) \in K_{a,v} \quad \forall 1 \le x < y \le n : \\ v_x \ne v_y, \gamma(k) \le D_{a,v}, \mu(k) \le u_{\max}\} \quad (10)$$

When $i \in V$ - actual peak and $k = (v_1, v_2, ..., v_t)$ - actual route of the search, $\forall 1 \le x \le 1: v_x \in V$, $v_1 = a$, $v_t = i$, $t \in IN$. k is included into the list $P_{a,i}$, when the initial depth-first search condition is fulfilled:

$$\psi(i) = \text{vacant},\ h < D_{a,j} \text{ and } u \le u_{\max}, \quad (11)$$

Equivalently is also:

$$(\forall 1 \le x < t : v_x \ne 1),\ \gamma(k) \le D_{a,j} \text{ and } \mu(k) \le u_{\max} \quad (12)$$

When every partial sequence $k = (v_1, v_2, ..., v_y)$ of the way k $(1 \le y \le t)$ fulfills the initial condition:

$$\forall 1 \le x \le t : v_x \ne v_y \quad \gamma(k) \le D_{a,j} \text{ and } \mu(k) \le u_{\max} \quad (13)$$

This way $k \in X^*_{a,i}$. k is included into the list, when $k \in X^*_{a,i}$. The rejected search routes are not analysed in the following calculations.

Estimation of expenses:

For the estimation of expenses the condition must be fulfilled:

$$\exists k \in IN : \forall v \in V \setminus \{a\}: |X_{a,v}| \le k \quad (14)$$

From $|P_{a,v}| \le |L_{a,v}|$ follows the final conclusion, that, $|P_{a,v}| \le k$, i.e. the size of the list is limited by k.

Calculating procedure parameters:

Preparation:

- The optimum way search: $\in O(N \cdot \log(N))$;
- $D_{a,v}$ and $\psi(v)$ formation : $2 \cdot N$ pitches;

Depth-first search procedure:

- initial condition (cycle check): 3 pitches;
- actual routes of the search to be copied into the list $P_{a,i}$: t pitches $\in O(N)$
- peaks I to be crossed or cleared: every 1 pitch;
- depth-first search invocation: $\in O(1)$;

The number of peaks checked: $|V| \cdot k = N \cdot k$.

Total expenses of the method:

$O(N \cdot \log(N)) + 2 \cdot N + N \cdot k \cdot (5 + O(N) + O(1)) \in O(N^2 k)$

Total expenses for the search of the way: $\in O(N^3 \cdot k)$

Occupied memory size:

While fixing the algorithm, the $P_{a,v}$ list size is a crucial factor because for all the final peaks $v \in V$ k ways to N peaks are estimated at the same time. The size of memory occupied is equal to $O(N^2 \cdot k)$.

4. Programming Realization the Algorithms of Search of the Optimal Way

For the realization of optimal route searching programme the objective programming language Delphi adjusted for Windows has been used. The connection between program and data is not effective when the data are stored in a comparatively hard disk. As route search algorithms require fast access to the data the program must have a special program in its RAM. The structure of the data reflects ties among network elements.

There are two classes of objects – peaks and arches (Fig 3). Graphs are quite large because stops and transport zones are split into a lot of arches. All the out coming arches are saved as lists. Oriented arches point at final peaks.

The main reason of a long counting period lies in the cycle checking (equal to $O(N \cdot \log(N))$). For the speeding

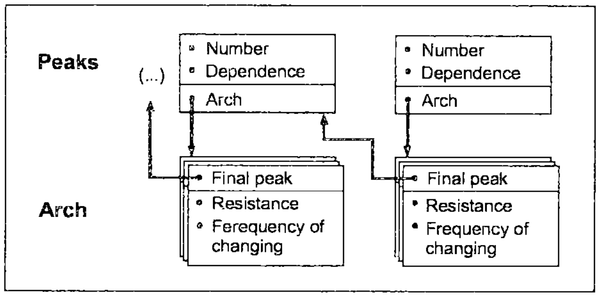

Fig 3. The structure of the graph data

of the process the following feature is used:

- In each graph the way of arrival and leaving peaks exchanges;
- At each leaving station only one arch goes out so in this way the next arrival is known;
- After arrival next leaving belongs to the same stop. All other leaving peaks belong to another stop because they are non – cyclic;
- Recounting peaks one stop leaving station follows the arrival station of the same stop.

All past peaks are listed not along the route, but by numbers. In this way each route description results in a sequence of peaks. The starting and final route stations are not written. In general they hide under a special list where the starting and final peaks are known. In spite of this the second road peak in the sequence of peaks is getting on peak.

According to the made peak sequence it is possible to see how the peaks were passed. Getting – on peak in roads out coming transport regions are always leaving station. The next sequence member from all leaving peaks is found by the only arch. The next arrival peak member in the route is the peak having the next number by its size - arrival station is final (in - between) route peak. In this way the next sequence member belongs to the next stop.

In order to evaluate practically the efficiency and applicability of algorithms the experimental calculation of route search in the model of public transport communication network has been made. Public transport network changed into graph is a real public transport network segment in Kaunas (Table). The calculation data were received from the Transport Problems Institute special group who was responsible for passenger flow research in Kaunas.

Using modifying Minieka algorithms the calculation takes 136 sec., while using combined modified search in depth and Dikstra method it takes only 0.54 sec. The main reason is in the fact that using the second method in any calculation moment only one search route is being analysed and using Dijkstra algorithms actual search tree segment is separated before the search into depth.

Public communication network model

Number of stops	15
Number of transport regions	6
Number of lines	22
Number of segments	26
Number of transport ties	34
Number of total public communication routes	56
Average changing frequency	0.2
Average number of stops in the way	4.3

5. Conclusions

1. For search of the optimal route in public communication network Minieka algorithm is modified and combined search into depth and Dijkstra algorithm method, operating by the basis of graphs is designed.

2. After the analysis of passenger behavior to choose the way resistance function has been made, for optimal way evaluation and comparison. It includes such factors: traveling time, price and comfort (the number of transport change).

3. Both methods might be applied to solve the problem as well as be integrated into the prognostication models of communication processes.

4. Having compared their efficiency (memory size and calculating time) it has been proved that the combined depth-first search and Dijkstra algorithm method are better.

Because of memory size the modified Minieka method is suitable for limited communication network.

References

1. Baublys A. Input of the theory of the transport system. Vilnius: Technika, 1997. 298 p (in Lithuanian).
2. Kristofides N. Graph theory. The algorithmic approach. Moscow: Mir, 1978. 432 p (in Russian).
3. Dievulis G. Method of finding the shortest path in the transport network. *Transportas*, 2000, Vol XV, No 1. Vilnius: Technika, 2000, p 3–11 (in Lithuanian).
4. Goldman A. J., Nemhauser G. J. A transport improvement problem transformable to a best-path problem. *Transportation Science*, Vol 1, No 1, 1967, p 295–307.
5. Nguyen Sang. An algorithm for the traffic assignment problem. *Transportation Science*, 1974, Vol 8, No 3, p 163–192.
6. Magnadi T. L., Wong R. T. Network design and transportation planning models and algorithms. *Transportation Science*, 1984, Vol 18, No 1, p 3–55.
7. Dinic E. Economical algorithms of finding of the shortest paths in the network. *Works of VNIISI*, 1978, No 4, p 36–44 (in Russian).
8. Theune D. Strong and effective methods to solution of problems of path. B. Teubner Verlag, 1995 (in German).
9. *Wermuth M. Representations of model for prediction. Planning of the city traffic – bases, methods, the purposes.* Springer Verlag Heidelberg, 1994, p 221–274 (in German).
10. Palšaitis R. Principles of planning of development of the transport system of Lithuania. *Transportas*, 1995, No 2(11). Vilnius: Technika, 1995, p 30–50 (in Lithuanian).