CYBERNETICS and COMPUTER TECHNOLOGIES

UDC 519.168 DOI:10.34229/2707-451X.25.3.2

V. VASYANIN, O. TROFYMCHUK, L. USHAKOVA

DATA STRUCTURES AND ROUTE REDUCTION PROCEDURES IN THE PROBLEM OF DISTRIBUTION TRANSPORT FLOWS IN A COMMUNICATION NETWORK

In the problems of the distribution and routing of flows in communication networks, the input data is vehicle routes or data transmission channels. For the distribution of flows in this problems in optimization algorithms use special data structures – abstract data types, which describe the relationship between distributed flows and routes, as well as route reduction procedures, which can significantly reduce the number of such connections. The paper develops data structures and route reduction algorithms that allow solving the problem of distribution of flows in the case when the number of specified routes is very large, and the amount of computer RAM is limited. Estimates of the time complexity of the algorithm for reducing routes by nodes and arcs, as well as the algorithm for generating a reference data structure, have been obtained.

Keywords: *multicommodity networks, discrete flows, problems of combinatorial optimization, computer modeling.*

Introduction. In the problems of optimizing the distribution and routing of flows in communication networks, the input data is vehicle routes or data transmission channels [1–7]. In terms of graph theory, a route is a simple path given by the enumeration of nodes and arcs of a network. Circular routes in which the initial and final nodes coincide are allowed. A set of routes can be initially specified or generated according to certain rules in the process of solving a problem. Because routes have different lengths in terms of the number of nodes (or arcs) included in them, shorter routes can coincide with longer routes, both in nodes and in arcs. For the distribution of flows in such problems, special data structures are used – abstract data types (ADT) [8, 9], with the help of which the relationship between distributed flows and routes is described. Route reduction procedures can significantly reduce the number of such connections in the case when the number of initial or generated routes in combinatorial optimization problems is very large.

The article discusses data structures and procedures for route reduction, which are used in algorithms for solving the problem of optimization of distribution and routing of flows [5, 6]. The developed algorithms for route reduction allow solving the problem of distributing flows when the number of specified routes is too many, and the amount of RAM of the computer is limited. Estimates of the complexity of algorithms for reducing routes by nodes and arcs, as well as the algorithm for forming a reference data structure, are obtained. The proposed algorithms were tested on networks with the number of nodes from 50 to 500 and the number of routes from 1225 to 124750, which showed their operability, good computational efficiency and they can be used in practical problems of distribution and routing of flows on large-dimensional networks.

1. Data Structures and Route Reduction Procedures. Let $G(N,P)$ – a multicommodity network with a set of undirected topological arcs P, $p=\left|P\right|$, a set of nodes N, $n=\left|N\right|$. A topological arc is a physical segment of a communication line. Network nodes correspond to

the points of departure, receipt and transshipment (switching) of flows. A matrix of flows is defined on the network $U = \| u_{ij} \|_{n\times n}$, $i, j = \overline{1,n}$. Flows u_{ij} from sources i into the drains j, $i, j = \overline{1,n}$ must be transported in vehicles or transmitted through communication channels. A set of vehicle routes or communication channels is also specified $\{m_k\}$, $k = \overline{1,l}$, each of which consists of a sequence of nodes and topological arcs of the network G, connecting the start and end nodes of a route or communication channel. In a particular optimization problem, the set of routes may not be specified, but searched. For data networks, routes can be represented by simple communication channels connecting adjacent nodes or switched communication channels connecting any sequence of nodes (dedicated channels). In the individual case, all specified routes can coincide with the topological arcs of the network. Plural $\{m_k\}$ can contain multiple routes, nodes connecting any pair. An additional route multi-network is defined $G_M(N, P_M)$, where N – a set of network nodes, P_M – the set of its oriented route arcs. Between any nodes i and j network G_M there is a route arc if they are connected by at least one vehicle route or communication channel with $\{m_k\}$.

As you know, the computational efficiency of algorithms largely depends on the successful representation of the data structures used in the process of solving the problem. When solving the problem of distribution and routing of flows, it is necessary to have such a data structure that, on the one hand, would allow for each flow u_{ij} quickly find the set of routes to which this flow can be allocated, as well as define the set of flows that can be implemented on this route. On the other hand, the data structure should ensure the storage of the results of solving the problem – a flow distribution scheme in which for each u_{ij} defined path from the original i-th node to the end j-th node with directions and transit nodes. Another important requirement for the organization of the data structure is the minimum of information stored in the process of solving the problem.

Let $\upsilon_k = \{\xi_1, \xi_2, ..., \xi_{\eta_k}\}$ – an ordered set of nodes with N on the route m_k, $|\upsilon_k| = \eta_k$, $q_k = \{(\xi_1, \xi_2), (\xi_2, \xi_3), ..., (\xi_{\eta_k - 1}, \xi_{\eta_k})\}$, $|q_k| = \eta_k - 1$ – ordered set of arcs with P, that make up the route m_k.

Since in most cases the forward and return routes tend to be the same, in order to save computer RAM, it seems reasonable to keep only direct routes for coincident routes, bearing in mind that the transport of the flow along such a route is allowed in both directions. If the forward and return routes do not coincide, then each of them must be presented separately.

To distinguish the routes, we will set the sign of the direction of movement along the route. Let $\{m_k\}$, $k = \overline{1,l}$ – the set of routes, taking into account the comments made, and MV_k – indication of the direction of travel along the route, where

$$MV_k = \begin{cases} 0, \text{ if movement is allowed in both directions,} \\ 1, \text{ if movement is allowed in one direction.} \end{cases}$$

Let's also introduce additional features of the routes: $MT_k = \lambda$, where λ – an integer that defines the type of route;

$$MZ_k = \begin{cases} 0, \text{ if movement along the route is allowed,} \\ 1, \text{ if movement is forbidden.} \end{cases}$$

Additional features will be used in the algorithms for route reduction and the formation of a reference data structure.

Let's define the reference structure H_S, consisting of a matrix of pointers $\Lambda = \| \Delta_{ij} \|_{n \times n}$ and elements E, linked in unidirectional linear lists. Index Δ_{ij} corresponds to the flow u_{ij} and points to the first element E, which is called the head of the list. Every element E may have a link to the next item. The last item in the list has a null reference ($NULL$ – link) and is called caudal. Every element E consists of a field FWD, that keeps the reference on the next element or value $NULL$; field AM route, indicating the address to which this flow can be distributed and feature fields DM. The feature field consists of five subfields, which take on the following values:

$$DM_1 = \begin{cases} 1, \text{ if the flow } u_{ij}, i < j \text{ is routed on } AM, \\ 0 \text{ otherwise;} \end{cases}$$

$$DM_2 = \begin{cases} 1, \text{ if the flow } u_{ij}, i > j \text{ is routed on } AM, \\ 0 \text{ otherwise;} \end{cases}$$

$$DM_3 = \begin{cases} 1, \text{ if for } u_{ij} \text{ is fulfilled } (i < j) \wedge (i \prec j), \\ 0, \text{ if for } u_{ij} \text{ is fulfilled } (i < j) \wedge (i \succ j); \end{cases}$$

$$DM_4 = \begin{cases} 1, \text{ if for } u_{ij} \text{ is fulfilled } (i > j) \wedge (i \prec j), \\ 0, \text{ if for } u_{ij} \text{ is fulfilled } (i > j) \wedge (i \succ j); \end{cases}$$

$$DM_5 = \begin{cases} 0, \text{ if the end of the chain of elements } E, \\ \text{defining the routes on which the flow } u_{ij} \\ \text{can be distributed without congestion is reached,} \\ 1 \text{ otherwise.} \end{cases}$$

Signs $i \prec j$, $i \succ j$ respectively imply that the node i precedes the node j, does it follow the node j when driving along the route AM in the order of passing the nodes with υ_{AM}.

Flow u_{ij} let's call it without transit if the following conditions are met for it: $\exists\, m_k \in \{m_k\}$, $k = \overline{1,l}$ for which it is true $(i, j \in \upsilon_k \wedge MV_k = 0) \vee (i, j \in \upsilon_k \wedge MV_k \neq 0 \wedge i \prec j)$. Otherwise, the flow will be called transit. At the initial formation of the structure H_S number of elements E on the list for no transit flow u_{ij} is determined by the number of routes to which the flow can be distributed, and if the flow is symmetrical u_{ji} also without transit, then the corresponding pointers Δ_{ij} and Δ_{ji} get a link to a single list. For transit flows that are to be transshipped one or more times from one route to another, the corresponding pointers from Λ zero links are obtained first.

Building chains of elements E for transit flows u_{ij} is carried out in the process of solving the problem and consists in the following: the routes to which the flow can be distributed are determined; for the found routes, elements are created E, which are linked to the list in accordance with the order of following the routes from i before j. In the information H_S for each transit the route node is written AM in which the flow is congested. Last item E in the chain for the transit flow u_{ij} in the field AY will contain a node j.

In the case when, when solving the problem, branching (splitting) of flows into flow, only one list is remembered, which determines the path that was chosen to distribute this flow during the optimization process, intermediate lists are not remembered. If, in the process of solving the problem, the transit flow is

transferred to the rank of transit, then the list selected for it is associated with the primary one. To differentiate the lists in this case, the subfield DM_5. For transit flows, each element E contains an additional field AY, to which the structure of elements is allowed E additional fields can be introduced to store a portion of the distributed flow.

Fig. 1 shows an example of a structure H_S which explains the concepts introduced. From the rules of initial formation H_S it is clear that the number of elements E in the lists H_S depends on the number of routes and is defined as follows: $E_{HS} = \sum_{k=1}^{l} \eta_k (\eta_k - 1)/2$. Magnitude E_{HS} for real-world networks can be very significant. So, for example, for a network containing 300 nodes when generated for all symmetrical $n(n-1)/2$ flows of the shortest paths of their distribution (individual routes) 44850 routes will be generated. With an average route length η_{cp} = 15 into the structure H_S will be included $44850\eta_{cp}(\eta_{cp} - 1)/2 = 44850 \times 105 = 4709250$ elements that would require about 45 MB of RAM to accommodate (given that to accommodate one item E 10 bytes required).

In this regard, let's consider the reduction procedures that allow you to reduce the dimension of the structure H_S. Let the route m_α reduced to a route m_β by nodes, if the condition is met: $\upsilon_\alpha \cap \upsilon_\beta = \upsilon_\alpha$, and is reduced along topological arcs if $q_\alpha \cap q_\beta = q_\alpha$.

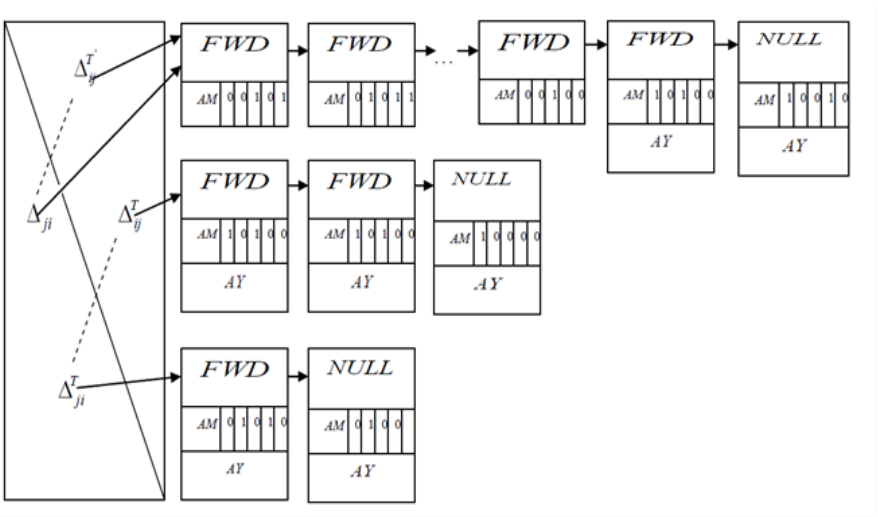


FIG. 1. Where: $\Delta_{ij}^{T'}$ – without transit flow, transferred to the rank of transit; Δ_{ji} – without transit flow; Δ_{ij}^{T}, Δ_{ji}^{T} – symmetrical transit flows

Obviously, route reduction reduces the number of connections in the network G_M and the number of elements in the structure H_S by the amount: $\sum_{\alpha \in \{k^*\}} \eta_\alpha (\eta_\alpha - 1)/2$, where $\{k^*\}$ – the set of numbers of reduced routes.

Let it be further LM_i, $i = \overline{1,l}$ – vector of route lengths by number of nodes VM_i, $i = \overline{1,l}$ – vector of route numbers, ordered in descending order of the number of nodes in the route. In the process of route reduction, we will build a reduction vector VR_i, $i = \overline{1,l}$ and reference vector RV_i, $i = \overline{1,l}$. The elements of vectors are defined as follows:

$$VR_i = \begin{cases} 0, \text{ if the route } i \text{ - th is not reduced to any other rout,} \\ k, \text{ if the route } i \text{ - th is reduced to some route } \xi, \end{cases}$$

where k – route number that has been reduced to a route ξ just before the route i. If i – the first reduced route, then $k = \xi$;

$$RV_i = \begin{cases} 0, \text{ if the route } i \text{ is reduced,} \\ \xi, \text{ if any route is reduced to a route } i, \\ \text{but it itself is not reduced to any other route,} \\ i, \text{ if the route } i \text{ itself is not reduced to any other} \\ \text{and no route is reduced to it,} \end{cases}$$

where ξ – the number of the last route reduced to a route i.

Each vector element $VR_i = 0$, that corresponds to the itinerary m_i determines the chain of other routes reduced to this route. Ordering routes in a chain in the opposite order of their reduction. On the first route in the chain for VR_i indicates the element RV_i.

Let's give a reduction algorithm. Let $S = \| s_{ij} \|_{l \times \eta_{\max}}$ – a matrix of routes, given by enumerating the numbers of network nodes, where $\eta_{\max}$ – the maximum number of nodes in a route. It is assumed that the missing node numbers in shorter routes have been replaced by 0. Let's denote by $BR1$, $BR2$ – working vectors of dimension n. For greater clarity, we will use parentheses in writing the pseudocode of the algorithm for indices of matrices and vectors.

Algorithm 1. Route reduction

1. For $\{ i \mid i = \overline{1,l} \}$ do $RV(i) \leftarrow i$, $VR(i) \leftarrow 0$.

2. If a homogeneous reduction is chosen, then $RT \leftarrow 'O'$, for mixed reduction $RT \leftarrow 'M'$. *** For homogeneous reduction, route types $MT_k = \lambda$ match.

3. If node reduction is chosen, then $PRT \leftarrow 'U'$, for arc reduction $PRT \leftarrow 'A'$.

4. If $PRT = 'U'$, then go to step 5, otherwise go to step 25.

5. For $\{ i \mid i = \overline{1,l-1} \}$ do step 6-24. *** Start of the cycle by i reduction by nodes.

6. $K1 \leftarrow VM(i)$.

7. If $VR(VM(i)) = 0$, then go to step 8, otherwise to step 24.

8. $BR1 \leftarrow 0$.

9. For $\{ k \mid k = \overline{1, LM(VM(i))} \}$ do $BR1(S(VM(i),k)) \leftarrow 1$.

10. For $\{ j \mid j = \overline{i+1,l} \}$ do step 11-23. *** Start of the cycle by j.

11. If $VR(VM(j)) = 0$, then go to step 12, otherwise to step 23.

12. $BR2 \leftarrow 0$.

13. For $\{ k \mid k = \overline{1, LM(VM(j))} \}$ do $BR2(S(VM(j),k)) \leftarrow 1$.

14. If $RT = 'M' \vee RT = 'O' \wedge MT(VM(i)) = MT(VM(j))$, then go to step 15, otherwise to step 23.

15. For $\{ k \mid k = \overline{1,n} \}$ do step 16-18. *** Start of the cycle by k.
16. If $BR2(k) = 1$, then go to step 17, otherwise to step 18.
17. If $BR1(k) = 1$, then go to step 18, otherwise to step 23.
18. Go to step 15.*** End of cycle by k.
19. $RV(VM(j)) \leftarrow 0$.
20. $RV(VM(i)) \leftarrow VM(j)$.
21. $VR(VM(j)) \leftarrow K1$.
22. $K1 \leftarrow VM(j)$.
23. Go to step 10.*** End of cycle by j.
24. Go to step 5.*** End of cycle by i.
25. For $\{ i \mid i = \overline{1,l-1} \}$ do step 26–55. *** Start of the cycle by i arc reduction.
26. $KS \leftarrow VM(i)$.
27. If $VR(VM(i)) \neq 0$, then go to step 55, otherwise to step 28.
28. For $\{ j \mid j = \overline{i+1,l} \}$ do step 29–54. *** Start of the cycle by j.
29. If $VR(VM(j)) \neq 0$, then go to step 54, otherwise to step 30.
30. $KL \leftarrow 0$; $KF \leftarrow 1$.
31. For $\{ k1 \mid k1 = \overline{1, LM(VM(i))} \}$ do step 32–34.
32. If $S(VM(j), KF) = S(VM(i), k1)$, then go to step 36.
33. If $S(VM(j), LM(VM(j)) = S(VM(i), k1)$, then go to step 48.
34. Go to step 31.
35. Go to step 54, continue the cycle j.
36. For $\{ k \mid k = \overline{1, LM(VM(j))} \}$ do step 37–40.
37. If $S(VM(j), k) = S(VM(i), k1)$, then go to step 38, otherwise to step 40.
38. $KL \leftarrow KL + 1$; $k1 \leftarrow k1 + 1$.
39. If $k1 > LM(VM(i))$, then go to step 41.
40. Go to step 36.
41. If $RT = 'M' \vee RT = 'O' \wedge MT(VM(i)) = MT(VM(j))$, then go to step 42, otherwise to step 47.
42. If $KL = LM(VM(j))$, then go to step 43, otherwise to step 47.
43. $RV(VM(j)) \leftarrow 0$.
44. $RV(VM(i)) \leftarrow VM(j)$.
45. $VR(VM(j)) \leftarrow KS$.
46. $KS \leftarrow VM(j)$.
47. Go to step 54, continue the cycle j.
48. For $\{ k \mid k = \overline{LM(VM(j)), 1, -1} \}$ do step 49–52.
49. If $S(VM(j), k) = S(VM(i), k1)$, then go to step 50, otherwise to step 52.
50. $KL \leftarrow KL + 1$; $k1 \leftarrow k1 + 1$.

51. If $k1 > LM(VM(i))$, then go to step 41.
52. Go to step 48.
53. Go to step 41.
54. Go to step 28.*** End of cycle by j.
55. Go to step 25.*** End of cycle by i.
56. Stop.

As a result of the work of the reduced algorithm, chains in the reduction vector for $VR_i = 0$, the beginning of which is indicated by RV_i, will be ordered by increasing number of nodes in reduced routes. After the reduction procedure is completed, a reference structure is built. At the same time, only those routes are visible, in which $VR_i = 0$. It is obvious that after performing the reduction procedure, the amount of RAM of the computer occupied by the reference H_S will be reduced by the amount of: $\Delta = \left(\sum_{k=1}^{l} \eta_k (\eta_k - 1)/2 - \sum_{\alpha \in \{k^*\}} \eta_\alpha (\eta_\alpha - 1)/2 \right) V_E$, where V_E – the amount of memory occupied by a single list item.

Algorithm 1 shows that the time complexity of reduction by nodes is on average $O(l^2(n + \eta_{cp}) + l\eta_{cp})$, and in arcs – $O(l^2 \eta_{cp})$ operations, where η_{cp} – average route length. It should be noted that the reduction of routes will slightly increase the complexity of the main optimization algorithm, since the number of routes viewed in the steps for selecting routes for the distribution of flows will increase. Nevertheless, route reduction is essential if the initial set of routes is all possible routes. In this case, the dimensionality of the problem increases sharply, taking on the character of constructing new, previously non-existent routes.

Let's consider the algorithm of the initial formation of the reference H_S with simultaneous construction of route arcs of the network G_M. Let $R = \| r_{ij} \|_{n \times n}$ – topological matrix of the network G, r_{ij} – arc length p_{ij} (if the arcs p_{ij} does not exist $r_{ij} = 0$); $D = \| d_{ij} \|_{n \times n}$ – matrix of route arc lengths (originally $d_{ij} = \infty$, $i, j = \overline{1,n}$). Since any two nodes can be connected by more than one route, the lengths of the arcs d_{ij} we will determine from the following expression: $d_{ij} = \min\limits_{\{\upsilon_k\}} (r_{\xi_\mu, \xi_{\mu+1}} + r_{\xi_{\mu+1}, \xi_{\mu+2}} + ... + r_{\xi_{\mu+c-1}, \xi_{\mu+c}})$, at $\xi_\mu = i$, $\xi_{\mu+c} = j$, $k = \overline{1,l}$. In the description of the algorithm, the record «Form a New Element $E(P)$» means the generation of a new element E and pointer installation P to the address of the item. References to the element's fields will be written in the form $P.\langle \text{field name} \rangle$, and setting the pointer in the form of $P \Rightarrow$.

Algorithm 2. Formation H_S

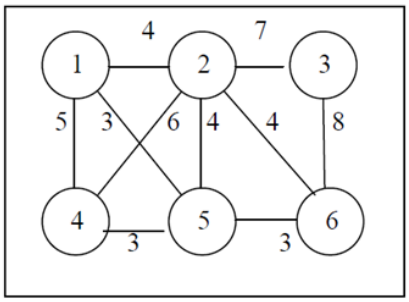
1. For $\{ i \mid i = \overline{l,1,-1} \}$ do step 2–18. *** Start of the cycle by i formation H_S.
2. If $VR(VM(i)) = 0 \wedge MZ(i) = 0$, then go to step 3, otherwise to step 18.
3. For $\{ k \mid k = \overline{1, LM(VM(i)) - 1} \}$ do step 4–17.***Start of the cycle by k.
4. For $\{ j \mid j = \overline{k+1, LM(VM(i))} \}$ do step 5–16.***Start of the cycle by j.

5. Form a New Element $E(P)$.
6. If $\Lambda(S(VM(i),k),S(VM(i),j)).FWD \neq NULL$, then go to step 7, otherwise to step 12.
7. $P1 \Rightarrow \Lambda(S(VM(i),k),S(VM(i),j)).FWD$.
8. While $P1.FWD \neq NULL$ do step 9–10.
9. $P1 \Rightarrow P1.FWD$.
10. Go to step 8.
11. $P1.FWD \Rightarrow P$.Go to step 14.
12. $\Lambda(S(VM(i),k),S(VM(i),j)).FWD \Rightarrow P$.
13. $\Lambda(S(VM(i),j),S(VM(i),k)).FWD \Rightarrow P$. Go to step 14.
14. $P.FWD \Rightarrow NULL$; $P.AM \leftarrow VM(i)$; $P.DM \leftarrow 0\{\, i \mid i=\overline{1,l}\,\}$; $P.DM(5) \leftarrow 1$.
15. If $S(VM(i),k) < S(VM(i),j))$, then $P.DM(3) \leftarrow 1$, otherwise $P.DM(4) \leftarrow 1$.
16. Go to step 4.*** End of cycle by j.
17. Go to step 3.*** End of cycle by k.
18. Go to step 1.***End of cycle by i.
19. $D \leftarrow \infty$.
20. For $\{\, i \mid i=\overline{1,l}\,\}$ do step 21–30. *** Start of the cycle by i formation D.
21. For $\{\, k \mid k=\overline{1,LM(i)-1}\,\}$ do step 22–29. *** Start of the cycle by k.
22. $SUM \leftarrow 0$.
23. For $\{\, j \mid j=\overline{k+1,LM(i)}\,\}$ do step 24–28. *** Start of the cycle by j.
24. $m \leftarrow j-1$.
25. $SUM \leftarrow SUM + R(S(i,m),S(i,j))$.
26. If $D(S(i,k),S(i,j)) > SUM$, then $D(S(i,k),S(i,j)) = SUM$.
27. If $MV(i) = 0 \wedge D(S(i,j),S(i,k)) > SUM$, then $D(S(i,j),S(i,k)) = SUM$.
28. Go to step 23.*** End of cycle by j.
29. Go to step 21.*** End of cycle by k.
30. Go to step 20.*** End of cycle by i.
31. Stop.

How easy it is to see the average labor intensity of the Algorithm 2 is $O((l-lr)\eta_{cp}^2)$ operations, where lr – the number of reduced routes, and η_{cp} – average route length. For the initial distribution of flows to networks G_M an algorithm for constructing the shortest paths is used according to the criterion: minimum transit congestion; if the number of overloads is equal, the minimum length of the path [10].

2. An example of the formation of a reference structure

Let a given network with the number of nodes $n=6$, number of arcs $p=10$ and a matrix of arc lengths R, which is shown in the Fig. 2. After reducing the specified routes by nodes (U) and arcs (A) the results shown in the Tabl. 1, 2.



R						
N	1	2	3	4	5	6
1	**0**	**4**	**0**	**5**	**3**	**0**
2	**4**	**0**	**7**	**6**	**4**	**4**
3	**0**	**7**	**0**	**0**	**0**	**8**
4	**5**	**6**	**0**	**0**	**3**	**0**
5	**3**	**4**	**0**	**3**	**0**	**3**
6	**0**	**4**	**8**	**0**	**3**	**0**

FIG. 2. Specified network and arc length matrix

TABLE 1. Reduction results

Options	Reduction	
	By nodes U	By arcs A
Number of nodes in the network	6	6
Number of arcs in the network	10	10
Number of routes in the network	14	14
Number of reduced routes, incl. in %	13, 92 %	6, 42 %
Number of elements in the ADT structure	72	72
Number of reduced elements, incl. in %	57, 79 %	17, 23 %

TABLE 2. The results are obtained after the reduction of the specified routes by nodes (U) and arcs (A)

No.	$S = \| s_{ij} \|_{14\times6}$						LM_{14}	VM_{14}	VR_{14}	RV_{14}	VR_{14}	RV_{14}
									U		A	
1	1	2	3	6	5	4	6	1	0	6	0	13
2	1	2	3	6	0	0	4	9	4	0	1	0
3	1	2	6	5	0	0	4	3	9	0	0	3
4	1	2	5	6	0	0	4	4	3	0	0	10
5	1	2	4	0	0	0	3	2	14	0	0	5
6	1	4	0	0	0	0	2	12	13	0	8	0
7	2	4	5	0	0	0	3	14	11	0	9	0
8	2	4	1	0	0	0	3	5	7	0	0	6
9	2	4	5	6	3	0	5	10	1	0	0	7
10	2	5	6	0	0	0	3	11	5	0	4	0
11	3	2	1	0	0	0	3	7	10	0	2	0
12	3	2	5	6	0	0	4	8	2	0	0	12
13	3	6	0	0	0	0	2	13	8	0	11	0
14	4	5	2	6	0	0	4	6	12	0	0	14

Fig. 3 excerpts from the reference structure H_S for flows u_{12} and u_{21} without route reduction and with reduction by nodes and arcs. When distributing flows in the first case, the routes will be visible in the following order – 8, 11, 5, 2, 4, 3, 1; in the second – 6, 13, 8, 7, 11, 10, 5, 14, 12, 2, 4, 3, 9, 1; in the third – 8, 6, 5, 4, 10, 3, 1, 13, 11, 2, 1. Thus, as already noted, after the reduction of routes in the main optimization algorithm, the total number of route views increases, which accordingly leads to an increase in its labor

intensity. In addition, in the case of arc reduction, the revision of routes by increasing the number of nodes may be disrupted if the same flow can pass along paths with different arcs. In our example, these are both flows – u_{12} and u_{21} (routes 1, 2, 3, 4, 5, 11 and route 8).

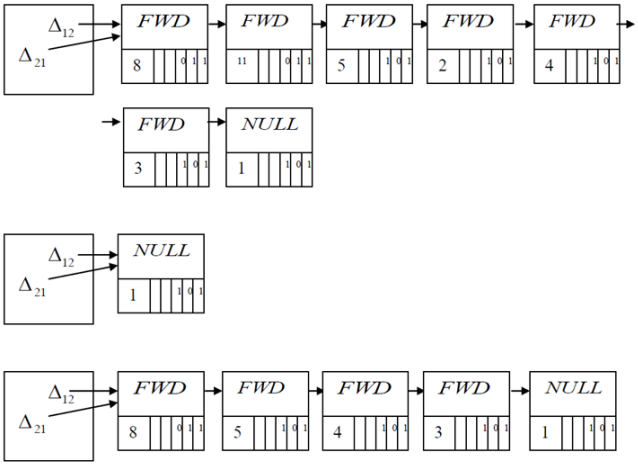


FIG. 3. Fragments of the reference structure H_S for flows u_{12} and u_{21} without route reduction and with reduction by nodes and arcs

Tabl. 3 shows a matrix of shortest path lengths D and reference matrix of built paths $C = \| c_{ij} \|_{n \times n}$, each element of which c_{ij}, $i \neq j$ determines the number of the penultimate node on the shortest path from i before j, $c_{ii} = 0$, $i = \overline{1,n}$. Matrices obtained for the network G_M algorithm [10].

TABLE 3. Shortest path length matrix D and reference matrix of built paths C

D						
N	1	2	3	4	5	6
1	**0**	**4**	**11**	**5**	**8**	**8**
2	**4**	**0**	**7**	**6**	**4**	**4**
3	**11**	**7**	**0**	**14**	**11**	**8**
4	**5**	**6**	**14**	**0**	**3**	**6**
5	**8**	**4**	**11**	**3**	**0**	**3**
6	**8**	**4**	**8**	**6**	**3**	**0**

C						
N	1	2	3	4	5	6
1	**0**	**1**	**1**	**1**	**1**	**1**
2	**2**	**0**	**2**	**2**	**2**	**2**
3	**3**	**3**	**0**	**3**	**3**	**3**
4	**4**	**4**	**4**	**0**	**4**	**4**
5	**5**	**5**	**5**	**5**	**0**	**5**
6	**6**	**6**	**6**	**6**	**6**	**0**

3. Results of the computational experiment

To test the performance and computational efficiency of the algorithms, a computational experiment was conducted on a PC with a clock frequency of 2.66 GHz and 2 GB of RAM running the Windows 10 operating system. The algorithms were tested on networks containing 50 to 500 nodes. For the entire dimension, the pseudorandom number sensor generated homogeneous networks with the number of output arcs from each node (a measure of the nodes) val = 5 and topological arc lengths ranging from 80 to 320. The results of the calculations are given in the Tabl. 4 and Fig. 4–7. Where lr and $lr\%$, E, Er and $Er\%$, $TFLOYD$, $TRED,U$, $TRED,A$, $TFORM$, $TVVSP$ – respectively, the number and percentage of reduced

routes; the number of elements generated and the number and percentage of reduced elements E in the structure H_S; time to build the shortest paths according to the criterion – the minimum length of the path on the network G Floyd's algorithm [8, 9] to generate routes for all $n^2 - n$ flows; time of reduction of routes by nodes and arcs; time of formation of the structure H_S; the time of constructing the shortest paths by the algorithm [10] according to the criterion – minimum transit overloads, if the number of overloads is equal – the minimum length of the path on the network G_M.

TABLE 4. Results of a computational experiment on networks containing 50 to 500 nodes

n	50	100	150	200	250	300	400	500
val	5	5	5	5	5	5	5	5
p	125	250	375	500	625	750	1000	1250
l	1225	4950	11175	19900	31125	44850	79800	124750
$\eta_{\max}$	7	9	10	13	11	12	13	14
η_{cp}	3	4	5	5	6	6	6	6
lr *lr* %	789, 64 %	3382, 68 %	7764, 69 %	14165, 71 %	22102, 71 %	31336, 69 %	55682, 69 %	88619, 71 %
E	7797	48961	138596	301079	510913	786289	1618257	2822650
Er *Er* %	3904, 50 %	27196, 55 %	79606, 57 %	181025, 60 %	308204, 60 %	467208, 59 %	967801, 59 %	1743565, 61 %
TFLOYD	0,02	0,02	0,05	0,08	0,16	0,28	0,64	1,20
TRED,U *TRED, A*	0,02 0,00	0,28 0,16	2,12 0,61	6,32 1,81	17,88 4,35	53,38 9,47	184,81 31,47	546,85 85,47
TFORM	0,00	0,02	0,03	0,06	0,11	0,27	1,11	2,65
TVVSP	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,00

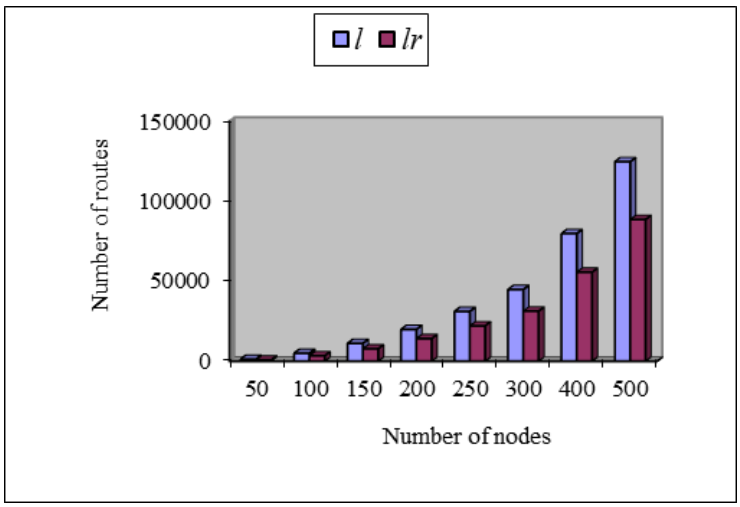


FIG. 4. Number of routes

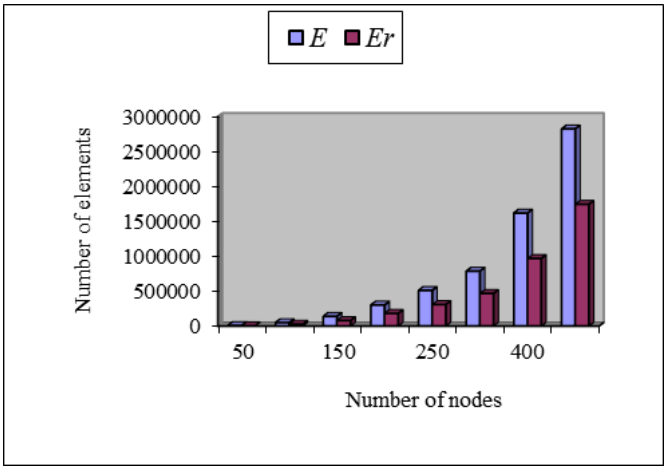


FIG. 5. Number of elements

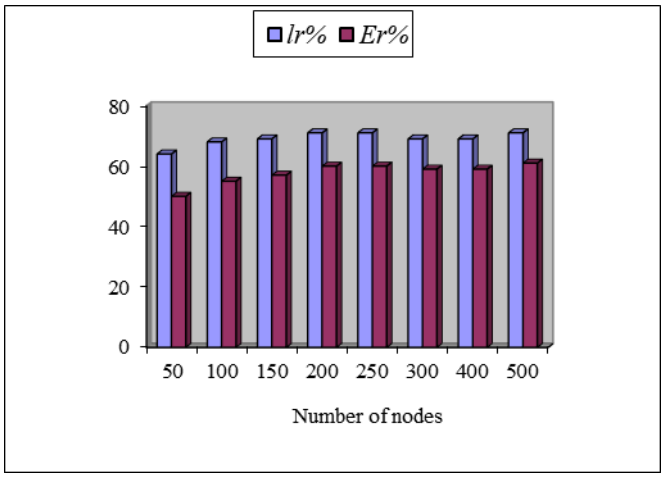


FIG. 6. Number of routes and elements in %

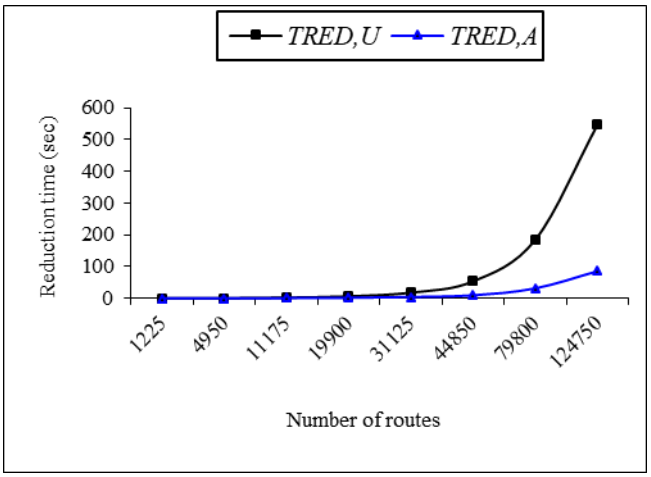


FIG. 7. Reduction time (sec)

Conclusion

1. The paper develops data structures and algorithms for route reduction, which allow solving the problem of distribution and routing of flows in the case when the number of specified routes is too many, and the amount of RAM of the computer is limited. Data structures based on linear dynamic lists are proposed, with the help of which you can quickly find all possible routes for distributing flows in the network. Estimates of the labor intensity of algorithms for reducing routes by nodes have been obtained – $O(l^2(n+\eta_{cp})+l\eta_{cp})$ and arcs – $O(l^2\eta_{cp})$, as well as an algorithm for the formation of a reference data structure – $O((l-lr)\eta_{cp}^2)$, where l and n – the number of routes (the number of specified communication lines) and nodes in the network, η_{cp} – average route length (number of nodes in the communication line), lr – number of reduced routes (number of reduced communication lines). The proposed algorithms were tested on networks with the number of nodes from 50 to 500 and the number of routes from 1225 to 124750, which showed their efficiency and good computational efficiency, and they can be used in practical problems of distribution and routing of flows on large-dimensional networks.

2. On the basis of the proposed data structures and heuristic approaches, polynomial algorithms and a computer program have been developed [11] for solving the NP-hard [12] problem of distribution and routing of flows of transport blocks. Estimates of the time complexity of algorithms are obtained. Algorithms allow you to get an approximate solution to the problem in time

$$T_o = O(n^3) + O(l^2(n+\eta_{cp})+l\eta_{cp}) \text{ (or } O(l^2\eta_{cp})+O((l-lr)\eta_{cp}^2)+ \gamma\,[\,O(n_g n^{4+\varepsilon})+ O[\lambda(e(e+\alpha)+Ke(e+\alpha))+e\log e+K_1 e]\,],$$

where γ – number of iterations of solving the problem of selecting arc capacities [13], n_g – the average number of parts in a branched flow, e – the number of arcs in the network, α – the number of different discrete bandwidths of the network arcs, λ, K, K_1 – some constants, $\varepsilon \in [0, 0.3]$.

Authorship contribution: Volodymyr Vasyanin & Oleksandr Trofymchuk – Research and development of the article, Formulated the research problem, Writing the manuscript, Proposing future research directions. Liudmyla Ushakova – Research and development of the article, Conducted analyses of get results, Writing the manuscript.

Funding. The authors did not receive any funding for conducting research or writing the article.

References

1. Trofymchuk O.M., Vasyanin V.A. Simulation of Packing, Distribution and Routing of Small-Size Discrete Flows in a Multicommodity Network. *Journal of Automation and Information Sciences*. 2015. **47** (7). P. 15–30. https://doi.org/10.1615/JAutomatInfScien.v47.i7.30
2. Trofymchuk O.M., Vasyanin V.A., Kuzmenko V.N. Optimization Algorithms for Packing of Small-Lot Correspondence in Communication Networks. *Cybernetics and Systems Analysis*. 2016. **52** (2). P. 258–268. https://doi.org/10.1007/s10559-016-9822-5
3. Trofimchuk A.N., Vasyanin V.A. Computer Modeling of the Hierarchical Structure of a Communication Network with Discrete Multiproduct Flows. *USiM*. 2016. No. 2. P. 48–57. (in Russian) https://doi.org/10.15407/usim.2016.02.048
4. Trofimchuk A.N., Vasyanin V.A., Ushakova L.P. Study of the Problem of Optimizing the Hierarchical Structure of a Sparse and Dense Communication Network. *Problems of Control and Informatics*. 2021. No. 1. P. 5–21. (in Russian) https://doi.org/10.34229/1028-0979-2021-1-1
5. Vasyanin V.A. Problem of Distribution and Routing of Transport Blocks with Mixed Attachments and Its Decomposition. *Journal of Automation and Information Sciences*. 2015. Vol. 47, Issue 2. P. 56–69. https://doi.org/10.1615/JAutomatInfScien.v47.i2.60

6. Vasyanin V.A. Computer Modeling of Distribution and Routing of Discrete Multiproduct Flows in a Communication Network. *USiM*. 2016. No. 3. P. 43–53. (in Russian) https://doi.org/10.15407/usim.2016.03.043
7. Vasyanin V.A., Trofymchuk O.M., Ushakova L.P. Problem of Groupage Cargo Routing in the Multicommodity Transport Network with Given Tariffs and Delivery Time Constraints. *Cybern Syst Anal*. 2022. 58. P. 966–976. https://rdcu.be/c2Vh9; https://doi.org/10.1007/s10559-023-00531-z
8. Cormen T., Leiserson C., Rivest R., Stein C. Introduction to Algorithms, 2nd Edition. Cambridg: McGraw-Hill Book Company, 2001. 1296 p.
9. Sedgwick R.. Fundamental algorithms in C++. Algorithms on graphs. St.Pt.: DiaSoftUP LLC. 2002. 496 p. (in Russian)
10. Vasyanin V.A. A Two-Criterion Lexicographic Algorithm for Finding All Shortest Paths in Networks. *Cybernetics and Systems Analysis*. 2014. **50 (**5). P. 759–767. https://doi.org/10.1007/s10559-014-9666-9
11. Certificate of copyright registration for the work "Computer Program for Optimization of Distribution and Routing of Discrete Flows in a Multi-Product Communication Network" / applicant and owner V.O. Vasyanin; A. S. dated 20.07.2016. No. 66794, State Intellectual Property Service of Ukraine; application dated 25.05.2016. No. 67224 on registration of copyright for the work. (in Ukrainian)
12. Garey M.R., Johnson D.S. Computers and Intractability: A Guide to the Theory of NP-Completeness. San Francisco: W.H. Freeman and Company, 1979. 338 p.
13. Trofymchuk O.M., Vasyanin V.A. Choosing the Capacity of Arcs with Constraint on Flow Delay Time. *Cybernetics and Systems Analysis*. 2019. **55** (4). P. 561–569. https://doi.org/10.1007/s10559-019-00165-0

Received 06.05.2025

Vasyanin Volodymyr,
Doctor of Technical Sciences, Chief of department,
Institute of Telecommunications and Global Information Space of the NAS of Ukraine, Kyiv,
https://orcid.org/0000-0003-4046-5243
archukr@meta.ua

Trofymchuk Oleksandr,
Corresponding Member of the National Academy of Sciences of Ukraine, Professor, Doctor of Technical Sciences,
Institute of Telecommunications and Global Information Space of the NAS of Ukraine, Kyiv,
https://orcid.org/0000-0003-3358-6274
itgis@nas.gov.ua

Liudmyla Ushakova,
Leading engineer,
Institute of Telecommunications and Global Information Space of the NAS of Ukraine, Kyiv.
https://orcid.org/0000-0002-9020-1329
archukr@i.ua

УДК 519.168

В. Васянін [*], О. Трофимчук, Л. Ушакова

Структури даних і процедури редукції маршрутів у задачі розподілу транспортних потоків у комунікаційній мережі

Інститут телекомунікацій і глобального інформаційного простору НАН України, Київ
[*] Листування: archukr@meta.ua

Вступ. У задачах розподілу і маршрутизації потоків у комунікаційних мережах вхідними даними виступають маршрути транспортних засобів або канали передачі даних. Для розподілу потоків у таких задачах в алгоритмах оптимізації використовуються спеціальні структури даних – абстрактні типи даних, за допомогою яких описується зв'язок між розподілюваними потоками і маршрутами, а також процедури редукції маршрутів, які дозволяють значно скоротити кількість таких зв'язків. У роботі розроблені структури даних і алгоритми редукції маршрутів, що дозволяють вирішувати задачу розподілу потоків у разі, коли кількість заданих маршрутів дуже велика, а обсяг оперативної пам'яті комп'ютера обмежений. Отримані оцінки часової складності алгоритму редукції маршрутів по вузлам і дугам, а також алгоритму формування довідкової структури даних. Проведено тестування запропонованих алгоритмів на мережах

з кількістю вузлів від 50 до 500 і кількістю маршрутів від 1225 до 124750, яке показало їх працездатність, хорошу обчислювальну ефективність і вони можуть бути використані в практичних задачах розподілу і маршрутизації потоків на мережах великої розмірності.

Мета. Наведено ефективний алгоритм скорочення маршрутів, що дозволяє вирішити задачу розподілу потоків у випадку, коли кількість заданих маршрутів дуже велика, а обсяг оперативної пам'яті комп'ютера обмежений.

Методика. Для отримання оцінок часової складності розроблених алгоритмів ми використовуємо числові приклади та моделювання.

Результати. Показано, що запропоновані алгоритмі мають достатню точність і швидкодію, що дозволяє стверджувати їх практичну застосовність для інженерних розрахунків на великих масштабних мережах.

Наукова новизна і практична значимість. Ми демонструємо надійність та ефективність запропонованих алгоритмів за допомогою ретельного комп'ютерного моделювання.

Ключові слова: багатопродуктові мережі, дискретні потоки, задачі комбінаторної оптимізації, комп’ютерне моделювання.