Methodology of Searching for All Shortest Routes in NoC

Nadezhda Matveeva, Elena Suvorova
Saint-Petersburg State University of Aerospace Instrumentation
Saint-Petersburg, Russia
n.matveeva88@gmail.com, wildcat15@yandex.ru

Abstract

Nowadays network technologies are broadly used in different areas of a human life. Data streams between network nodes are dramatically increasing. Systems-on-chip are evaluating very quickly and are used in many different areas of human life such as space, medicine and so on. Requirements to modern network hardware are increasing constantly. These requirements are: little size, big performance, many supported features and so on. Network-on-chip (NoC) plays very important role in a system-on-chip operation. It connects together all IP blocks, nodes of a system and provides data transmission. Therefore proper design of NoC increases its performance. Many modern NoCs support the adaptive routing mode. This mode assumes usage of several routes for specified pair source - destination. To improve network performance and decrease delays these routes must be the shortest. In this paper we offer an algorithm producing set of all shortest routes between source and destination for load balancing feature and performance increasing. In this paper the algorithm of the shortest noncyclical equal-length routes set building is offered.

Index Terms: Embedded networks, Network-on-chip (NoC), Adaptive routing, Shortest route, Network.

I. Introduction

Many modern NoCs use adaptive routing feature. This means, that routers have information about possible routes of data transmission to a destination and distribute traffic uniformly to increase network performance. Exploration of algorithms and methods which can be embedded into routers hardware in order to provide functioning and load distribution are out of scope of this paper. The fact of alternative routers usage is important. Adaptive routing is used in image/video/audio processing systems and in systems, collecting data from different sensors.

Usually the main requirement to alternative routes is data transfer time. Thus, these routes should have same length. That’s because we payed attention to the problem of searching all different shortest paths (with same length) without loops.

II. Methodology of Searching for All Shortest Routes

For NoCs supporting adaptive routing it is necessary to have information about set of possible data transmission routes. The methodology described in this paper is used for building shortest routes set. All shortest routes have equal length and does not contain loops.

The terms used in this paper are listed below:

- Information data/transmission route - a list of nodes through which data/information transmission occurs for specified pair source - destination.

- Length of data/information transmission route - number of routers/switches through which data transmission occurs for specified pair source - destination.
- Adjacent nodes - two nodes are adjacent if they are end nodes of a one edge.
- The shortest route - a route between two nodes such that the number of its constituent edges is minimized.
- Achievable node - it means, that there is a route to this node.

There are many different algorithms providing ways to find one shortest route or shortest routes between all graph nodes. Deikstra, Bellman - Ford and Lee algorithms [1] are well known and can be used to build only one shortest path between pair of nodes. Other different topologies such as torus, hypercube and grid are very common in practice. For such topologies the existence of many shortest equal routes is typical.

Topologies with several equal-length routes are typical for NoC. That is why it is important to take into account all routes while building routes list. This is essential for uniform distribution of data streams and increase of network performance.

The proposed methodology is based on the Lee's algorithm. The following rules are used to build routes list between specified pair source - destination:

- The shortest route is selected for data transmission.
- If there are several equal-length routes between source and destination then they all can be used with uniform data streams distribution.

Suggested algorithm for building a list of shortest data transmission routes is shown in Fig. 1.

NoC is originally presented in the form of graph, where nodes represent nodes of a system (switches, hosts and so on) and edges - links between these nodes. Source and destination are specified further. The following states are acceptable for a node: "not marked", "marked" and "watched". Originally all nodes have "not marked" state. Two lists OldFront and NewFront are used to search for shortest routes. Nodes for which search for adjacent nodes is required are put into OldFront list. Adjacent nodes for nodes from OldFront list are put into NewFront list. The only node "source" is put into OldFront list on the first iteration and it is marked as "marked". Further while OldFront list not empty (this means that destination is unachievable) or destination node is not put into it (all shortest routes are found) the following operations should be performed:

- For every node from OldFront list search is performed to find all adjacent nodes from “not marked” and “marked” set which are put into NewFront list. References to achievable nodes are formed.
- All nodes from OldFront list are marked as “watched”.
- OldFront list is cleared.
- All nodes from NewFront list are marked as “marked”.
- Nodes from NewFront list are copied into OldFront list. OldFront list is cleared.

If destination is achievable then it is possible to build list of shortest routes using references to nodes.

The proposed idea of graph nodes marking allows to watch nodes for routes searching only once. This provides absence of loops in data transmission routes being searched. Storing of references and links between nodes during processing allows to restore shortest routes found unambiguously.

The check of presence of destination node in OldFront list on every iteration allows to avoid searching for routes with length greater then the length of the shortest path. This gives an opportunity to decrease the processing time in irregular network topologies.

A. Example

Let us consider the following example to clarify the proposed algorithm. As the topology, where it is necessary to distribute traffic uniformly for all the shortest routes of equal length, we consider mesh [2]. This network topology is used in the most modern NoCs [3]. It is shown in Fig. 2. The abbreviations used in this paper are listed below:

- TN - terminal node, source or destination of data.
- SW - switch, communication element.

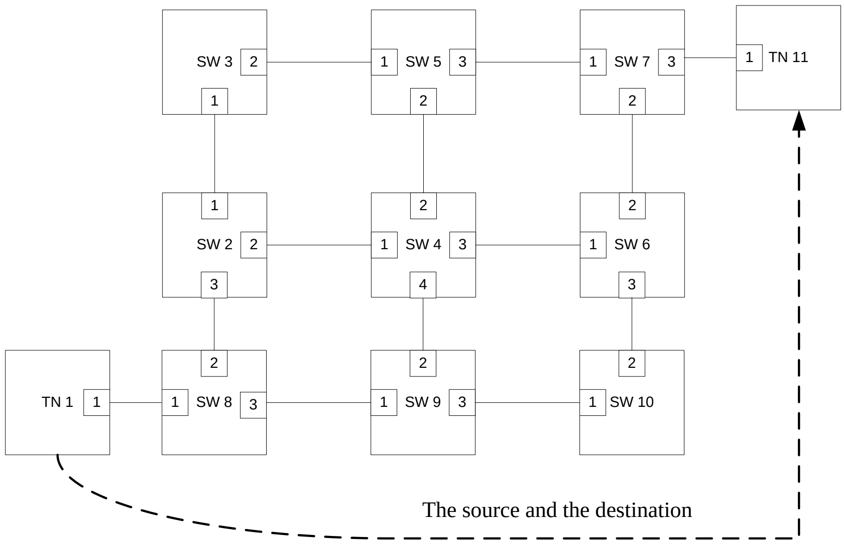

Fig. 1. Mesh network topology

Shortest routes are created in accordance with the algorithm above. Acceptable states for a node are shown in Fig. 3.

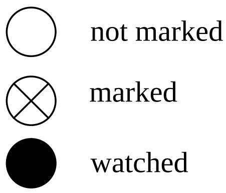

Fig. 2. The following states are acceptable for a node

At the beginning all the nodes have "not marked" status. A node "source" is put into OldFront list on the first iteration and it is marked as "marked" (Fig. 4). For every node from OldFront list search is processed to find all adjacent nodes from “not marked” and “marked” sets which are to be put into NewFront list. Each time an achievable node is found the reference for it is formed.

All adjacent nodes are marked as “marked”. Node for which search for adjacent nodes was performed is marked as "watched" (Fig. 5).

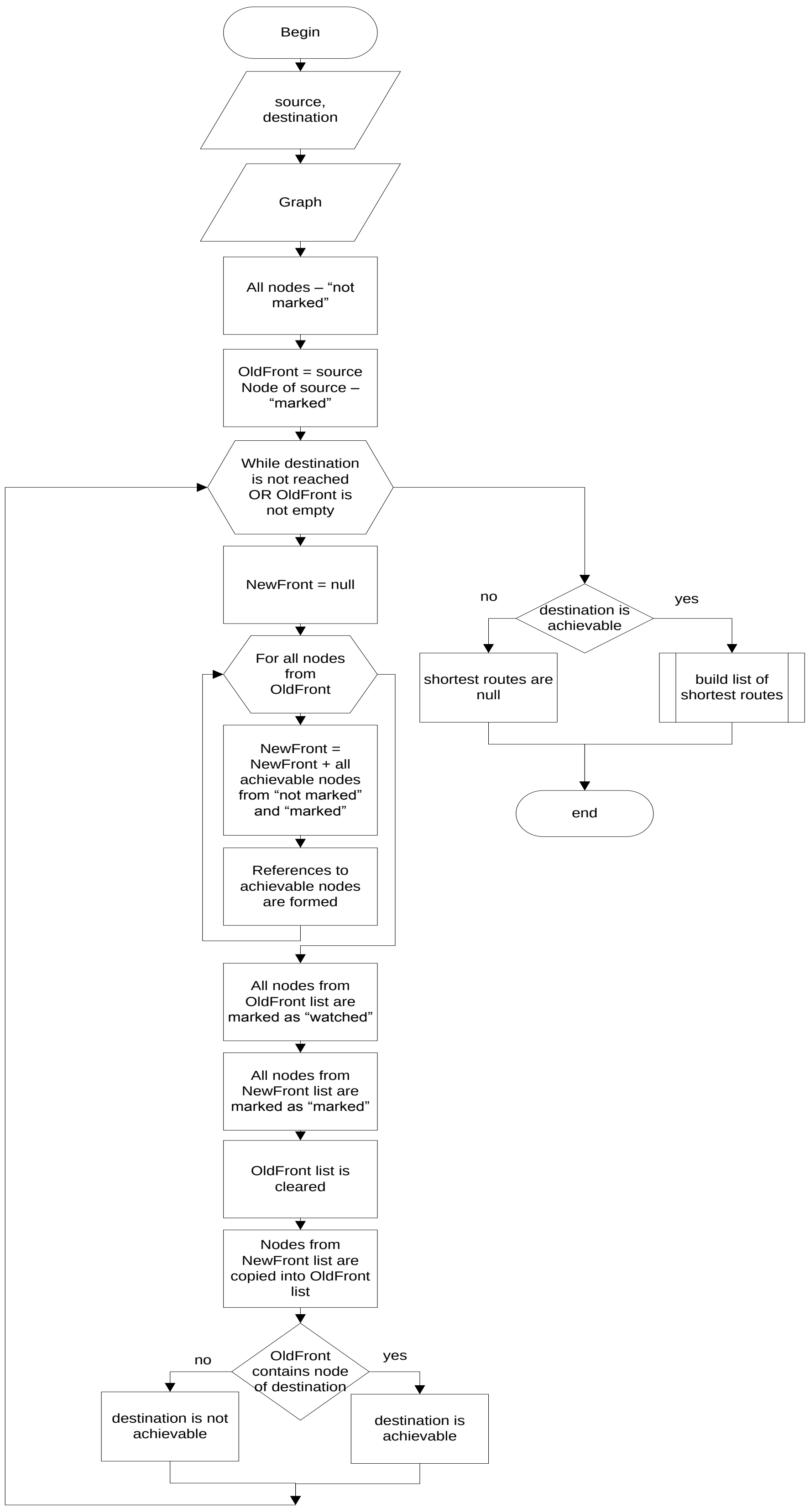

Fig. 3. The block diagram of algorithm for building a list of shortest data transmission routes

Similarly, the algorithm continues iteratively. This is shown in Fig. 5- Fig. 10. The algorithm continues while OldFront list is not empty (this means that destination is unachievable) or destination node is not put into this list (all shortest routes are found).

Node “destination” is put into OldFront. It has "marked" state (Fig. 10). Hence, there are the routes between the source and destination. Then the routes from previously generated list of references to adjacent nodes are restored.

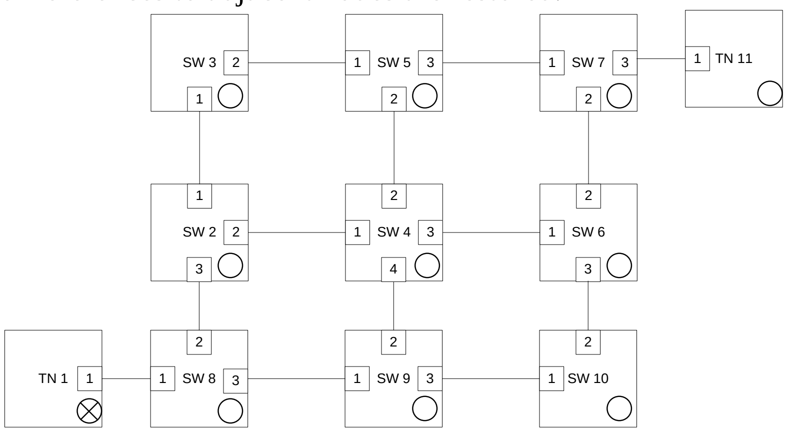

Fig. 4. Start of algorithm operation

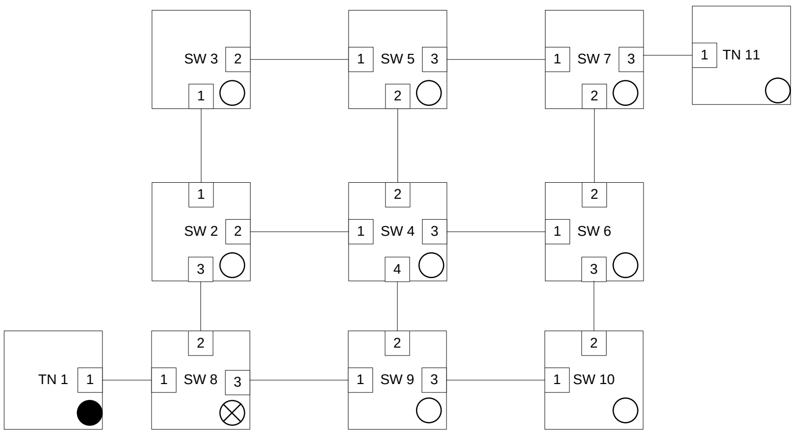

Fig. 5. Operation of algorithm, iteration 1

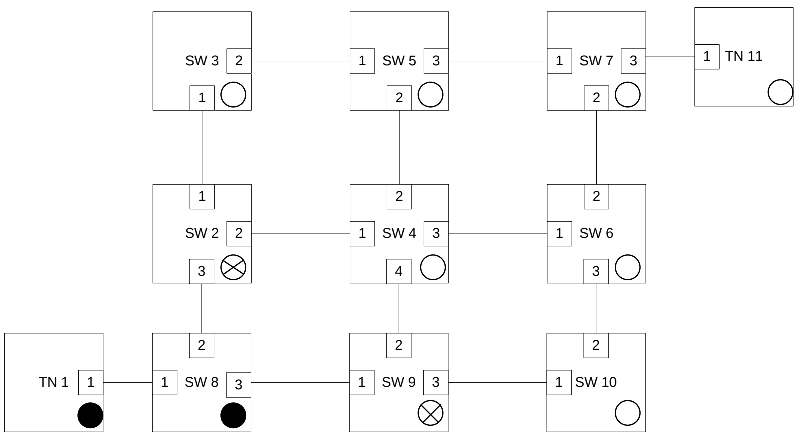

Fig. 6. Operation of algorithm, iteration 2

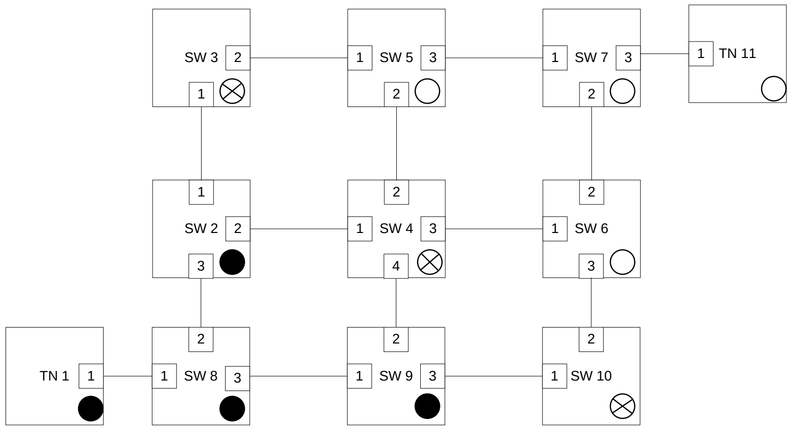

Fig. 7. Operation of algorithm, iteration 3

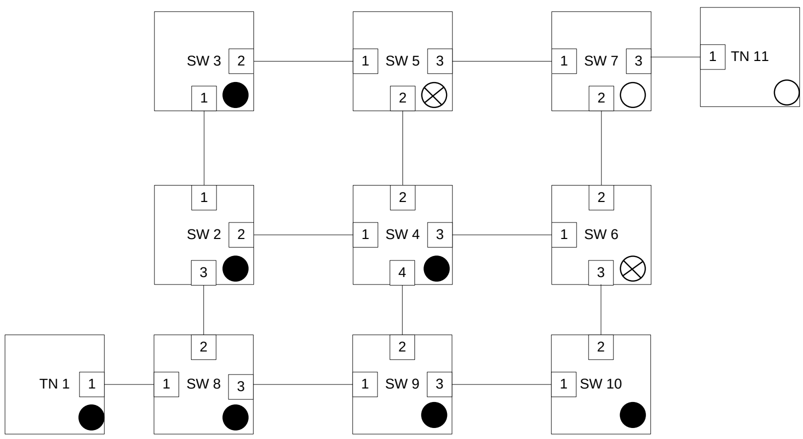

Fig. 8. Operation of algorithm, iteration 4

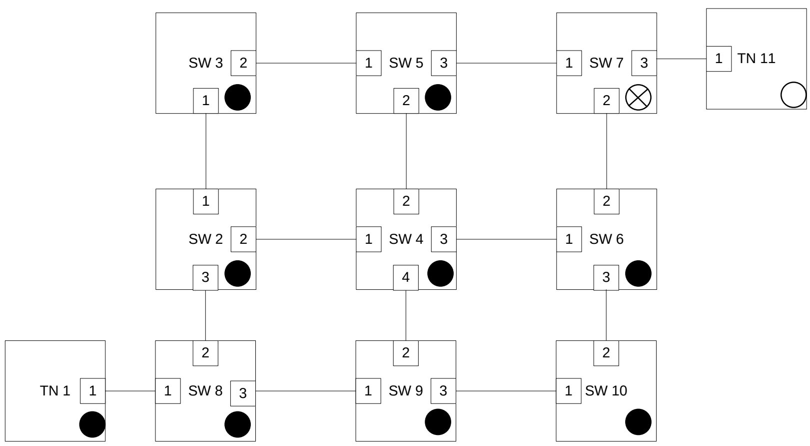

Fig. 9. Operation of algorithm, iteration 5

Graphic representation of linked lists which were found during shortest routes search is shown in Fig. 11. It contains several linked lists where references to adjacent nodes for every watched node are stored. This allows restoring the shortest routes between source and destination quickly.

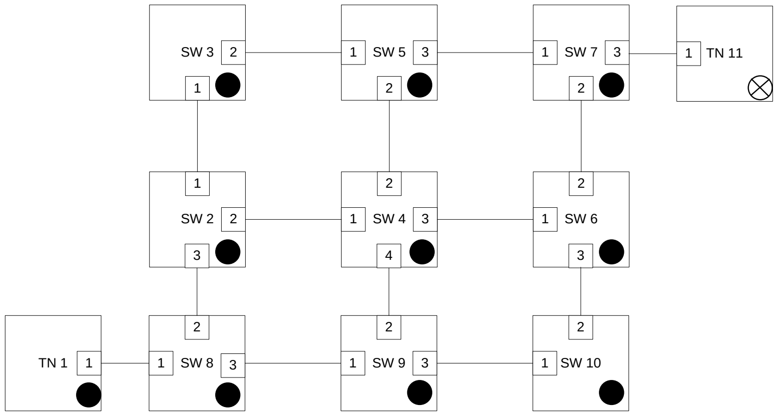

Fig. 10. Operation of algorithm, iteration 6

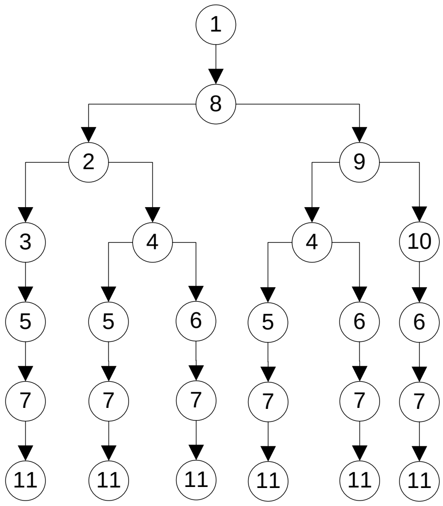

Fig. 11. Graphic representation of linked lists found during search of shortest routes

Graphic representation of shortest routes between source and destination is shown in Fig. 12.

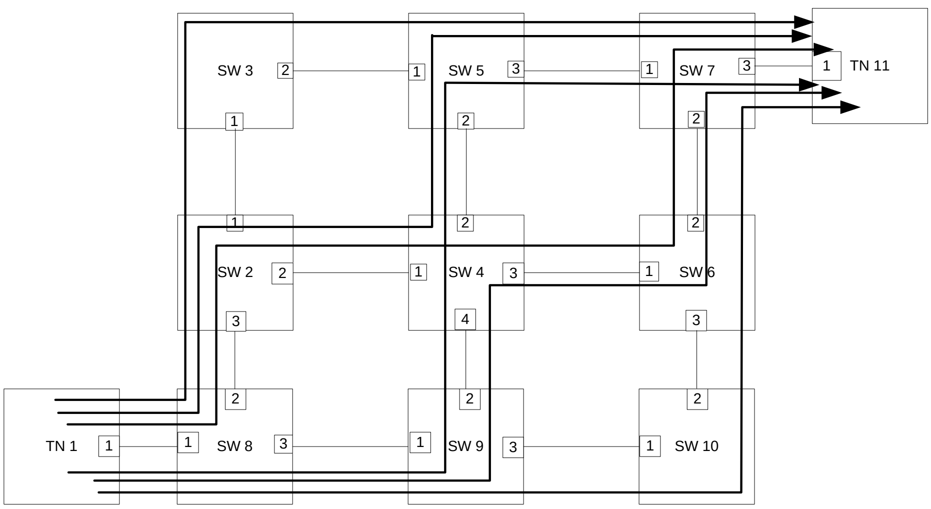

Fig. 12. Graphic representation of shortest routes between the source and destination

B. Computational complexity evaluation

The presented method is based on the Lee's algorithm. Its computational complexity is estimated as $O(n)$. The main distinguishing feature of this method is the idea of recovery not one (the any shortest of several routes), but a complete set of routes. For this reason, the complexity of our proposed method is higher. Computational complexity will increase by $O(l * k)$, where l – length and k – number of shortest routes. This occurs due to the additional pass through the structure of connections between network nodes to recover the shortest route. References structure forming has no significant effect on the increase in the complexity, because there only saving of references to adjacent nodes occurs. But this structure increases the amount of memory required. The amount of memory depends on the implementation, programming language and others.

This methodology has been implemented in the C# language. The application runs at a computer with the following characteristics:

- processor – Intel Core i-7, 2,93 GHz,
- RAM – 3 Gb.

TABLE I contains information about the time of the shortest route searching for the presented network topologies.

For network topology №1 $l = 6$, $k = 10$, $l * k = 60$ (operation), for network topology №2 $l = 8$, $k = 35$, $l * k = 280$ (operation). If we compare the number of additional operations required to construct routes and run time, then we make a conclusion that it is possible to build a set of the shortest routes for rather big network (for example several hundreds of nodes) in a reasonable time.

TABLE I
TIME TO SEARCH THE SHORTEST ROUTE

№	Network topology	Time to search the shortest route, ms
1	The network consists of 2 terminal nodes and 12 switches. The first terminal node is the source, the second terminal node is the destination. The topology of communications is 4x3 mesh.	1
2	The network consists of 2 terminal nodes and 20 switches. The first terminal node is the source, the second terminal node is the destination. The topology of communications is 4x5 mesh.	2

III. CONCLUSION

This paper proposes a method of finding all the shortest routes between the source and the destination. These shortest routes do not contain cycles and have the same length. This way it is possible to detect unreachability of the destination. When applying this methodology iteratively it is possible to build the set of alternative data transmission routes for all required pairs "source - destination".

This method can be used in local area networks and NoCs using adaptive routing. It allows to form alternative data transmission routes and is required for load balancing feature and for data transmission reliability increasing.

REFERENCES

[1] Thomas H. Cormen, Charles E. Leiserson, *Introduction to algorithms*, The MIT Press, 2002.
[2] William James Dally, Brian Towles, *Principles and Practices of Interconnection Network*", Elsevier, 2004.
[3] Axel Jantsch, Hannu Tenhunen, *Networks on Chip*, Kluwer Academic Publishers, 2003.