Fault-Tolerant Routing in Networks-on-Chip Using Self-Organizing Routing Algorithms

Aleksandr Romanov, Nikolay Myachin, Andrei Sukhov
HSE University
Moscow, Russian Federation
{a.romanov, nmmyachin, amsukhkov}@hse.ru

IECON 2021 - 47th Annual Conference of the IEEE Industrial Electronics Society | 978-1-6654-3554-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IECON48115.2021.9589829

***Abstract*—This work focuses on using self-organizing algorithms employed in the field of wireless sensor networks to networks-on-chip (NoCs) and presents a method for choosing hierarchical coordinates and a greedy forwarding algorithm for path finding. The features of algorithm functioning in the conditions of an overloaded NoC and the presence of fault nodes are considered, and a set of rules for bypassing blocked network sections is given. The proposed hierarchical coordinate method and a greedy forwarding fault tolerant routing algorithm are universal and suit for any NoC topology. To test the proposed solutions in NoCs with more than 300 nodes and mesh topology a special software simulator was developed, which confirmed the efficiency and stability of routing algorithms and coordinate assignment. Dependence of lengthening of the route and increase in the number of unreachable nodes on the number of disabled nodes (when using the routing algorithm proposed) was also investigated. It was shown that the proposed routing algorithm significantly exceeded fault tolerant algorithms and was only slightly inferior to the optimal algorithm (Dijkstra's algorithm) in the capability of constructing a route for the movement of packets through the network under nodes failures conditions.**

Keywords*—self-organizing routing in networks-on-chip, virtual coordinate method, hierarchical routing method, greedy forwarding method.

I. Introduction

Recently, further increase in processor performance is practically independent of their clock rate. The main development path is creation of multiprocessor chips; at the same time, the number of coprocessors is growing rapidly and reaches tens of thousands of cores in graphical chips. But general purpose chips do not lag behind either. For example, the WSE2 chip from Cerebras, made using a 7 nanometer process technology, consists of 850,000 computing cores [1]. In this case, the performance achieved from the simultaneous operation of multiple computing cores can be high only if there is effective communication between them [2]. The high throughput of the communication subsystem of such a multicore system-on-chip can only be achieved by building a high-performance NoC. At the same time, the communication subsystem itself can occupy up to 30 % of the chip area and up to 40% of its power consumption [3]. Therefore, NoC communication subsystem has such requirements as simplicity of built-in routers, ultra-low network delays, low power consumption, high throughput at the lowest possible hardware costs [4].

Very often, ideas and technologies that have proven themselves in one area can be successfully used in a completely new area. Such creative borrowing facilitates, cheapens, and accelerates the process of developing a new technology, as well as makes it possible to predict its prospects. Therefore, it is necessary to study close and distant analogies in order to take advantage of existing developments. In our opinion, there are many similar technologies for NoCs from neighboring areas of science, but not all of them are actively used. In this study, we search for such methods for routing in NoCs.

This study is devoted to finding analogies for routing technologies in NoCs. Section 2 of the article is devoted to an overview of works on the features of routing in NoCs. More details about this search for analogies are described in Section 3. Based on the analysis of the similarities and differences between routing goals in NoCs and in wireless self-organizing networks a virtual coordinate system and greedy forwarding algorithm are proposed in Sections 4 and 5. Comparative testing of the developed routing method (including conditions of nodes faults) is described in Section 6.

II. Related Works

The current state of research on NoC routing is presented in the review [5]. It is stated that NoC performance is highly dependent on the routing algorithm. The authors propose a classification of routing methods in accordance with the number of problems they solve. The most important problems encountered in the routing process are deadlocks, livelocks, starvation, and failures [6]; and achieving freedom from them is the main goal of routing. According to this, two main classes of routing methods are distinguished: mono objectives and multi objectives; and it is shown that it is problematic to achieve all multi objectives when routing.

Deadlock occurs when packets are waiting for the release of routers occupied by other packets, and there is a circular dependency between them. There are many different solutions to combat deadlocks by using Virtual channels, Turn model methodology [7], as well as various methodologies based on Channel Waiting Graphs [8]. However, the disadvantage of all these methods is that, as is well shown in [9], they do not work when the structure of the network is changed, i.e. they are not tolerant to the occurrence of faults when the nodes temporarily or permanently cease to function.

Livelocks are the result of use of adaptive routing methods when a packet cannot reach the destination node due to congestion of transmission channels near it and moves near the node along a circular path. This situation is usually a feature of the routing algorithm itself, which does not occur in greedy forwarding algorithms (the minimal path routing) [5].

Congestion is an uneven distribution of traffic in the network when some of its sections are congested and other sections are idle. The slowdown in communication processes in busy sections of the network leads to a decrease in the performance of the entire NoC. The reason for this is usually disadvantages of the communication topology, deterministic routing algorithms [5], as well as uneven traffic generation (and, as a consequence, presence of hotspots [10]).

Another problem that may arise in NoCs but is not mentioned in [5] is starvation [6], which is a situation when a packet never reaches its destination because certain resources do not grant access to them (but provide access to other packets). Starvation can be caused by suboptimal packet queuing at the inlet of routers, or by using packet prioritization techniques to ensure quality of service.

And finally, the most dangerous situation is network faults [11], which is the failure of individual nodes or connections on a permanent or temporary basis. It should be also noted that faults can be single (rare) and multiple. When the number of fault nodes becomes significant, this situation is close to the situation of congestion. Developers do not often take directly into account the situations of faults when designing routing algorithms and NoC designs because it is believed that the probability of faults is rather low, or can be eliminated during the manufacture of the chip. Nevertheless, just when making real chips, the incorrect operation of their individual sections is the cause of expensive rejection, the probability of which sharply increases with increasing NoC; and faults can occur during operation of the chip for a whole range of reasons, including aging of the chip [12]. Therefore, it is important to develop methods to combat faults or eliminate their consequences without failure of the entire network and methods for predicting faults [13].

Thus, giving the most catastrophic impact of faults on the network functioning in comparison with other problems, in this study, we focus on the development of design methods for fault-tolerant NoCs. At the same time, taking into account the ever increasing sizes of NoCs, we propose to use self-organizing routing algorithms. This approach partially fulfills other objectives as well, since sections of the network with deadlocks, livelocks, congestion, or starvation can also be marked as faulty, and then solutions for faults can be applied to them.

III. Self-organizing wireless networks and NoCs

In the classical theory of computer networks, many algorithms, methods and communication protocols capable to form the basis for NoC routing have been developed. In this study, it is expected to take a step in this direction and apply the routing algorithms and methods, developed for wireless self-organizing networks, for NoCs. Consider some of the popular methods and solutions used in wireless sensor networks that can be used for routing in NoCs.

The peculiarities of self-organizing networks require specialized approaches for constructing coordinate systems in them. Work [14] shows that it is possible to achieve precise localization and tracking of a destination node in a wireless sensor network. The proposed solution based on a local coordinate system consistent with global coordinates. An algorithm for creating such a coordinate system without using a global positioning system, globally available radio beacon signals, or accurate estimates of the distances between sensors is proposed. As a result, the authors prove that the resulting coordinate system is reliable and automatically adapts to failure or addition of sensors.

Work [15] proposes ad hoc positioning system (APS) based on a distributed phased positioning algorithm that works as an extension to vector distance routing and GPS positioning, which allows calculating the approximate location of all nodes in the network in conditions when only limited part of nodes have the ability to independently determine their location.

In many implementations of wireless peer-to-peer and sensor networks, location determination is essential. In this case, it is often enough to operate with virtual coordinates, and not with real ones. For example, in work [16] the authors solve the problem of obtaining virtual coordinates using the information about the connections.

Coordinate systems are often built on the basis of information about the operating conditions of the network, so work [17] proposes to reduce the effect of interference and increase the throughput in wireless mesh networks through the use of space-time diversity. Based on measurements of the received signal levels, a virtual coordinate system is constructed used to determine the sets of paths along which transmissions will pass with the least inter-stream interference. Based on the sets of non-interfering paths, the gateway node determines the order in which it schedules frames for different connections.

After the coordinate system is specified, geographic routing systems are applied, including those based on greedy forwarding algorithms. For example, in work [18], first an etalon of the minimum possible communication costs is justified, and then a greedy routing algorithm is proposed to develop a special topology of a fault-tolerant connection for a given application kernel graph that corresponds to etalon.

Thus, based on the analysis of solutions used in self-organizing wireless networks, a number of ideas on how to use approaches from this area in NoCs can be formulated:

1) Since the structure of core connections is fixed, the network (although consisted of hundreds or thousands of nodes) remains static, which makes it expedient to use virtual coordinates assigning a unique address to each network node. Moreover, such an assignment can be carried out in advance, even at the stage of NoC design. In this work, several methods for constructing virtual coordinate systems were studied, including a method based on a set of hierarchical vertices.

2) NoCs are much more structured than self-organizing wireless networks. In them, any node (except the border one) has a fixed number of neighbors; so, building a route with a minimum number of transitions between nodes in NoC is not a problem. But with an increase in the number of computational operations, nodes and even entire NoC segments can become inoperative due to failures. Moreover, with an increase of the load, the number of fault network sections will increase. At the same time, configurations of idle nodes during NoC operation can change rapidly; therefore, NoC, working with a significant computational load, is like a mobile self-organizing network.

3) Coordinates of nodes can be assigned at the stage of initial network configuration; they represent a numerical vector (a set of integers). The coordinates can be used to accurately recover the identification number, but it is impossible to assign coordinates knowing only the node identification number.

4) To construct a route between computational nodes, it is proposed to use the greedy forwarding method [19]. In order to apply this method, it is proposed to use a metric based on the dot product of two vectors, which makes it possible to determine the next hop of the route. The greedy forwarding-based routing algorithms proposed allow automatic bypassing congested sections of the network.

5) Special attention should be paid to the elaboration of various mathematical aspects of self-organizing routing methods. One of them is a method for choosing the centers of hierarchies, which serve as reference points for virtual coordinates, as well as a method for determining their number for unambiguous assignment of coordinates.

IV. Virtual coordinate system for NoCs

Before proceeding to the description of the proposed coordinate system and routing rules in NoCs, it should be noted again a very important property of NoCs, which is that the communication subsystem between the nodes of the network has a fixed structure, and the neighbors of each node are fixed. So, the coordinates assigned to the nodes will be static, i.e. they will not change over time. This is a significant difference between NoCs and wireless self-organizing networks, in which coordinates are variable and can change. Therefore, it is enough to assign coordinates to nodes once, after which these coordinates will be saved and used. When transmitting a packet with data over the network, a node can be considered temporarily inoperative in case of failure (or due to deadlock, livelock, congestion and starvation), while routing rules should help bypass the inoperative node with minimal route lengthening.

The basic requirement for a virtual coordinate system is that the coordinates of each node are unique and not repeatable. Let us formulate the routing problem. Let there be a set of X computational nodes. These nodes can exchange data with neighboring nodes. It is required to link not only adjacent nodes, but also elements from the original set located at a distance. At the same time, for NoCs it is important that the route between the nodes is not pre-built, and the packet with data begins to immediately be transmitted from node to node. If a node is inoperable, it should be bypassed automatically. Setting a coordinate system involves assigning several variables (coordinates) to each node. The rules for assigning these variables depend on the network topology. This paper considers the most common and simple topology – mesh, which is a flat topology, where each node is connected to four neighboring ones. After assigning coordinates, we can consider the distance between nodes and introduce the concept of a vector to describe the transition between two nodes.

The proposed approach for specifying the coordinate system is based on a combination of two methods. The first method is the virtual coordinate method [20] – for constructing a coordinate system for a given set of nodes. The second method is the hierarchical routing method [21], which is based on the choice of the central point of the hierarchy and the construction of a hierarchical tree.

The combined approach assumes the choice of several centers of hierarchies. For flat (mesh) topology it is necessary to select four centers of the hierarchy, so that the coordinates of the node relative to these centers are unique for any node of the original set. The algorithm for choosing the centers of the hierarchy and marking the network for an arbitrary flat set of nodes consists of several stages:

1) An arbitrary node from the initial set is selected, and the neighborhoods of this node are sequentially constructed. In the last of the marked out neighborhoods, a node is randomly selected. This node will be the center of hierarchy A. Since the mesh topology is rectangular, the center of hierarchy A will be one of the corner nodes. Thus, in the case of mesh, one can immediately select any corner node, but then the algorithm will lose its universality property and will not be suitable for other topologies.

2) Neighborhoods are marked around the first center of the hierarchy A. All nodes in the set will receive the first coordinate with the designation A_i. In the last neighborhood, a node is again randomly selected. This is the second center of the hierarchy; let us designate it as C. This designation was chosen specifically to ensure a consistent order of coordinates, since in the case of mesh, this node will be located in the corner diagonally opposite the initial center of hierarchy A.

3) Neighborhoods are marked out relative to the second center of the hierarchy C. All nodes in the set will receive a second coordinate with the designation C_i. One of the nodes for which the sum $A_i + C_i$ is maximum should be selected as the third center of the hierarchy B. Among the nodes with the maximum sum, let us choose the one for which the modulus of the difference $|A_i - C_i|$ will be minimal so that the third center of the hierarchy is equally equidistant from the first two.

4) Neighborhoods are marked around the third center of the hierarchy B. At this stage, all nodes in the set get the third coordinate (B_i). Now each node i are characterized by three coordinates A_i, B_i, C_i. A search is made for the node that is farthest from all three centers of the hierarchies. A feature of the coordinate system is that the smallest of the three coordinates shows the minimum distance from a given node to three centers of hierarchies. Thus, the most distant node will be the one where the smallest of the three coordinates will be the largest among all the nodes in the set. So, the search takes place in two stages: first, for each node, the minimum coordinate is selected, and after that the node with the smallest coordinate has the greatest value from the entire set. If there are several candidate nodes, then the one for which the sum $A_i + B_i + C_i$ is the largest is selected.

5) Neighborhoods are marked around the fourth center of the hierarchy D. All nodes of the set receive the fourth coordinate denoted as D_i.

6) The network has been marked, and all the nodes have received the coordinates. Each of the coordinates of node i is the number of hops from the corresponding center of the hierarchy to the current node. So, the coordinate is the number of the neighborhood in which node i falls relative to a certain center of the hierarchy.

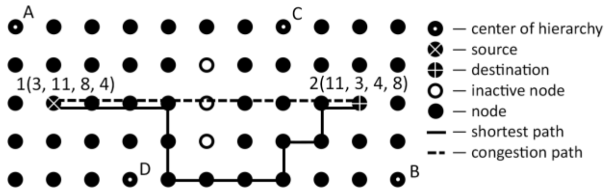

Fig. 1. Routing procedure with congestions.

Note the following features of the proposed coordinate system: firstly, it is not orthogonal, it is impossible to single out the basis vectors in it; secondly, all the coordinates have only positive values. This feature means that it is impossible to determine the direction of movement of the packages from the coordinates themselves. This imposes certain requirements on the routing algorithm.

V. Greedy Forwarding Routing Algorithm for Use in NoCs

In accordance with the proposed coordinate system, for any two nodes j and i it is possible to construct a vector describing their relative position M_{ji} with coordinates $(A_j - A_i,\ B_j - B_i,\ C_j - C_i,\ D_j - D_i)$.

The coordinates of this vector show the number of neighborhoods between two nodes; depending on the relative position of the nodes, these coordinates can be either positive or negative. Thus, this vector fully characterizes the route between two nodes.

Let us formulate an algorithm for constructing a route as a sequence of transitions between the nodes. The algorithm is based on the rule for selecting the next node of the route among the neighboring nodes of the current one. The proposed algorithm is a special case of the greedy forwarding method.

Suppose it is necessary to find a path from node i with coordinates $(A_i,\ B_i,\ C_i,\ D_i)$ to node j with coordinates $(A_j,\ B_j,\ C_j,\ D_j)$. The route can be characterized by vector M_{ji} with coordinates $(A_j - A_i,\ B_j - B_i,\ C_j - C_i,\ D_j - D_i)$.

The idea of the routing algorithm is that from node i the packet is transmitted to the neighboring node $k(i)$, for which the dot product of vectors $M_{ji} * M_{ki}$ is maximum (Fig. 1). That is, there will be selected that node $k(i)$, through which it is possible to get as close as possible to the final node j.

The dot product is defined in the standard way as the sum of products of the corresponding coordinates of two vectors.

The formulated rule allows building any route with minimal overhead costs due to the use of a virtual coordinate system: there is no need to determine the origin of coordinates, as well as the directions of the coordinate axes. Also, since all coordinates of any node are positive integers, there is no need to expend additional hardware resources to store the sign of numbers.

The proposed method can be used to build routes in networks with congestions (we use term "congestion" to determine situation when NoC is overloaded and significant part of nodes is fault). To do this, it is enough to consider the overloaded node as fault (inoperative) and ignore it when finding the next node by the greedy forwarding method. That is, when congestion occurs, the node is temporarily considered inoperative, and the route is bypassed. After the node is restored to work, it can immediately be used as an intermediate route node.

VI. Experimental Testing of the Proposed Algorithm

In order to confirm the effectiveness of the proposed coordinate assignment method and the routing algorithm, testing was carried out. To perform the computational experiment, we used NoCs instances with at least 300 nodes. For testing, a special simulator was developed with the help of which the following experiments were carried out:

1) Determination of centers of hierarchies for networks with different topologies and marking of nodes with virtual coordinates. As a result of studying more than a thousand different variants of random flat NoC topologies [22] (mesh, torus, hypercube, octagon, circulant, BFT, WK-recursive, etc.) with a different number of nodes, the correctness of the method was shown – each of the nodes got the unique coordinates.

2) Comparison of lengths of the routes obtained using the proposed algorithm with algorithms for finding the shortest paths (for example, Dijkstra's algorithm [23]). In the absence of overloads and fault nodes, the proposed algorithm was minimal.

A. *Experimental testing of the proposed algorithm in mesh topology with nodes faults*

The algorithm was tested under conditions of faults of separate nodes. Gradually increasing the number of fault nodes, the possibility of organizing communication between three pairs of randomly selected nodes from the initial set was studied. An assessment of the increase in the average length of the route between nodes with an increase in the number of fault nodes was carried out.

The principle of greedy forwarding has a number of disadvantages, the most significant of which is the possibility of current route segment falling into the local minimum. This is a situation when all currently no-fault neighbors are located farther from the destination than the current node. That is, the dot products will take negative values. This situation in NoCs occurs only in congested networks when blocking entire groups of nodes.

Let us describe the ways to resolve such situations. Although the dot product can take positive and negative values, in any case, it is necessary to choose an adjacent node for which this product is maximum. When the dot product is the same for two nodes the node that is closer to the geometric center of the network configuration should be chosen. At the same time, testing shows that there are a number of configurations when building a route becomes impossible. In Fig. 2 node 165 is the start of the route; node 173 is the end of the route; nodes 105, 107, 168, 179, and 184 are temporarily fault. Turquoise arrows show the route of the package.

In order to avoid deadlocks, it is proposed to use a number of rules:

1) The current node of the route polls its neighbors before transmitting a packet to them.

2) If a neighboring node has fewer than two no-fault neighbors (apart from the current node), then such a node should also be marked as fault and should be excluded from

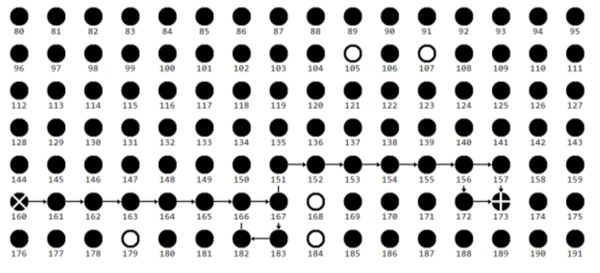

Fig. 2. The route of the packet in terms of fault nodes.

candidates for the continuation of the route. This rule does not apply to the starting and ending nodes of the route.

3) The current node in the route checked for the number of fault neighboring nodes. If the number of neighbors (not including the previous route node) is less than two, then this node should be considered a local minimum and taken as the initial route node; building a new route should be started, considering all no-fault nodes new. Thus, it becomes possible to move through some nodes of the old route.

The proposed algorithm has one exception. It concerns corner nodes; in the case of a mesh topology, they have only two neighboring nodes, which makes them automatically inaccessible. If the fault node is located nearby, it will be impossible to attain the corner node closer than two hops. To correct this situation, the network protocol implementing the routing algorithm must be supplemented with a special rules:

1) Each corner node sends special labels to its neighbors.

2) If the current node, when marking the route, obtains such a label from the neighbor, all the restrictions are canceled, and the packet is transmitted to this node.

We carried out a whole series of tests to compare our algorithm with existing solutions. The family of fault tolerant algorithms [24] and the optimal algorithm were chosen as objects for comparison. Optimal algorithm means building a route using Dijkstra's algorithm [23]. In practice, it is problematic to implement such an algorithm in the form of a routing protocol because of its complexity and the need for each node to know the complete network topology. At the same time, the simulator allows using it to build optimal routes and, using them as reference, evaluate the effectiveness of other routing algorithms.

The first dependence which we checked as a result of computer simulation is the lengthening of the route from the percentage of fault nodes (Fig. 3). In the course of the experiment, pairs of nodes setting the starting and ending points of the routes were selected. After that, the number of fault (disabled) nodes which were selected at random gradually increased; the relative lengthening of the routes between the selected nodes was estimated.

Fig. 3 shows that with a small number of fault nodes (up to 3 %), the proposed algorithm is almost as good as the optimal one; and with a number of fault nodes up to 6 %, its efficiency is higher than that of the One-Fault-Tolerant algorithm [24]. With a larger number of fault nodes, it shows worse results. Nevertheless, such a mode of network operation seems unlikely to us.

The main advantage of the algorithm appears in the second test. A computer simulator makes it possible to estimate the proportion of routes between arbitrary network nodes that cannot be built using a given routing algorithm (unreachable nodes). Dependence of this share on the number of fault nodes in percent is shown in Fig. 4.

If the number of fault nodes does not exceed 7 %, the proposed algorithm does not differ from the optimal one; a route between any two nodes can always be built. The One-Fault-Tolerant algorithm can have problems as early as 1 % of fault nodes. That is, from the point of view of the possibility of constructing a route, the algorithm proposed by us is many times superior to the reference one.

Thus, the algorithm obtained is very simple in implementation and independent of the network topology; it does not require numerous requests to the network (including broadcast ones) for building a route. Typically these queries are needed to build routing tables. The proposed routing algorithm does not require routing tables or their analogs. All the information necessary for routing is contained in the address assigned to the node. Moreover, this information is sufficient in the conditions of real network operation when some nodes may temporarily not work. Such fault sections of the network are easily bypassed due to the peculiarities of the system for assigning virtual coordinates to network nodes and the method for calculating the distance between nodes.

VII. Conclusion

This article discusses organization of routing in NoCs based on analogies with self-organizing wireless networks. There are many similarities between the two areas of engineering, but NoCs are more organized. The topology in NoC is usually fixed, and all the nearest neighbors of any node are strictly defined. At the same time, when the load on the network increases, some NoC nodes may temporarily become inoperative, i.e. NoC can be viewed as a system with dynamically changing topology and configuration of nodes. For such cases, we propose to apply self-organization methods.

The initially fixed NoC topology makes it possible to assign nodes virtual coordinates based on a hierarchical approach. To do this, in planar topologies it is proposed to select four reference vertices, and the coordinates of the nodes will be the distances to these four vertices. As a computational experiment on the simulator shows, it is possible to choose the position of the reference vertices in such a way that each node will receive a unique set of coordinates.

For the proposed coordinate system, a routing algorithm based on a simple system of rules was developed. These rules reflect the principle of greedy forwarding. It assumes transmission of a packet with data to that of the neighboring nodes that is closest to the end point of the route. We use the concept of a vector as the difference between the coordinates of two nodes to characterize the distance between them. The principle of maximum dot product of two vectors allows determining the next waypoint. The principle of greedy forwarding is universal and makes it possible to build a route even in conditions of congestion and related node failures.

The effectiveness of the solutions proposed was tested experimentally on a specially designed simulator. It allows setting various configurations of nodes, finding the centers of the hierarchy, and assigning virtual coordinates to nodes. The work of the proposed algorithm was tested in conditions of

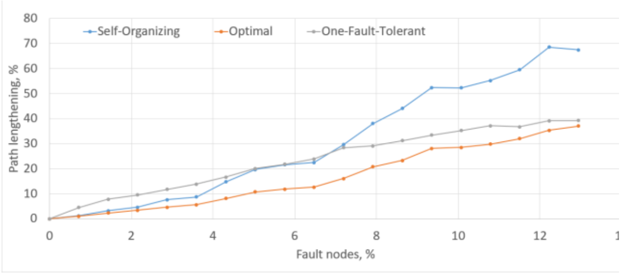

Fig. 3. Routing procedure with congestions: path lengthening.

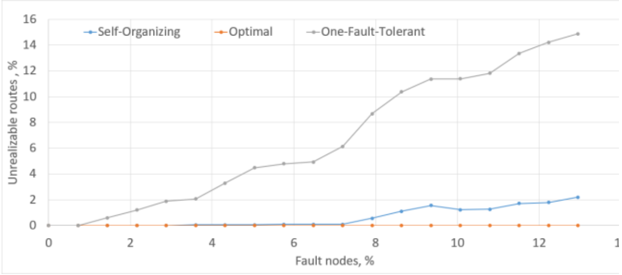

Fig. 4. Routing procedure with congestions: unreachable nodes.

network congestion and node failure. A family of fault tolerant algorithms and an optimal algorithm (Dijkstra's algorithm) were chosen as objects for comparison. The dependence of the lengthening of the route and the increase in the number of unreachable nodes on the number of inoperative nodes was investigated. It was shown that the algorithm proposed was significantly superior to fault tolerant algorithms and only slightly inferior to the optimal algorithm in terms of the possibility of constructing a route.

ACKNOWLEDGMENT

The study was implemented in the framework of the Basic Research Program at the National Research University Higher School of Economics (HSE University).

REFERENCES

[1] S. K. Moore, “Cerebras’ New Monster AI Chip Adds 1.4 Trillion Transistors,” *IEEE Spectrum*, 2021. https://spectrum.ieee.org/cerebras-giant-ai-chip-now-has-a-trillions-more-transistors (accessed Jul. 29, 2021).

[2] S. Kumar *et al.*, “A network on chip architecture and design methodology,” in *Proceedings IEEE Computer Society Annual Symposium on VLSI. New Paradigms for VLSI Systems Design*, 2002, pp. 117–124, doi: 10.1109/ISVLSI.2002.1016885.

[3] J. Hu and R. Marculescu, “Energy-aware mapping for tile-based NoC architectures under performance constraints,” in *Proceedings of the Asia and South Pacific Design Automation Conference, ASP-DAC*, 2003, vol. 2003-January, pp. 233–239, doi: 10.1109/ASPDAC.2003.1195022.

[4] Y. Chen and C. Han, “Channel modeling and analysis for wireless networks-on-chip communications in the millimeter wave and terahertz bands,” in *IEEE Conference on Computer Communications Workshops*, Apr. 2018, pp. 651–656, doi: 10.1109/INFCOMW.2018.8406954.

[5] A. Benmessaoud Gabis and M. Koudil, “NoC routing protocols – objective-based classification,” *J. Syst. Archit.*, vol. 66–67, pp. 14–32, 2016, doi: 10.1016/j.sysarc.2016.04.011.

[6] S. Das, C. Karfa, and S. Biswas, “Formal Modeling of Network-on-Chip Using CFSM and its Application in Detecting Deadlock,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 28, no. 4, pp. 1016–1029, Apr. 2020, doi: 10.1109/TVLSI.2019.2959618.

[7] J. G. Christopher and M. N. Lionel, “The turn model for adaptive routing,” *J. ACM*, vol. 41, no. 5, pp. 874–902, Sep. 1994, doi: 10.1145/185675.185682.

[8] M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila, J. Flich, and H. Tenhunen, “Path-based partitioning methods for 3D networks-on-chip with minimal adaptive routing,” *IEEE Trans. Comput.*, vol. 63, no. 3, pp. 718–733, 2014, doi: 10.1109/TC.2012.255.

[9] C. Jackson and S. J. Hollis, “A deadlock-free routing algorithm for dynamically reconfigurable Networks-on-Chip,” *Microprocess. Microsystems*, vol. 35, no. 2, pp. 139–151, Mar. 2011, doi: 10.1016/J.MICPRO.2010.09.004.

[10] E. Kakoulli, V. Soteriou, and T. Theocharides, “Intelligent hotspot prediction for network-on-chip-based multicore systems,” *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 31, no. 3, pp. 418–431, Mar. 2012, doi: 10.1109/TCAD.2011.2170568.

[11] J. M. Montanana, D. De Andres, and F. Tirado, “Fault tolerance on NoCs,” in *27th International Conference on Advanced Information Networking and Applications Workshops*, 2013, pp. 138–143, doi: 10.1109/WAINA.2013.221.

[12] M. Radetzki, C. Feng, X. Zhao, and A. Jantsch, “Methods for fault tolerance in networks-on-chip,” *ACM Comput. Surv.*, vol. 46, no. 1, pp. 1–38, Jul. 2013, doi: 10.1145/2522968.2522976.

[13] C. Feng, Z. Lu, A. Jantsch, J. Li, and M. Zhang, “FoN: Fault-on-Neighbor aware routing algorithm for Networks-on-Chip,” in *IEEE International SOC Conference*, 2010, pp. 441–446, doi: 10.1109/SOCC.2010.5784672.

[14] R. Nagpal, H. Shrobe, and J. Bachrach, “Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 2634, pp. 333–348, 2003, doi: 10.1007/3-540-36978-3_22.

[15] D. Niculescu and B. Nath, “Ad hoc positioning system (APS),” in *IEEE Global Telecommunications Conference*, 2001, vol. 5, pp. 2926–2931, doi: 10.1109/GLOCOM.2001.965964.

[16] T. Moscibroda, M. Wattenhofer, R. O’Dell, and R. Wattenhofer, “Virtual coordinates for ad hoc and sensor networks,” in *2004 Joint Workshop on Foundations of Mobile Computing*, 2004, pp. 8–16, doi: 10.1145/1022630.1022633.

[17] H. Lim, C. Lim, and J. C. Hou, “A coordinate-based approach for exploiting temporal-spatial diversity in wireless mesh networks,” in *Annual International Conference on Mobile Computing and Networking*, 2006, pp. 14–25, doi: 10.1145/1161089.1161093.

[18] P. Shah, A. Kanniganti, and J. Soumya, “Fault-tolerant application specific Network-on-Chip design,” in *7th International Symposium on Embedded Computing and System Design*, Feb. 2018, pp. 1–5, doi: 10.1109/ISED.2017.8303920.

[19] J. Xie, D. Guo, X. Shi, H. Cai, C. Qian, and H. Chen, “A Fast Hybrid Data Sharing Framework for Hierarchical Mobile Edge Computing,” in *IEEE Conference on Computer Communications*, Jul. 2020, pp. 2609–2618, doi: 10.1109/INFOCOM41043.2020.9155502.

[20] M.-J. Tsai, H.-Y. Yang, and W.-Q. Huang, “Axis-Based Virtual Coordinate Assignment Protocol and Delivery-Guaranteed Routing Protocol in Wireless Sensor Networks,” in *26th IEEE International Conference on Computer Communications*, 2007, pp. 2234–2242, doi: 10.1109/INFCOM.2007.258.

[21] M. Amadeo, G. Ruggeri, C. Campolo, A. Molinaro, and G. Mangiullo, “Caching Popular and Fresh IoT Contents at the Edge via Named Data Networking,” in *IEEE Conference on Computer Communications Workshops*, Jul. 2020, pp. 610–615, doi: 10.1109/INFOCOMWKSHPS50562.2020.9162741.

[22] N. E. Jerger, T. Krishna, and L. S. Peh, *On-Chip Networks: Second Edition*. Morgan and Claypool Publishers, 2017.

[23] Y. Deng, Y. Chen, Y. Zhang, and S. Mahadevan, “Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment,” *Appl. Soft Comput. J.*, vol. 12, no. 3, pp. 1231–1237, 2012, Accessed: Nov. 30, 2018. [Online]. Available: http://www.mathnet.ru/rus/at607.

[24] D. Park, C. Nicopoulos, J. Kim, N. Vijaykrishnan, and C. R. Das, “Exploring fault-tolerant network-on-chip architectures,” in *International Conference on Dependable Systems and Networks*, 2006, pp. 93–104, doi: 10.1109/DSN.2006.35.