# Efficient Routing Table Minimization for Fault-Tolerant Irregular Network-on-Chip

Rafael Gonçalves Mota, Jarbas Silveira, Jardel Silveira
LESC-DETI-UFC
Federal University of Ceará
Fortaleza, Brazil
faelmota@gmail.com; {jarbas; jardel}@lesc.ufc.br

Lucas Brahm, Avelino Zorzo, Filipo Mór, César Marcon
PPGCC-PUCRS
Pontifícia Universidade Católica do Rio Grande do Sul
Porto Alegre, Brazil
{lucas.brahm, avelino.zorzo, filipo.mor, cesar.marcon}@pucrs.br

***Abstract*— Table-based routing is a common approach for a fault-tolerant Network-on-Chip (NoC). This approach is hard to scale, since the table size tends to grow according to the NoC size. To surpass this problem, some works, such as the Region-Based Routing (RBR), have proposed techniques for saving routing tables area. This work proposes an alternative routing technique finding among communication pairs selected by the RBR technique, shrinking the amount of bits used for setting up regions. The experimental results have shown that the proposed technique reduces the maximum and mean number of regions, decreasing the space used by routing tables, providing more scalable and efficient routing NoC.**

***Keywords*— *NoC; irregular topology; fault-tolerance, routing***

## I. Introduction

A Network-on-Chip (NoC) is an architecture that provides efficient communication among hundreds, or even thousands, of cores in a System-on-Chip (SoC). Mesh is one of the most used NoC topologies, because of its regularity and scalability, where routers are interconnected as a grid shape and each router has one core connected to a local port.

A mesh NoC with faulty links is an irregular architecture [1][2] and a well-known routing technique for irregular architecture based on routing tables [3][4][5]. However, routing tables reduce the scalability of the communication architecture, due to the spaces required for these tables that grow according to the network size. Thus, considering that NoCs often have to accommodate a large number of communication cores, table-based routing may become prohibitive shortly.

This work proposes a new method for efficient minimization of routing tables using (i) Segment-Based Routing (SBR) [4] to generate routing constraints in a way to prevent deadlock occurrences, and (ii) Region-Based Routing (RBR) [5] to find routing paths between communication pairs. RBR decreases the space needed to save routing tables dividing the network into regions. However, when the NoC has a high irregularity level, even RBR requires huge tables to allow an efficient routing. The method proposed here decreases the number of regions generated by RBR, such as the number of bits needed to save each region. This new method changes the original way RBR selects its paths, making use of only one path among the communication cores, which, in turn, requiring fewer bits to save a region.

The remainder of this work is divided as follows. Section II summarizes the SBR and RBR fundamentals. Section III details the proposed method. Section IV reports and discusses the experimental results. Finally, Section V presents the major findings and implications of this study.

## II. Basics of SBR and RBR

SBR is a routing algorithm based on turn prohibition which is composed of two phases: (i) segment computation and (ii) routing restrictions placement [4]. At the segment computation phase, the algorithm partitions the NoC into segments composed of routers and links, each segment is characterized by a turn restriction that makes the network deadlock-free. At the placement restrictions phase, the algorithm defines that each segment contains a localized turn restriction that suits the routing of elements inside segment. SBR guarantee a deadlock freedom and fully connected NoC.

On the routing-based table algorithm, the path that a message will go through is determined by querying the routing tables [6]. This implementation is often used for irregular networks routing. The main problems of this technique are size, energy consumption and querying time of routing tables that grow proportionally with the NoC size [1]. Consequently, this technique may not fit the requirements of networks containing a large number of routers.

RBR is an implementation of a specific imposed routing that groups several targets on regions for each router [8]. Each region is defined for every destination reachable from the same set of input ports and is sent to the same output ports of a specific router.

The RBR algorithm involves three steps: (i) the path computation, (ii) the region computation, and the (iii) region merge [5]. The network topology and the restriction set, imposed by SBR, enable to compute the possible routing paths for a specific source-destination pair. The algorithm, used to find paths, first tries to calculate all minimum paths, and if no minimum path exists for a specific source-destination pair, the algorithm computes a non-minimum path [5]. Thus, if there are not minimum paths, a single non-minimum path is used.

The algorithm groups the computed paths into structures called ‘routing options’ that cover one input port, one output port, and one destination. Two routing options in the same router might have its output ports merged if both of them have

978-1-5090-6113-6/16/$31.00 ©2016 IEEE 

the same input port for some specific destination. Similarly, they might have its input ports merged if both have the same output port for a specific destination router. After the routing options merging, the algorithm computes the regions from these options. To do this, routers reachable through the same output ports and starting from the same input ports are grouped creating regions. Each region is represented by its input ports, output ports and by two routers (upper left and lower right), defining a rectangle containing all grouped destinations. Later, the algorithm merges the computed regions to avoid surpassing the maximum number of regions allocated for the routing tables. Two regions are allowed to merge if they form a new rectangle and if the output ports of one region is a subset of the output ports from another.

Fig. 1 exemplifies the routing restrictions imposed by the SBR algorithm and the RBR mechanism. The dashed lines represent the regions for the encircled router. All packets arriving on the analyzed router traveling to a destination in one of the configured regions must be sent through a specific port. For example, if a packet is destined for region R1, it should be forwarded through the west port.

Regions may be set for more than one output port, maintaining the adaptability of the algorithm. By guaranteeing its adaptability, the network becomes trustworthy, and delays decrease occur because congested routes or unexpected new faulty links may be circumvented. The determination of regions demands the joint use of input and output ports to analyze routing restrictions avoiding cyclic dependencies, and, therefore, possible deadlocks. It is important to mention that although the destinations of two regions may be overlapped, they will have a different set of input ports [4].

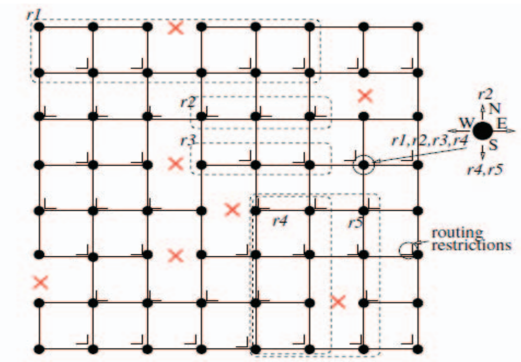


Fig. 1. An irregular 8×8 mesh NoC showing RBR applied to five regions [8].

## III. THE PROPOSED METHOD

The proposed method differs from RBR in the way it performs the path selection. A decrease in the number of used paths reduces the number of routing options, decreasing the number of regions [8]. Regarding this premise, we reduced the number of selected paths between a source-destination pair using link weight metric, which represents the number of paths crossing a specific link [4] and helps to determine the highest congested area of the NoC. This approach, guarantees a similarity increase between routing options on each router, which, in turn, merges the routing options easier decreasing the number of regions RBR produces. The algorithm computes all minimum paths among all communicating pairs to allow this selection. We grouped these pairs in ascending order according to their link weight. Then, the algorithm selects the weightiest path and updates the weights of the links. This update may affect other communicating pairs. Thus, the selection method repeats until the detection of a convergence on the links weights occurs.

We implemented metrics to evaluate the number of generated regions, explore the efficiency of the proposed method and its impact on the NoC performance. To evaluate the efficiency of region reduction, we used the mean and the maximum number of regions as metrics. For this, we calculated the mean number of regions for the whole NoC and then collected its maximum values, allowing the comparison of sizes and fault percentages for each network.

### A. *Generation of Irregular Mesh Topologies*

The method to generate irregular grids (represented by a graph) makes use of the network size and the desired percentage of faulty links, where the vertices represent routers and the edges represent links. On this graph, a faulty link is simply absent. The algorithm first generates a complete graph including all possible routers and links according to the network size. Subsequently, the exclusion of links occurs randomly in a manner that does not change network reachability until reaches the desired number of faulty links.

### B. *Calculation of Maximum Fault Percentage*

Reachability is a communication property, which allows a packet, coming from a specific destination, to reach all the network routers. The fault percentage of network links directly affects reachability. When the fault percentage reaches a certain level, depending on the network size, reachability starts to decrease due to the presence of isolated NoC routers.

We implemented a method to verify if a network within a certain amount of routers accepts a given percentage of faults, which enables to keep the maximum number of faults under its upper limit. This method evaluates the minimum amount of links allowed for each dimension and checks if its fault percentage keeps the network above this limit.

Equations (1), (2) and (3) describe the maximum quantity of links without faults that enables, at least, one path to connect all routers; the quantity of links a regular N×M NoC mesh topology contains; and the maximum percentage of faulty links for having a connected network.

$$\text{NoFaultyLinks} = \text{N} \times \text{M} - 1 \tag{1}$$

$$\text{Links} = 2 \times \text{N} \times \text{M} - (\text{N} + \text{M}) \tag{2}$$

$$\text{MaxFaultyLinks} = (\text{Links} - \text{NoFaultyLinks}) \,/\, \text{Links} \tag{3}$$

## IV. EXPERIMENTAL RESULTS

This section compares three selection path methods: (i) the RBR standard, which has the goal to search for all minimum paths; (ii) a random selection method that, similarly to our approach, searches for a unique path but without considering a special rule for path selection; and (iii) our proposed approach. The comparison, which encompasses the mean and the maximum number of regions and the amount of bits used for region storing, enables to demonstrate the superior performance of our proposed method in reducing the amount of regions and occupied space. All data about the number of regions and quantity of bits used were obtained by running a Monte Carlo simulation with thousands of iterations, where each iteration generates a new faulty link scenario performing a new irregular network topology. These simulations start setting as constants a fault percentage and a specific network scale; then, generate thousands of random topologies and collect the mean and maximum region numbers of the whole network to use for analysis.

### A. *Maximum and Mean Number of Regions*

Fig. 2 shows the maximum quantity of regions as a function of the percentage of faulty links according to the network size. The number of faults starts at zero increasing by increments of one unit until it reaches its limit. The resulting curves in Fig. 2 show a simulation corresponding to the classical RBR path selection and the path selection proposed here for a 7×7 mesh NoC (i.e., a NoC with 49 routers).

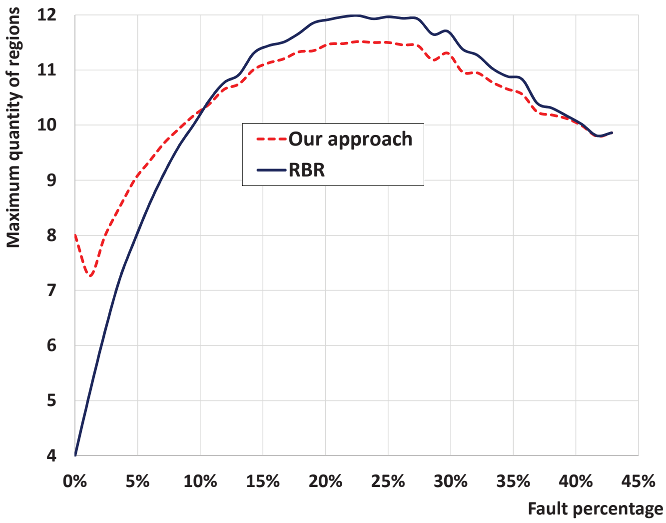


Fig. 2. Maximum quantity of regions regarding the percentage of faults.

Fig. 2 displays two regions separated by a threshold represented by the intersection of both curves. Before this intersection, the classical RBR performs better, generating fewer regions. After the crossing point, the proposed method outstrips RBR. In the 7×7 mesh topology, the MaxFaultyLinks is 42.86%. At this point, both curves reach the same quantity of regions because there is only one possible path among all router pairs. Thus, both methods select the same path, which leads to the generation of the same regions.

Fig. 3 compares the average quantity of regions reached by RBR and our proposed approach. It is possible to notice the resemblance between these curves and the curves shown in Fig. 2. However, on Fig. 3 the threshold is reached at a higher fault percentage. Fig. 2 shows that the proposed method is effective for NoCs with fault percentages beyond a certain threshold (in the 7×7 mesh NoC this percentage is 10.71%). Besides, even before this threshold, the difference between the proposed and classic methods is small. Fig. 3 shows that this threshold occurs when the fault percentage is equal to 14.29%.

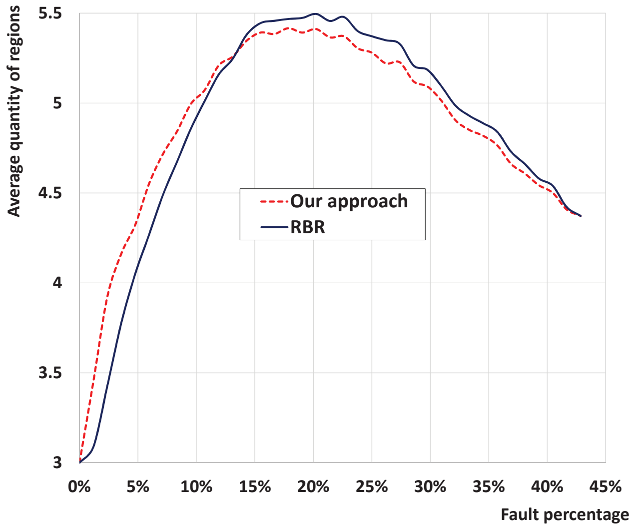


Fig. 3. Average quantity of regions according to the percentage of faults.

### B. *Evaluation of Fault Percentage Thresholds*

The thresholds presented in Fig. 2 and Fig. 3 are the starting points where the proposed algorithm starts to perform better when compared to RBR. Table I shows these thresholds for some NoC sizes (i.e., containing 16, 25, 36, and 49 routers).

Table I. Threshold of fault percentage regarding some NoC sizes.

| NoC size (number of routers) | 16 | 25 | 36 | 49 |
|---|---|---|---|---|
| Fault percentage (%) | 16.7 | 15.0 | 14.9 | 14.3 |

Table I suggests the existence of an inverse proportion between the decreasing values for the intersection points previously described with the increasing number of routers. Thus, the bigger the network, the lower the point where the proposed method outstrips the RBR algorithm. Therefore, the proposed method is more efficient for larger networks with more faults, which is common nowadays.

### C. *Evaluation of the Quantity of Bits Used*

According to RBR, 22 bits are required for storing a region. Fig. 4 shows that the first 5 bits represent the input port, the subsequent 6 bits identify the upper left router, the next 6 bits designate the lower right router and the last 5 bits indicate the output ports (one bit for each port – north, south, west, east and local).

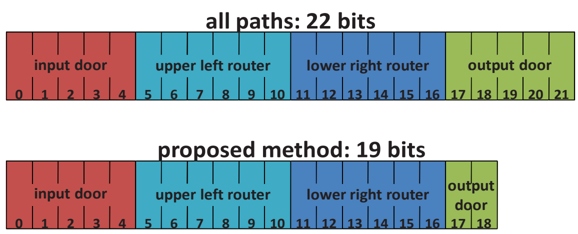


Fig. 4. Number of bits for storing a region.

The proposed method makes use of only one path between 2 routers and regions may have only one output port enabled, allowing to reduce the number of bits from 5 to 2, saving 3 bits for each region. This reduction is possible because it is possible to map up to 4 ports using 2 bits, all required ports are potentially represented. Thus, multiplying the average quantity of regions by the number of bits required for storing every region, it will result in the mean number of required bits.

### *D. Comparing RBR, Our Method and Random Selection*

Fig. 5 compares the average quantity of bits required by RBR, our proposed algorithm, and the random selection.

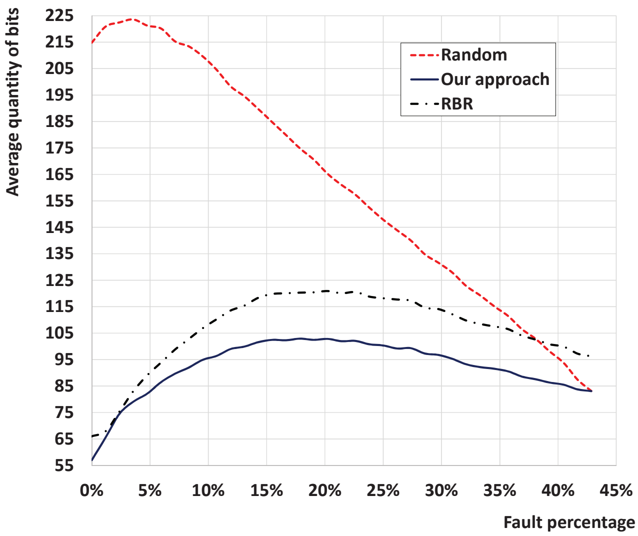


Fig. 5. Average quantity of bits required for each selection method according to the percentage of faulty links on NoC.

Firstly, we analyze our approach compared to the random selection method regarding that both use the same number of bits to store a region; i.e., both exclude the bits corresponding to the output ports. As one can attest, our proposed method shows a better performance regarding the average amount of required bits. It happens because our proposal allows a higher level of similarity between routing options, enabling that more paths cross the same NoC region. Therefore, although both approaches require the same number of bits for region representation, our method demands fewer bits because the number of generated regions is smaller, which makes this solution more efficient than the random selection.

Comparing our approach with RBR, it is apparent that our approach requires fewer bits for all fault percentages. Inspecting Fig. 5 and Fig. 2 enables the identification of the specific point at which the proposed method generates a greater number of regions and where the number of bits required for table storage is considerably lower resulting in a reduction in the total amount of required memory. Hence, for any percentage of faulty links, the proposed method performs better than RBR, specifically in regards to the reduction in the number of required bits.

### *E. Average Quantity of Regions Regarding NoC Sizes*

Fig. 6 illustrates the average number of regions for the simulated networks. Results show an increase in the average number of regions as the number of routers on percentages of faulty links increase. Although for a network completely fault free, the average amount of regions is the same regardless of the network size. However, an increase in the number of faulty links tends to increase the average quantity of regions.

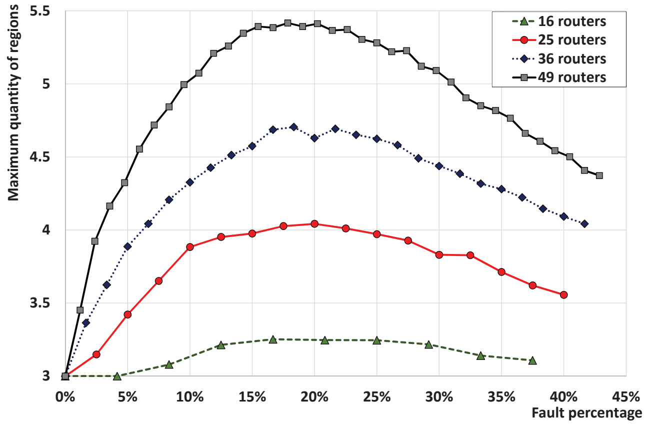


Fig. 6. Comparison of the maximum quantity of regions for four NoC sizes.

Fig. 6 shows two regions limited by a peak, which is equivalent to the maximum value of the average number of regions. Thus, for all simulated networks a fault percentage generates a maximum regions average. After this percentage, the average number of regions starts to drop again.

## V. Conclusions

This work proposes a new method for path selection that allows finding the paths that produce higher levels of similarity between routing options. The experimental results show that our proposed method is superior to RBR [5] for a specific percentage of faulty links. Additionally, the results point out that the proposed method presents an average reduction of 12.94% of required memory when compared to RBR. Besides, the proposed method is more efficient than RBR for all tested percentages of faulty link.

## References

[1] E. Bolotin et al. **Routing Table Minimization for Irregular Mesh NoCs**. *Design, Automation Test in Europe Conference Exhibition (DATE)*. pp. 942-947, 2007.

[2] X. Qingli et al. **An Efficient Routing Scheme for Irregular Mesh NoCs**. *IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC)*, pp. 121-124, 2013.

[3] A. Bose et al. **A Low Latency Scalable 3D NoC Using BFT Topology with Table Based Uniform Routing**. *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 136-141, 2014.

[4] A. Mejia et al. **Segment-Based Routing: an Efficient Fault-Tolerant Routing Algorithm for Meshes and Tori**. *International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 25-29, 2006.

[5] A. Mejia et al. **Region-Based Routing: A Mechanism to Support Efficient Routing Algorithms in NoCs**. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, v. 17, n. 3, pp. 356-369, Mar. 2009.

[6] L. Liu et al. **A Novel Approach using a Minimum Cost Maximum Flow Algorithm for Fault-Tolerant Topology Reconfiguration in NoC Architectures**. *Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 48-53, 2015.

[7] J. Siveira et al. **Scenario preprocessing approach for the reconfiguration of fault-tolerant NoC-based MPSoCs**. *Microprocessors & Microsystems*, v. 40, n. C, pp. 137-153, Feb. 2016.

[8] J. Flich et al. **Region-Based Routing: An Efficient Routing Mechanism to Tackle Unreliable Hardware in Network on Chips**. *Symposium on Networks-on-Chip (NOCS)*, pp. 183-194, 2007.