Providing a fault-tolerant algorithm for on-chip interconnection networks

Shahram Jamali¹, Fateme Sarkohakiand², Reza Fotohi²

¹Department of Computer Engineering, Mohaghegh Ardabili University, Ardabil, Iran
²Department of Computer Engineering, Germi branch, Islamic Azad University, Germi, Iran

Corresponding Author email: Fotohi.Reza@Gmail.com

ABSTRACT: As technology scales, fault tolerance has become a key concern in the on-chip communication. On-chip network reliability can be increased with the development of fault-tolerant routing algorithms. Distributed routing algorithms are a class of algorithms in which routing decisions are taken without the knowledge of the total network status. Given the time and place events of system and on-chips, dynamic routing algorithms can adapt themselves in network on chip (NOC) required to the practical applications according to that. In this paper, a new dynamic routing algorithm (routing algorithm with error detection) for NOC practical programs that have capabilities to deal with failures for static and dynamic routing and detecting them from hardware errors will be presented. Also, our proposed algorithm was compared with some distributed routing algorithms, whose goals were to maximize fault tolerance. This proposed dynamic routing algorithm can adapt itself in the network on chip (NOC) is required for practical applications. Finally, the simulation results of our proposed method were compared with several other algorithms.

Keywords: Fault tolerance, network on chip, flooding algorithm, Random Walk algorithm

INTRODUCTION

Since a number of different approaches of fault tolerance have been proposed for large-scale systems, these algorithms are not the function of the NOC implementation toward the area and major overhead storage. Recently, working on fault-tolerant algorithms specifically for the NOC has been started, which includes various forms of the Gossip algorithm. We propose a new fault-tolerant algorithm and compare it with the flooding probabilistic algorithm and direct flooding and Random Walk algorithm have previously suggested. Our model will be that Algorithm for uniform connectivity applied in probabilistic flooding algorithm has been used to make a fair comparison. Flooding algorithm and Random Walk algorithms will be analyzed and it will be shown that the flooding algorithm has a better performance than Walk Algorithm. [Benini et al., 2002; Dally et al., 1998]. In the next section, we will introduce a new dynamic algorithm for the NOC practical applications, with the ability to locate and dealing with failure for static and dynamic routing, and detecting them from permanent hardware errors and moreover, the distribution of load on whole of network using stress factors [Benini et al., 2002; Pavlidis et al., 2006; Bobda et al. 2005]. It will be compared to the direct flooding algorithm and it will be shown that the proposed algorithm has better performance.

The rest of paper has been organized as follow: 2 Flooding probabilistic algorithm, 3 direct flooding algorithm, 4 algorithm 5 Random Walk, and routing algorithms with error detection, 6 Comparison and simulation results, and 7 Conclusion

Probabilistic flooding algorithm

Flooding algorithm is a modification of a simple flooding protocol. In this algorithm, when a message is generated, it will be passed through all its neighbors with probability p, and is deleted with probability p-1. P must be chosen carefully so that flooding algorithm performance to be achieved without a large number of redundant copies of messages. Likely, by getting multiple copies of the same message at each node time is increased relatively.

This error frequently occurs in the transition stage, when the message is not needed to be spread to the destination except for a limited number of nodes but the message is sent to all nodes and they will lead that forward. The messages do not disappear during this period, early stage of flooding is formed of two cycles that all nodes during working on a two hubs should receive a copy of the message for insurance. An
implementation of this algorithm in hardware needs a pseudorandom number generator to decide in all cases to repeat a specific message.

**Direct Flooding Algorithm**

The destination is not particular in probabilistic flooding algorithm and receiving packets, and is regardless of where they have been routed in the network. Direct flooding algorithm uses the highly ordered NOC structure to guide flooding to the destination. In this algorithm, P the probability of passing this message to any of the outgoing links, is not constant, but varies depending on destination. When Algorithm starts, packets are likely routed to the destination node, and allow packets to be located adjacent to the target packets fast. To approach the packets, it is better to send all packets that face failed nodes to various paths to reduce the risk. Precisely the probability that a message is sent on a desired output port is computed as follow:

Firstly, DC, Manhattan distance between the current node and the destination node is determined. This distance is also calculated for all adjacent nodes to current node i.e. distances, DN, DS, DW, and DE. For the remaining nodes (DC ≥ DX) MX multiplication factor is equal to (DX, 4). This function is a scenario where the distant target weight is multiplied by the factor intensity, and thus all data packets follow identical path. Using a multiplication factor of up to 4, it will be allowed the port to be highly regarded in most cases and as a result latency (delay) is lowered. That’s while sending packet on the favorite port still takes considerable time. Finally, each one of MX multiplication factors is based on gossip rate. G is to find the probability of transmission to a particular link. G is a user-defined constant that can be variable to control the level of performance against fault tolerance. This algorithm, such as probabilistic flooding, has a statistical chance of not repeating a received message. Thus there are two cycles of primary flow to the area, and as a result all nodes within two operational hubs will receive a copy of this message.

**Walk Random algorithm**

Results obtained from two previous algorithms, showed significant additional overhead, and generally limitation on overall performance of the network.

In this section, we introduce the Random Walk algorithm, which is limited to a predetermined number of copies of the message it sends to the network. These messages seek a non-deterministic way, where each node that receives one of these messages should carry it to forward to one of its outgoing links.

In order to simplify and create an additional package, the initial stage of flooding can be used, as used in the Random Walk algorithms.

The initial stage of flooding allows a limited number of packages to be produced easily, while limits the effect of any local errors.

Selection is determined among outbound links by a collection of random probabilities PN, PS, PW, PE in which the sum of all probabilities is 1. Therefore, all incoming messages are paired with exactly one outgoing message as is so in a standard network. Although normalization has been often considered as a major and slow action, due to the limited number of possible values for the multiplication factor, a reasonable approximation can be obtained from combinative structure of hardware. Value for all probabilities (PN, etc.) is calculated as follow: First, MX multiplication factors are calculated as calculated in above flooding discussion. Multiplication factors and then normalization are specified to create probabilities PN, PS, PW, PE, then a random number is generated, by which one of output ports among 4 ports is selected for driving ahead. Instead of using several ports in direct flooding algorithm, only a single random decision unit is required on each input port. This reduction in random decision unit helps normalization process to compensate additional logic required. In general, any messages in the network are toward the destination but follows a little differently way. Thus, even if some links are broken on the way, at least one message must be healthy.

**Routing algorithm with error detection (proposed algorithm)**

In this algorithm, by paying attention to the relative position of the current node and the destination node in dynamic routing algorithms, each router can forward packets to the nearest destination. Each router in the routing algorithm locates the destination address in the packet by detecting errors and then decides about the appropriate path for the packet to reach the nearest target and at the same time to prevent any defective links or density. Momentary length of packet queue from each router is stress factor has been used to disseminate information density. The proposed algorithm consists of both stress and signal verification to re-routing packet around faulty links and network load distribution to reduce the possibility of density. Detailed routing algorithm for short has been shown in Algorithm 1. Given that (, ) are Cartesian address of the destination and routers present. Only when the destination node is vertically or horizontally is beside present router some of ports have been preferred with “1”. An algorithm covers a link in two cases: permanent or temporary density faults or permanent faults. Permanent Coverage on faulty link reduces the average network delay this saves the time required to send packets on this link and waiting for acceptance in addition to avoid for packets lost. Density link coverage reduces stopping network traffic or current packet loss and sum of delay
averagely reduces with network traffic deviation from hot spots. Note that the algorithm doesn’t send the packet on the same port that when all other ports are covered will be backed. This is done for two reasons: First, based on statistics such a network (for this case) seldom happens. Second, if it happens, usually one of the masked links can be temporarily covered, therefore, the general expectation within the router buffer can save the network time and load instead of sending backed (returned) packet. For each sent packet, transmitting router before removing the packet from the output buffer waits for signal verification from the receiving router and that’s to avoid the loss of packets in the network and better ensure. If the packet does not arrive to receiving router, or if received packet owns errors, packet arrival will not be verified and receiving router should send the packet again. Dynamic routing to the destination, in addition to taking the stress factor, has the ability to avoid deadlock and live lock conditions. Therefore, this algorithm reduces locking problems in network that can cause congestion or packet loss.

Read the \((X_D, Y_D)\) for the packet to be routed.

compare the packet destination address with current Router address \((X_R, Y_R)\)

decide the preferred routing port/s to push the Packet closer to the destination.

if the number of preferred port(s) = 1

Then check the masking of this port

If the preferred port is unmasked

Then send the packet through this port

Else check on the masking of the two other ports

If one of them is masked

Then send on the other port

Else send on the one that has less stress factor

End if

End if

else check the masking of the preferred two ports

If both of them are unmasked

Then send on the one that has less stress factor

Else if one of them is unmasked

Then send the packet on this port

Else send the packet on the third port

End if

End if

Algorithm 1: A routing algorithm with error detection (proposed algorithm)

Comparison of simulation results

First, we will compare three algorithms of probabilistic flooding, direct flooding and Random Walk algorithms and to evaluate the performance of these algorithms, two separate measures: fault tolerance and delay have been used. How the reliability of each algorithm for fault tolerance that can be transient and permanent error messages with routing. Conception of fault tolerance is reliability condition of each algorithm that can route messages in despite of transient and permanent faults. Fig. s 1 to 3 show fault tolerance of these algorithms, for transient faults such as interference and software errors, and Fig. 4 shows the algorithms delay.

![Fault Tolerance for Random Walk algorithm](image-url)
Figure 2. Fault-tolerance for Probabilistic Flooding algorithm

Figure 3. Fault-tolerance for direct Flooding algorithm

Figure 4. Comparison of delay for the three algorithms, Random Walk, direct and Probabilistic flooding algorithms.
As can be seen from the algorithm, Random Walk and Flooding algorithm may offer similar levels of fault tolerance, while direct Flooding algorithms have more fault tolerance. Since these algorithms are formed of attached packets, just the first time that a message reaches the destination are included in this measurement. Fig. 4 shows the direct flooding algorithm results to find the shortest path with the lowest delay while the other two algorithms have had comparable delays. You can also observe the increase effects in Random Walk algorithm and Probabilistic flooding algorithm in reduction of algorithm latency. Random Walk algorithm provides comparable performance with Probabilistic flooding algorithm. While direct flooding algorithm provides better performance in all cases. In the rest of paper, the direct flooding algorithm will be compared with routing algorithm with error detection (proposed algorithm) in terms of reliability, latency and energy consumption, and the results will be expressed.

Figure 5. Comparison of the proposed algorithm and direct Flooding algorithm
Figure 5 displays a comparison between the proposed algorithm and direct flooding algorithm in terms of capacity, latency and power consumption. Based on fault tolerance point of view, routing algorithm with error detection is able to work and deliver 100% of the packets in the high injection rate (to 20%) in the presence of a several small errors, while flooding algorithm directly can only guarantee an average of 93.67 percent packet delivery. In addition, the direct flooding algorithm cannot work correctly beyond the injection rate of 10% due to congestion, as shown in Fig. 5 (b and c).

CONCLUSION

In this paper, several routing algorithms to achieve maximum fault tolerance introduced in the NOC and were compared. On the first, three Fault-tolerant communicative algorithms in NOC field were introduced and compared to each other. From the obtained results, we found that direct flooding algorithm, has better fault tolerance than Random Walk algorithm and probabilistic flooding algorithm and in delay discussion this algorithm is to find the shortest path with the lowest delay. We then introduced a routing algorithm for routing with error detection and compared it with direct flooding algorithm. The results of this comparison showed that the routing algorithm with error detection in terms of fault-tolerant is capable of delivering 100% of the packets are sent while the direct flooding algorithm is only able to deliver 93.67 percent of packets.

REFERENCES