IDFT: An Intermediate Node Based Deterministic Fault Tolerant Routing in 2D Mesh

C.X. Yang
Institute of Artificial Intelligence and Robotics
Xi'an Jiaotong University
Shaanxi, China
State Key Laboratory of Mathematical Engineering and Advanced Computing
Wuxi

X.T. Ding
Institute of Artificial Intelligence and Robotics\
Xi'an Jiaotong University
Shaanxi, China

X.W. Ren
Institute of Artificial Intelligence and Robotics
Xi'an Jiaotong University
Shaanxi, China

P.J. Ren
Institute of Artificial Intelligence and Robotics
Xi'an Jiaotong University
Shaanxi, China

***Abstract*-In order to alleviate the problem raised in Gomez's method[1], where messages with different source-destination pairs have great probability choosing the overlapped paths therefore, new transmission bottleneck might be generated nearby, we present an intermediate node based deterministic fault-tolerant routing, named IDFT. Our method selects the qualified intermediate node among all the candidates by its residual bandwidth without over-stressing any particular node. In addition, deterministic routing is applied along both subpaths to guarantee deadlock-free and in-order message delivery. It thereby achieves traffic workload balance across the network and graceful performance degradation even in the presence of faults.**

Keywords- fault tolerance; bandwidth utilization; deterministic routing

I. INTRODUCTION

On-chip networks(OCN) has been becoming the pervasive communication fabric in massively parallel integrated manycore architectures, ranging from high-end general-purpose computing to application-specific SoC devices [2][3]. Now, tens of OCN prototypes have been built, Tera-scale computing group in Intel has developed an 80-core Tera-Flops Chip [4] and the Single-chip Cloud Computer with 48 cores [5]. However, for deep sub-micron (DSM) VLSI process, compon-ents are especially suspect to variations caused by production and environmental influences. Faults on chips, resulting from the increasing number of integrated components in a single chip [6] [7], will increase complexity and probability. Therefore, it is essential for a highly resilient, light-weight and scalable OCN architecture to satisfy the communication requirement while overcoming unpredictable failure occurrence [8].

Redundancy is the fundamental mechanism to support fault tolerance. Spatial redundancy supplies adequate components which is well cope with permanent faults. Temporal redundancy by re-execution or retransmission for failed computation and data transmission is suitable for handling transient and intermittent faults. Information redundancy by adding information for error detection and correction helps with all fault situations. It's notable that, various kinds of faults may occur concurrently in an OCN system, so that a combination of techniques may be required. OCN takes advantage of natural redundancy including computation and communication resources to address reliability and fault-tolerance issues. Fault-tolerant routing is among the most active areas in recent OCN research. Once a packet encounters faults, it is routed through an alternative path to bypass faults. It should consider the following issues:

Heavy-weighted Router: Look-up table based routing can be reprogrammed to deal with different failure situations; however, table requirement is proportional to the network size. On the other hand, dynamic routing poses an increase in the complexity of router microarchitecture as well as routing algorithm. Both of them produce more power consumption and longer service time. Considering the tight area and power budget, deterministic routing algorithm does not need routing table with less complex router than oblivious and adaptive routing.

Load-balance: Faults in the network decrease the transmission capacity of switches and links. Most of previous researches are mainly focused on fault avoidance but overlooked traffic load-balance, since messages are always routed around the fault region and turn them into congested areas. The imbalanced load is not only impact the network performance but also potentially creates power and thermal hotspots which will further deteriorate the reliability issue.

Deadlock: Messages waiting for each other to be delivered in a cycle pattern make the transmission unreachable. In particular, wormhole routing is prone to deadlock under heavy workload situation, because flits belonged to the same packet can be spread across multiple routers as they traverse toward the destination. Once the head flit is blocked, it will hold all the buffers along its path.

Out-of-order Delivery: In-order packet delivery is a strict requirement for many applications and higher-level NoC protocols, e.g. cache coherence. With adaptive routing, packets can reach the destination in an out-of-order fashion due to the difference level of congestion along each possible path. Although reordering packets at the destination can address this problem, it will significantly increase latency and resource overheads (reordering buffer and associated control logic). Routing algorithm with single deterministic path can avoid out-of-order delivery with good scalability and constant hardware overhead, which is very attractive for massively parallel manycore system.

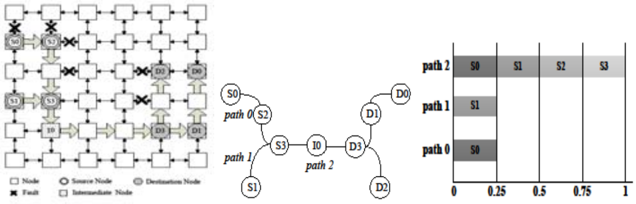
An intermediate node based deterministic fault-tolerant routing algorithm is presented in this paper, named IDFT, which is derived from Gomez's methodology [1]. IDFT algorithm transmits messages through intermediate nodes to bypass faults and uses single deterministic route along every subpath to avoid deadlock and guarantee in-order delivery. Link bandwidth is taken into consideration for choosing the qualified intermediate node in order to achieve better load-balance. Experimental results demonstrate that, when the fault-rate is 5%, 10% and 15%, IDFT algorithm achieves up to 31.7%, 37.5% and 49.9% higher saturation throughput than Gomez's method and reduce average packet latency by 13.6%, 21.6% and 19.2% respectively with 2VCs on average. This paper is organized as follows; Section 2 and 3 describe related work and motivation. The proposed routing algorithm is introduced in Section 4. Experimental results are provided in Section 5. Finally, Section 6 concludes this paper.

II. RELATED WORK

The study of fault-tolerant routing protocols has been the focus of interest of many academicians. Rajendra V. Boppana first developed fault-tolerant wormhole routing algorithm for mesh networks [11]. They introduced faulty blocks which are surrounded by fault rings or fault chains. Messages are routed using minimal paths, unless they are encountered with faults. Then messages are routed along fault rings or fault chains to avoid fault blocks. Four virtual channels of each physical link are used to make the routing method deadlock-free. However, many fully functional routers had to be disabled to meet the fault region or position restrictions. Packets routed around fault regions will lead to a significantly unbalanced link utilization, premature saturation and degrade system performance.

Jie Wu proposed a fault-tolerant routing method based on odd-even turn model [9]. This method avoids deadlock by restricting some turns' locations. Jie Wu's method has many limitations. The first is that it cannot tolerate faults which belong to different blocks. Furthermore, faults are not allowed at the edges of the network, and the nodes on the boundary of fault blocks could not be destinations.

Gomez presented a method for solving the problem by using intermediate nodes [1]. When faults are encountered, the algorithm bypasses these faults through intermediate nodes. The source node first forwards packets to a suitable intermediate node, and then the intermediate node sends packets to their destinations. Each sub-path uses different virtual channels to avoid deadlock. Multiple intermediate nodes are used to tolerate more fault situations. However, this algorithm uses minimal paths, there is a great possibility for packets to be routed through the same intermediate node or using overlapped path, bringing congestion and traffic imbalance subsequently. Jun Chen took the effect of the symmetrical turn models into account and proposed another fault-tolerant routing protocol based on positive-first and negative-first turn model [12]. However, the fault model is limited to orthogonal convex polygon. Therefore, healthy nodes have to be marked as faults to meet the shape restriction.



(a) without congestion control. (b) flows without congestion. (c) paths utilization

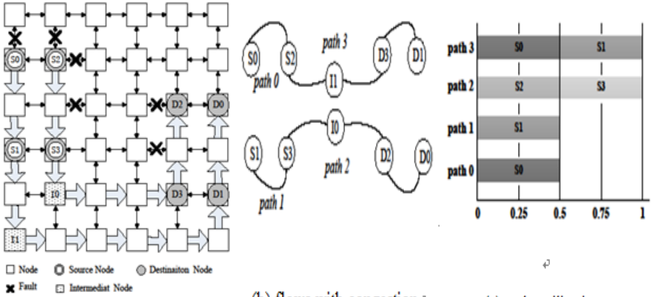
FIGURE I. A CASE STUDY OF FLOWS SHARING THE SAME INTERMEDIATE NODE.

The above mentioned fault-tolerant routing protocols may require a large number of virtual channels resulting in more hardware cost. Additionally, some of them mark the healthy node as disable one. It will sacrifice some computational and communication capacity. In this paper we propose an efficient fault-tolerant routing algorithm for 2D mesh derived from Gomez's published methodology. IDFT algorithm obtains following contributions: 1. With deterministic routing, in-order delivery is guaranteed; 2. Fault regions in this paper are not limited to any shapes or locations, therefore no healthy node is disabled; 3. Our algorithm achieves better load-balance by leveraging the knowledge of residual bandwidth of links for making routes decision.

III. MOTIVATION

Gomez proposed an intermediate node based fault-tolerant routing algorithm, which stands out among all the previous ones. Theoretically, reasonable number of failures can be addressed, which is very important to system performance and reliability. The fault model is no longer limited to any special shapes and locations. On the other hand, many previous works need to deactivate healthy nodes to form a rectangular shape in order to guarantee deadlock-freeness [9] [12]. The number of deactivated nodes increases significantly with the number of failure nodes. As for Gomez's methodology, none of healthy node need to be disabled, so that the available network communication resources could be fully utilized without losing any communication ability of healthy node. However, as described in the following there still exist some disadvantages.

Figure 1(a) shows a typical example. Nodes S0, S1, S2 and S3 route packets to D0, D1, D2 and D3, separately. Since Gomez's methodology always choose the shortest path, thus all the flows with different source-destination pair are routed through the same intermediate node I0 to their destinations. Assuming all the source nodes try to send packets at the maximum rate (e.g. equal to the bandwidth of links), because the intermediate node I0 is shared by all the flows, node I0 is overstressed and turned into the bottleneck, assuming bandwidth of these links is equally utilized by different flows using time-division multiplexing mechanism, path0 (or path1) between S0 and S2 (or S1 and S3) are 25% utilized accordingly, even though these links still have spare capacity to transfer more messages. Therefore, multiple flows pick the same intermediate node and contend for the same link will reduce network throughput and increase packet latency. The actual utilization of links is depicted in Figure 1(c).



(a) with congestion control (b) flows with congestion (c) paths utilization

FIGURE II. IMPROVED ROUTING WITH CONGESTION CONTROL.

We applied a congestion management to alleviate the contention problem caused by multiply flows choosing the same intermediate node. Figure 2(a) shows an example, packets from S0 and S1 sent to D0 and D1 are using another intermediate node I1 with non-minimal path, compared with Figure 1(b) path0 (path1) between node S0 and S2 (S1 and S3) is 50% utilized as shown in Figure 2(c). Therefore, we can see that utilization ratio of links' bandwidth can be improved by relaxing the minimal routing restriction. IDFT assign a weight to every edge, which is proportional to its residual bandwidth. Figure 2 shows an example with congestion control. Packets from S0 and S1 are sent to destination nodes D0 and D1 through another intermediate node I1, which is not through their corresponding minimal paths. Path2 and Path3 are both fully utilized. With suitable congestion management, non-minimal paths can be effective in bandwidth utilization, which improves the performance of the network.

Furthermore, for Gomez's algorithm [1] to avoid deadlock, different escape virtual channel is required for each phase (all the sub-paths share the same adaptive channel(s)), that is, from source to the intermediate node one escape channel is used, then packets need to adopt another escape channel at the intermediate node. So, an additional virtual channel is needed for each additional intermediate node. The total number of virtual channels are at least $I + 2$, (i.e., $I + 1$ escape channels and at least one adaptive channel, I is the number of intermediate nodes). However, adding virtual channels doesn't come for free, and involves adding extra buffer space and control logic, results more area and power consumption. Even worse, with adaptive routing, packets can reach the destination with different order from their sequence. Although reordering out-of-order packets in the destination side can address this problem, will increase significant latency and resource overheads.

To address aforementioned problems, we proposed IDFT, applying the above-mentioned congestion management. Deterministic routing is applied along each sub-path to avoid deadlock and keep in-order packets delivery. The advantage of choosing deterministic routing is not only because its simplicity in terms of routing decision and implementation, but also because of reducing the number of required virtual channels. The detailed implementation is described in the following.

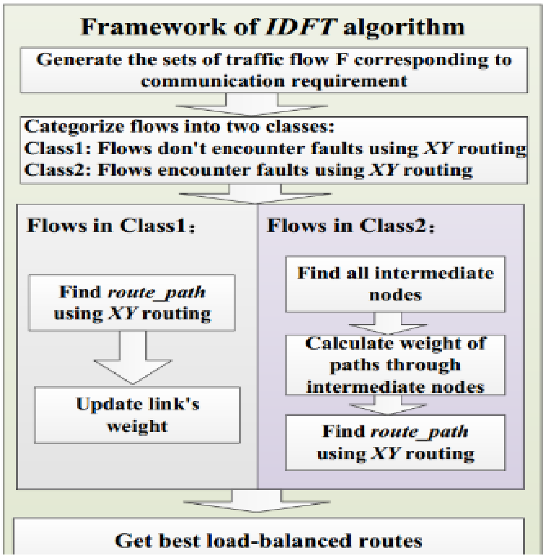

Framework of IDFT algorithm
Generate the sets of traffic flow F corresponding to communication requirement
Categorize flows into two classes:
Class1: Flows don't encounter faults using XY routing
Class2: Flows encounter faults using XY routing
Flows in Class1:
Find route_path using XY routing
Update link's weight
Flows in Class2:
Find all intermediate nodes
Calculate weight of paths through intermediate nodes
Find route_path using XY routing
Get best load-balanced routes

FIGURE III. RAMEWORK OF THE IDFT ROUTING ALGORITHM.

IV. METHODOLOGY

Figure 3 demonstrates the framework of IDFT algorithm. First, we generate the sets of traffic flow F according to communication requirement. Then, Flows are categorized in to different classes according to whether they can encounter faults using XY routing. If flows can get to their destination without encountering fault, they are routed along XY routing path. Other flows are assigned to both minimal and non-minimal paths with ample bandwidth through intermediate node. Thus, IDFT obtains better load-balanced routes.

A. *Identify the Qualified Intermediate Nodes*

Intermediate-node based fault tolerant routing is implemented by two phases, firstly, packets are routed from the source node to the intermediate node, then from the intermediate node to the destination. In the following, we use S, D and I to denote the source node, destination node and intermediate node. When using XY routing in each phase, packets can be routed to their corresponding destinations through the intermediate node I to avoid faults. The qualified intermediate node I are selected according to the following criteria:

1. S can reach I using XY routing.

2. I can reach D using XY routing.

These two requirements guarantee that packets can be routed from S to D through I without encountering faults. Generally speaking, in the absence of contention, latency is proportional to the physical distance between the source and destination, so routing algorithms are always expected to choose the shortest path and reduce the number of hops as less as possible. However, network congestion and performance degradation occurs at choke points, where the amount of incoming traffic exceeds the outgoing bandwidth. Gomez's methodology is mainly focused on selecting suitable intermediate node to minimize the length of the path between source and destination, this ignores throughput and latency degradation due to contention with other packets over shared resources. As can be observed in Figure 2, compared with Gomez's approach, without the shortest path restriction, traffic balance can be achieved by our proposed routing algorithm, even though more traversal is occurred. As a result, network performance is improved because of better load balance.

B. Methodology Based on Weight and Intermediate Nodes

There are some situations in which one intermediate node can't avoid all faults.

Then we extend our methodology to multiple intermediate nodes. But notice that the number of virtual channels increases with the increasing number of intermediate nodes. In our proposed algorithm, the number of virtual channels equals to the number of intermediate nodes.

As mentioned in the previous, there might be multiple qualified intermediate nodes and paths in which packets could be routed without encountering faults. The different paths subsequently bring different performance as mentioned in section 3. In some situations, the bandwidths of some links which lay in the overlapped paths are over-utilized, while others might sit idle. Thus, these links whose bandwidths are over-utilized could arouse traffic bottleneck and network might meet with premature saturation. Making choice among all the possible paths is necessary to improve performance. Therefore, our methodology is proposed.

Among all the intermediate nodes, we mark every link a weight, which is proportional to its residual bandwidth, and routes always choose the path with maximal available bandwidth. The weighting function of the algorithm is similar to [9]. In the initial phase, we assign the weight of each link to 1. Once flows are routed through it, the weight of this link will be updated as follows.

$$\mathrm{W} = \begin{cases} \dfrac{1}{1 - \dfrac{\sum_i d_i}{C}} & C > \sum_i d_i \\ \infty & C \le \sum_i d_i \end{cases} \tag{1}$$

Where C denotes the bandwidth capacity of link, and d_i is the bandwidth demand by flow i. When a link can't offer enough bandwidth, its weight will be infinite value. According to Dijkstra wighted shortest-path algorithm, minimum weight path is derived to route packets. Thus, among all paths decided by different intermediate nodes, the one which remains adequate bandwidth will be selected, and the traffic workload will be distributed evenly across the entire network. T_j is used to present different category of intermediate nodes. $l(x, y)$ denotes the minimal distance between node x and y. Thus, when the node I can meet the condition that $l(S, I) + l(I, D) = l(S, D) + j$, I is belonging to T_j. Route is a minimal path if $j = 0$. To get better performance, the selected path is always the path with the minimal distance among all the satisfied intermediate nodes. One example of the classification is depicted in Figure 4.

Algorithm 1: *IDFT* Methodology

```
1. Generate the sets of traffic flows F corresponding to communicationg requirements
2. Category folws into two classes
Class1:Flows will not encounter faults using XY routing
Class2:Flows will encounter faults using XY routing
3. foreach flow in Class1: do
    a. Find route_path using XY routing
    b. Update link weight according to formula(1):
    foreach link in route_path: do
       update link's weight according to formula(1)
4. foreach flow in Class2: do
    a. Calculate all intermediate nodes and categorize
    intermediate nodes into different sets Tj
    b. for j=1;j < n;j++ do
        if $Tj \neq \emptyset$    then
            foreach intermediate node in Tj: do
                calculate the residual bandwidth of links that
                in the path through the intermediate node
        if residual bandwidth $\geq$ demand bandwidth then
            I. Find route_path using Dijkstra algorithm
            II. Update link's weight according to formula(1):
            foreach link in route_path: do
                update link's weight according to formula(1)
            III. Break
5. Output routing solution
```

It is obvious that hop count affects system performance. When there is ample bandwidth, the intermediate nodes in T_j with a low value of j will be the best option to route packets. When there is insufficient bandwidth for packets to be transmitted using a minimal path, intermediate nodes in T_j with a higher value of j would be chosen, thus packets will be routed in a non-minimal path.

C. IDFT Algorithm

Algorithm 1 shows how the route path is chosen. At first, traffic flows are produced according to communication requirement of an application. Then categorize all flows into two classes depending on whether they will encounter faults along the XY routing path. Packets that will not encounter faults directly use XY routing to their destinations. Others are routed through intermediate nodes. Then the best choice is selected using the weight function among all the possible intermediate nodes, using Dijkstra weighted shortest-path algorithm, once the best route path is found, the weight of links which in the route path should be updated. In this way, the routing solution is carried out. This algorithm always assigns flows to paths with ample bandwidth, reducing network congestion and traffic imbalance.

V. EXPERIMENTAL RESULTS

In order to evaluate the effectiveness of our proposed algorithm against previous works, a cycle-accurate multicore simulator, HORNET [14], is used. We use Finter to present Gomez's algorithm. In order to get the results independent of the relative distribution of the faults, 100 simulations are performed for each fault rate case. Table 1 summarizes the configurations of our simulation. In this section, we use average packet latency, throughput and power distribution to evaluate network performance. Node faults can be regard as four links error around the fault node. Fault links and fault nodes are selected randomly, fault rate represent the percentage of fault links.

A. *Performance Over Injection Rate*

At first we measure OCN performance by means of saturation throughput and average flit latency in the presence of faults.

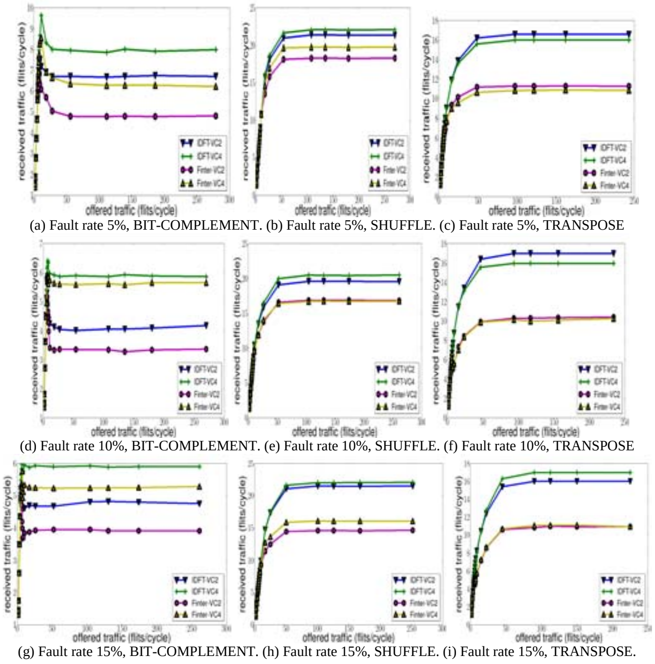
When a channel can’t offer enough bandwidth to messages, the channel will suffer network congestion. Finter focuses on fault tolerance and deadlock-free issues, resulting in heavy traffic load around the same intermediate nodes and overlapped links. In the case of obstruction, a large number of flows are contending for the same links. When the bandwidth of links is heavily over utilized by some flows, other flows have to wait until the bandwidth of links is released, increasing packet latency and reducing throughput. Without any congestion management, Finter methodology exacerbates network congestion. In contrast with Gomez’s method, IDFT spreads traffic load over the entire network by using either minimal or non-minimal paths, so that network suffers less network congestion.

Figure 4 shows the results of saturation throughput for those two algorithms with different VC numbers and fault rates under various traffic patterns. The saturation throughputs to those algorithm are approximately equal at low injection rate. But with the increase of injection rate, saturation throughput of our proposed algorithm IDFT is higher than Gomez’s by reducing network congestion. Adding more VCs gains higher saturation throughputs with additional hardware consumption.

Node that, in order to reduce congestion, IDFT algorithm may route packets in non-minimal paths, resulting in more hop count. Transmission time will increase with the increase of hop count. However, packet latency is decided by two factors, hop count and network congestion. As shown in Figure 5, at low injection rate, IDFT obtains less packet latency than Finter algorithm. Compared to routing with 2VCs, routing with 4VCs has less packet latency due to the release of head-of-line blocking, but with additional hardware consumption.

TABLE I. NETWORK CONFIGURATION SUMMARY.

Topology	8 × 8 2D mesh
Routing	IDFT, Finter
Flit size	64-bit
Link bandwidth	4 flits/cycle
Per-hop (link) latency	1 cycle
VC number	2, 4
VC buffer size	8 flits
Transistor type (Orion2.0)	LVT (Low Vth)
Clock frequency (Orion2.0)	3GHz
Manufacturing Process (Orion2.0)	65nm TSMC technology
Fault rate	5%, 10%, 15%
Traffic workload	TRANSPOSE, SHUFFLE,BIT-COMPLEMENT,
Warm-up cycles	10,000
Analyzed cycles	1200,000



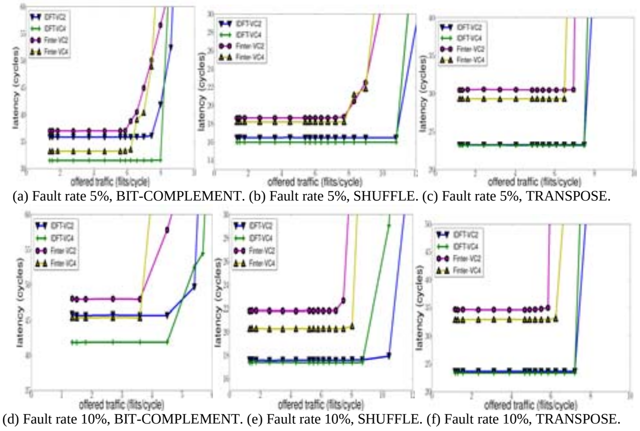
(a) Fault rate 5%, BIT-COMPLEMENT. (b) Fault rate 5%, SHUFFLE. (c) Fault rate 5%, TRANSPOSE

(d) Fault rate 10%, BIT-COMPLEMENT. (e) Fault rate 10%, SHUFFLE. (f) Fault rate 10%, TRANSPOSE

(g) Fault rate 15%, BIT-COMPLEMENT. (h) Fault rate 15%, SHUFFLE. (i) Fault rate 15%, TRANSPOSE.

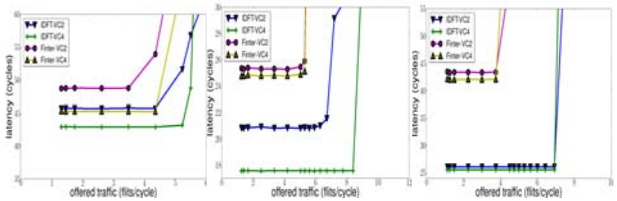
FIGURE IV. THROUGHPUT RESULTS FOR DIFFERENT FAULT RATES.

Table 2 shows how performance improvement varies with the increasing fault rate. According to our results, performance improvement rises with the increasing fault rate. Packets have more chance to encounter faults at higher fault rate. With IDFT algorithm, more flows are routed in paths with ample bandwidth. On the contrary with the trend, the saturation throughput goes down with BIT- COMPLEMENT traffic. In this traffic pattern, sources send packet to their diagonal destinations, thus these flows share more paths than other traffic patterns, and the number of paths with adequate bandwidth is decreased with the increase of fault rate. Therefore the networks would suffer heavy network congestion, resulting in network throughput dropping.



(a) Fault rate 5%, BIT-COMPLEMENT. (b) Fault rate 5%, SHUFFLE. (c) Fault rate 5%, TRANSPOSE.

(d) Fault rate 10%, BIT-COMPLEMENT. (e) Fault rate 10%, SHUFFLE. (f) Fault rate 10%, TRANSPOSE.



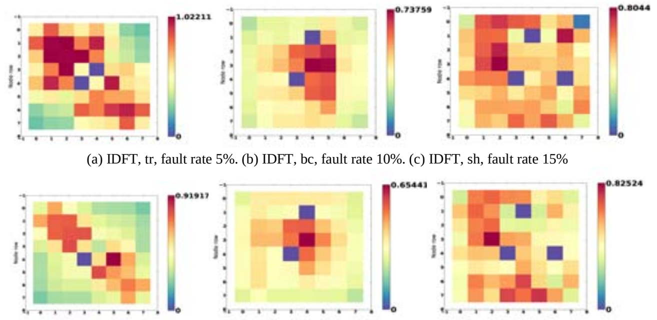
(g) Fault rate 10%, BIT-COMPLEMENT. (h) Fault rate 10%, SHUFFLE. (i) Fault rate 10%, TRANSPOSE.

FIGURE V. LATENCY RESULT FOR DIFFERENT FAULT RATES.

In summary, IDFT can get better performance than Gomez's method. IDFT obtains 31.67%, 37.47% and 49.87% (38.7%, 30.83% and 44.9%) throughput improvement and reduces 13.6%, 21.6% and 19.2% (14.03%, 21.7% and 21.2%) average packet latency with 2VCs (4VCs), respectively, when fault rates are 5%, 10% and 15%.

TABLE II. THROUGHPUT IMPROVEMENT AND LANTENCY REDUCTION WITH DIFFERENT FAULT RATES COMPARED WITH GOMEZ'S ALGORITHM.

		Throughput Improvement %				**Latency Reduction %**			
	Fault Rate	BIT-COMPLEMENT	SHUFFLE	TRANSPOSE	avg	BIT-COMPLEMENT	SHUFFLE	TRANSPOSE	avg
IDFT with 2VCs	5%	39.5	14.1	41.4	31.7	3.1	11.5	26.3	13.6
	10%	24.1	25.2	63.1	37.5	4.6	23.3	36.9	21.6
	15%	21.8	62.0	65.8	49.9	7.8	26.0	23.7	19.2
IDFT with 4VCs	5%	26.6	28.4	12.7	38.7	5.2	12.2	24.7	14.0
	10%	13.2	20.1	59.2	30.8	7.8	23.5	33.9	21.7
	15%	11.7	56.8	66.2	44.9	5.3	28.8	29.4	21.2



(a) IDFT, tr, fault rate 5%. (b) IDFT, bc, fault rate 10%. (c) IDFT, sh, fault rate 15%

(d) Finter, tr, fault rate 5%. (e) Finter, bc, fault rate 10%. (f) Finter, sh, fault rate 15%.

FIGURE VI. POWER DISTRIBUTION.

B. Power Distribution

In our next experiment we further investigate the power distribution. The power distribution reflects link utilization and traffic load. Figure 6 demonstrates that with IDFT algorithm the traffic load spreads evenly overall network. Our proposed algorithm utilizes both minimal and non-minimal paths with ample bandwidth to transfer massages, reducing network congestion and improving traffic balance.

VI. CONCLUSIONS

With an increasing fault rate, network will suffer the following issues: heavy-weighted router, network congestion, load-imbalance, and prone to deadlock. Existing fault-tolerant algorithms mainly focus on fault tolerance and deadlock-free property, but overlook traffic load balance and network congestion. This paper proposed an intermediate node based deterministic fault-tolerant routing,named IDFT, derived from Gomez's methodology to mitigate the problems. IDFT algorithm uses intermediate node to bypass faults. Each sub-path uses deterministic dimension order routing, XY routing, to avoid deadlock and guarantee in-order delivery, without additional hardware resources. IDFT assign link's weight proportionally to its available bandwidth, packets are routed not limited to the minimal paths, but taking load-balance as the prime consideration. So that, all the links of the networks are evenly utilized and the network will suffer less congestion. Experimental results demonstrate that IDFT algorithm improves significantly performance such as network throughput and packet latency compared to Gomez's algorithm.

ACKNOWLEDGEMENTS

This research is partially funded by NSFC grant No.610303036 and the Open Project Program of the State Key Laboratory of Mathematical Engineering and Advanced Computing No.2014A09.

REFERENCES

[1] M. E. Gomez, N. A. Nordbotten, J. Flich, P. Lopez, A. Robles, J. Duato, T. Skeie, and O. Lysne, A routing methodology for achieving fault tolerance in direct networks. IEEE Transactions, 55(4), pp. 400–415, 2006.

[2] W. J. Dally and B. P. Towles, Principles and practices of interconnection networks. Morgan Kaufmann, 2003.

[3] R. Ho, K. W. Mai, and M. A. Horowitz, The future of wires. Proceedings of the IEEE, 89(4), pp. 490–504, 2001.

[4] Y. Hoskote, S. Vangal, S. Dighe, N. Borkar, and S. Borkar, Teraflop prototype processor with 80 cores. Hot Chips, 19, 2007.

[5] J. Jeffers and J. Reinders, Intel Xeon Phi Coprocessor High Performance Programming. Morgan Kaufmann, 2013.

[6] The international technology roadmap for semiconductors, 2008 update. http://www. itrs.net/Links/2008ITRS/Update/2008 Update. pdf.

[7] J. Wu, A fault-tolerant and deadlock-free routing protocol in 2d meshes based on odd-even turn model. Computers, IEEE Transactions, 52(9), pp. 1154–1169, 2003.

[8] C.-T. Ho and L. Stockmeyer, A new approach to fault-tolerant wormhole routing for mesh-connected parallel computers. Computers, IEEE Transactions, 53(4), pp. 427–438, 2004.

[9] K. Aisopos, C.-H. O. Chen, and L.-S. Peh, Enabling system-level modeling of variation-induced faults in networks-on-chips. Proceedings of the 48th Design Automation Conference. ACM, pp. 930–935, 2011.

[10] M. Lis, K. Shim, M. Cho, and S. Devadas, Guaranteed in-order packet delivery using exclusive dynamic virtual channel allocation. *Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, USA, Tech. Rep. MIT-CSAIL-TR-2009-036*, 2009.

[11] J. Chen, D. Xu, and L. fu Xie, A fault-tolerant routing protocol in 2d torus based on positive-first and negative-first turn models. Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference. IEEE, pp. 1–5, 2009.

[12] P. Ren, M. Lis, M. H. Cho, K. S. Shim, C. W. Fletcher, O. Khan, N. Zheng, and S. Devadas, Hornet: A cycle-level multicore simulator. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions, vol(6), pp. 890–903, 2012.