

Performance Evaluation of Probe-Send Fault-tolerant Network-on-chip Router

Sumit Dharampal Mediratta¹, Jeffrey Draper²

¹NVIDIA Graphics Pvt Ltd, ²USC Information Sciences Institute

¹Bangalore, India-560001, ²Marina del Rey, USA-90292

{¹smediratta@nvidia.com, ²draper@isi.edu}

Abstract

With increasing reliability concerns for current and next generation VLSI technologies, fault-tolerance is fast becoming an integral part of system-on-chip and multi-core architectures. Another trend for such architectures is network-on-chip (NoC) becoming a standard for on-chip global communication. In an earlier work, a generic fault-tolerant routing algorithm in the context of NoCs has been presented. The proposed routing algorithm works in two phases, namely path exploration (PE) and normal communication. This paper presents fundamental insights into various novel PE approaches, their feasibility and performance trade-offs for k-ary 2-cube NoCs. The dependence of the normal communication phase on the probability of finding paths and their quality in the first phase emphasizes the PE's significance. One major contribution of this work is the investigation of application of constrained randomness to PE for optimizing the quality of paths. Another contribution is the proposed use of merging of traffic to reduce the reconfiguration time by a large amount (73.8% on an average).

1. Introduction

Widespread reliability challenges are expected [1] in near-term VLSI fabrication technologies because of the evolutionary changes in scaling current materials and devices and revolutionary changes associated with new materials and devices, and introduction of multiple changes in a short time period. Thus, fault-tolerance must be considered a necessity, rather than a feature. Another concern for VLSI architectures is increasing global wire lengths with associated issues like long-delay and skin-affect causing significant problems at higher frequencies of operation. As a shared resource, bus-based architectures exhibit poor performance and fault-tolerance, and have inherent scaling limitations due to physical implementation issues. Consequently, NoC is becoming a standard for on-chip global communication [2] [3].

Some approaches for achieving fault tolerance have been proposed for the NoC paradigm. Please refer [3] for their limitations. In contrast to previously proposed

approaches, the proposed parallel (not sequential backtracking) and distributed (not centralized) routing algorithm [3] [4] finds a path if it exists between a pair of nodes beforehand and removes indeterminism from the completion of communication. Path exploration remains oblivious to injection rates and message lengths during normal system operation [3]. Fault information does not need to be propagated globally with the proposed approach. This relieves the burden of error free fault information propagation. These features make the proposed algorithm suitable for wide range of applications.

This paper presents various PE approaches, their feasibility and performance trade-offs for k-ary 2-cube NoCs. PE is significant for finding paths and their quality. The effect of the number of virtual channels (VCs) and faults on the performance metrics of a PE approach is also evaluated. One major contribution is the investigation of application of constrained randomness to PE. Another contribution is the proposed merging of traffic to reduce the reconfiguration time by a large amount (73.8% on an average) as compared to the simultaneous PE approach.

Section 2 gives required background on the performance evaluation framework. Section 3 provides evaluations of various PE approaches. Section 4 investigates the effect of number of VCs and faults on the considered best PE approach from section 3. Section 5 concludes this paper.

2. Background

This section describes the routing algorithm, performance metrics, and an overview of the simulation framework.

2.1. Routing algorithm

The router works as follows in the first path exploration (PE) phase [3] [4] with the objective of finding existing simple paths between all pair of nodes: -

1. At the source node, send PE packet to all output ports (Fig. 1) and record output port direction in every packet.
2. At all other nodes deliver the packet to the current node for extracting path information and forward the packet to

only unvisited neighbors. Record direction information in delivered packets and turn information in forwarded packets. Send PE packet as a return packet (Fig. 1) to the source node if it is the first packet that has arrived from a particular source. Save path in the route cache if it contains any node for which the current node does not have a valid path; else discard the path.

3. When return packets arrive at a source node, then cache path information for visited nodes in the path in the similar manner discussed in step 2.

In the second phase, the router behaves like a source wormhole router using the cached paths in source node. Interested reader can refer to [3] [4] for more details.

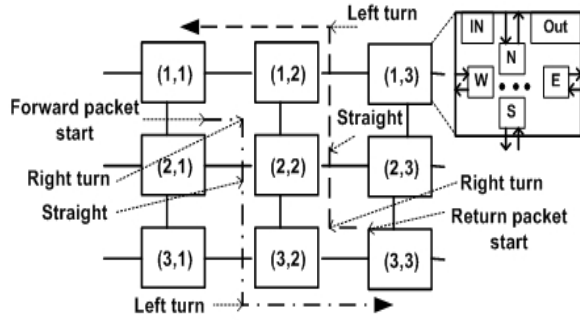


Fig. 1. Forward and return paths.

2.2. Performance metrics

The given metrics decouple workload specific dynamic performance from the reconfiguration phase and give a good indicator of the quality of paths. Additionally, workload specific dynamic performance is highly application-dependent, making the following statistical metrics much more important for a *generic network* rather than optimizing performance for certain traffic patterns.

2.2.1. Reconfiguration time. Reconfiguration time is an indication of the time taken to find a valid set of all – pair paths. This is important for handling dynamic faults, and quick reconfiguration may also be used to find different sets of paths for random approaches (section 3.2) if the quality of some set of initially found paths is not sufficient for certain applications.

2.2.2. Average and standard deviation (SD) of path lengths of all paths. Average path length (APL) gives an indication of the congestion free latency and reliability (higher APL implies a higher number of components and hence lower reliability) of the paths discovered. SD indicates uniformity in communication latency for different destinations. APL and SD of APL are calculated as follows: -

$$APL = \frac{\sum_{i=1}^P l_i}{P} \quad SD \text{ of APL} = \sqrt{\frac{\sum_{i=1}^P (l_i - APL)^2}{P}}$$

Where, l_i denotes the path length of the i^{th} path and P represents the total number of paths from all functional nodes to all other functional nodes.

2.2.3. Average and SD of number of paths per channel. An average paths per channel (APPC) metric has been proposed in this work and gives an indication of the interference in terms of number of overlapped channels used among different paths. It provides an indicator of the probable amount of congestion and reliability of found paths. A lower APPC means less probability of congestion and less dependence on shared resources. APPC and SD of APPC are calculated as follows: -

$$APPC = \frac{\sum_{i=1}^U p_i}{U} \quad SD \text{ of APPC} = \sqrt{\frac{\sum_{i=1}^U (p_i - APPC)^2}{U}}$$

Where, p_i denotes the number of paths using the i^{th} used (active) channel (input or output) in the set of total U used (not *all*) channels by the all-pair path set. The ideal value of APPC becomes 1 in this case. APPC gets uncorrelated with APL in the general case using this approach. SD of the number of PPC indicates the load balancing quality of a path set. A higher SD means some hot spot channels may be present in the network, considering uniform probability of all destinations being exercised. A lower SD is desirable because of the implied load balancing offered. But, a comparatively higher SD value may be desirable with lesser PPC than average for channels used by traffic displaying high temporal locality.

2.3. Simulation framework

The router microarchitecture was implemented in RTL level VHDL, with generics used for all parameters of interest. The default value of router parameters used are flit size equal to phit size of 512 bits (L1 cache line size of Itanium 2 processor [5]), 4 VC [3], deadlock detection timer value of 256 cycles ([6] suggests it to be good number for avoiding false deadlock detection), and deadlock buffer size of 1024 entries (sufficient for explored network sizes). Network size is currently limited to 49 nodes due to Cadence VHDL simulation environment constraints. Simulation was run for 100 μ s, with a router cycle time [3] of 3ns. Validating the VHDL simulation framework was challenging given the highly detailed hardware simulation. Cached paths were traversed from source to destination by a C++ program to create the PPC distribution for the calculation of performance metrics. Hence, it can be assured that all paths used for metric calculations are valid paths. All results and calculations were manually validated for small networks.

3. Evaluation of various PE approaches

In this section, various approaches of performing PE are investigated. At first, emphasis is given on reducing

reconfiguration time. Secondly, various approaches of improving quality of paths are explored.

3.1. Reconfiguration time optimization

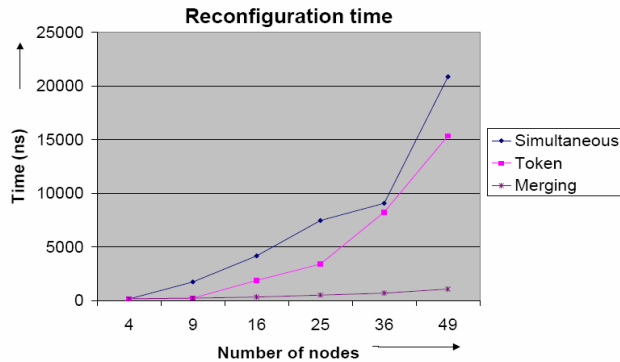


Fig. 2. Reconfiguration time optimization.

Evaluation of approaches used for reconfiguration time optimization (Fig. 2) is as follows: -

3.1.1. Simultaneous approach. In this approach, all nodes start PE simultaneously. This approach suffers from very large reconfiguration time because of large traffic.

3.1.2. Token approach. In this approach, only one node, called the token initialization node, starts PE. When the token initialization node caches paths for all possible destinations then it passes a token to a neighbor node. The neighbor node starts PE if it has not already extracted paths for all of its destinations from PE packets of other nodes reaching it before token; else it passes on the token. Token passing can be implemented in many ways, but a token ring structure has been assumed in the simulation environment. Token network does not have the requirement of being fault free itself [4]. The token approach reduces traffic because many nodes do not need to start path exploration. On an average, a 35.8% reduction in reconfiguration time was observed over the simultaneous approach.

3.1.3. Merging. The token approach still suffered from poor scalability because of the exponential increase in traffic with network size (found empirically by curve fitting for total number of paths in the search space). So, traffic has to be reduced dynamically such that it does not interfere with the probability of finding a path. One such approach is merging of the search space, where when some number of same type (e.g. forward or return PE flits) of path flits from a particular source-destination pair request an output channel, then one is granted the channel and the rest are discarded. This results in an effective merging of the search space traffic, and a large reduction in reconfiguration time (73.8% on an average as compared to simultaneous approach).

3.2. Quality of paths optimization

Merging employs a greedy mechanism of caching paths and sending them as return flits to the source. Introducing more randomness into the PE phase to arrive at a possibly better solution of paths is accomplished by using a random number seed and run time. The random mechanism employs the greedy approach as a backbone, but it allows for overwriting of paths if the probability of caching that path (as obtained from a random number generator) is greater than a certain threshold. 32-bit non-local and asymmetrical cellular-automata CA25443 was used as a random number generator because it passed more empirical randomness tests, displayed large cycle length [7] and allowed efficient implementation.

3.2.1. Constrained random (CR) path exploration.

Token passing and simultaneous mechanisms with merging were investigated for different probabilities of caching a path i.e. 0.25, 0.5, and 0.75. A wait interval, after the greedy mechanism completes, of 4096 cycles is used, which is enough to achieve a near optimal value for the parameter being directly optimized for most approaches. The purely random approach suffered from a lack of sense of good and bad solutions. So, directing it, with some constraint to make it evolve towards good solutions, was required. The constraint used was the path length. So, in addition to a random probability qualification, an arrived packet was cached only if its path length was less than that of the cached path. This resulted in an asymptotic convergence to path sets with minimum path lengths in the steady state. PPC or other parameters can also be used instead of path length as a constraint.

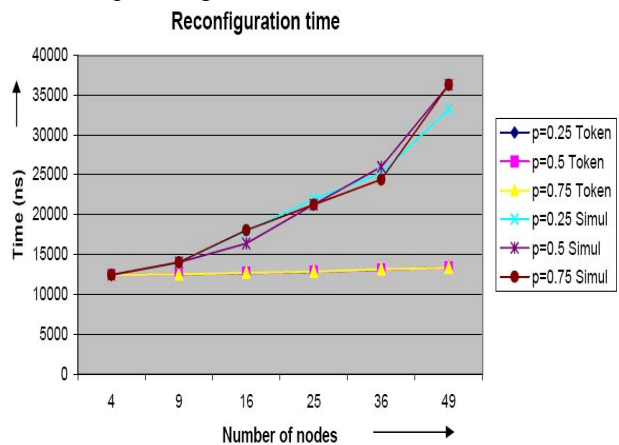


Fig. 3. Reconfiguration time for CR mechanisms.

Reconfiguration time is comparable (Fig. 3), irrespective of probability of caching paths, because very few paths satisfied both constraints. To verify the affect of variation in reconfiguration time with respect to probability, a separate corresponding set of experiments was conducted in an unconstrained environment and reconfiguration time was observed to increase for higher

probability cases. Simultaneous mechanisms take more time than token mechanisms because of increased traffic. To justify the wait interval timer value and compare the performance of approaches with the ideal case, Dijkstra's algorithm was implemented in C++. Average and SD was calculated for all-pair shortest path lengths (Fig. 4).

APL is comparable (Fig. 4) for all approaches because of the characteristic of asymptotic converging of the path length to the minimum value given sufficient time. But, token approaches result in slightly bad results in terms of path length because they take lesser time compared to simultaneous approaches. The simultaneous case with probability 0.25 takes less time than other higher probability approaches, has fewer options because of reduced search space and hence results in slightly bad solutions. This affect gets amplified because of the symmetric and parallel nature of the algorithm. The simultaneous approach with probability 0.5 achieves better solution than probability 0.75 for larger network sizes because the probability 0.75 case needs more time to converge to an optimum solution than the probability 0.5 case given much larger traffic. On the other hand, the chances of finding better solutions in the probability 0.75 case are more than any other probability case. SD of path

length is an indication of the temporal use of channels by different paths arriving at different instances of time. SD is slightly higher for the token mechanism (Fig. 4) because the token mechanism achieves suboptimal solutions as compared to simultaneous mechanisms as explained above for APL. A sudden surge in the simultaneous case with probability 0.25 is an amplified indication that it has not converged to optimum solution because of less options and time taken. The simultaneous approach with probability 0.75 seems to require more time for converging to an optimal solution for larger networks.

Being optimized for minimum path length, APPC and SD are (Fig. 5) comparable for all approaches. A slightly bad APPC solution by the token mechanism and simultaneous case with probability 0.25 can be explained along the lines of the explanation for APL above and its positive correlation with PPC in the framework used. SD of PPC is an indication of the spatial use of channels by paths over a given period of time equal to reconfiguration time. Simultaneous shows a major improvement in terms of SD of PPC (Fig. 5) as compared to the token mechanism because larger numbers of different kinds of paths are probable to arrive at a node during reconfiguration because of the comparatively larger amount of traffic. Also, the

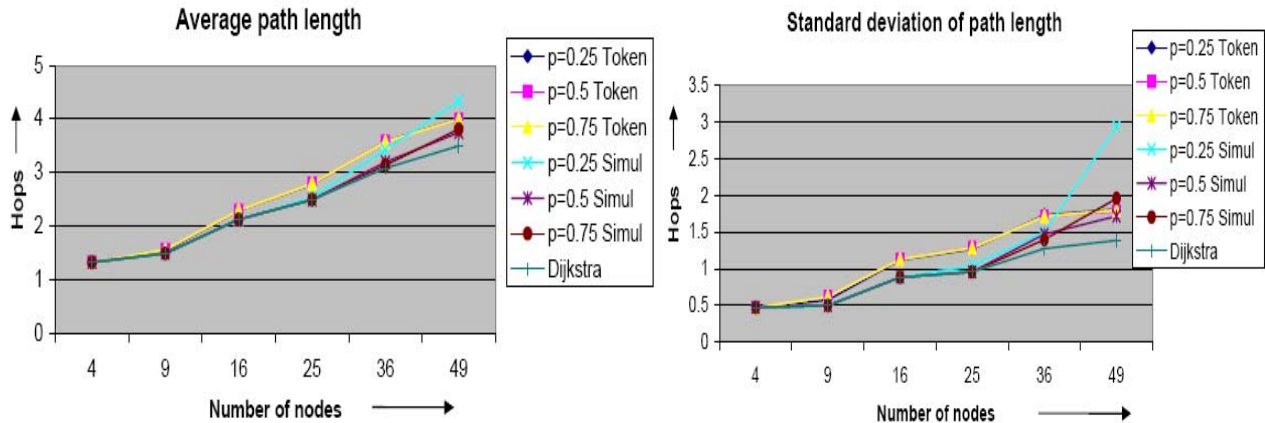


Fig. 4. Path length for constrained random mechanisms.

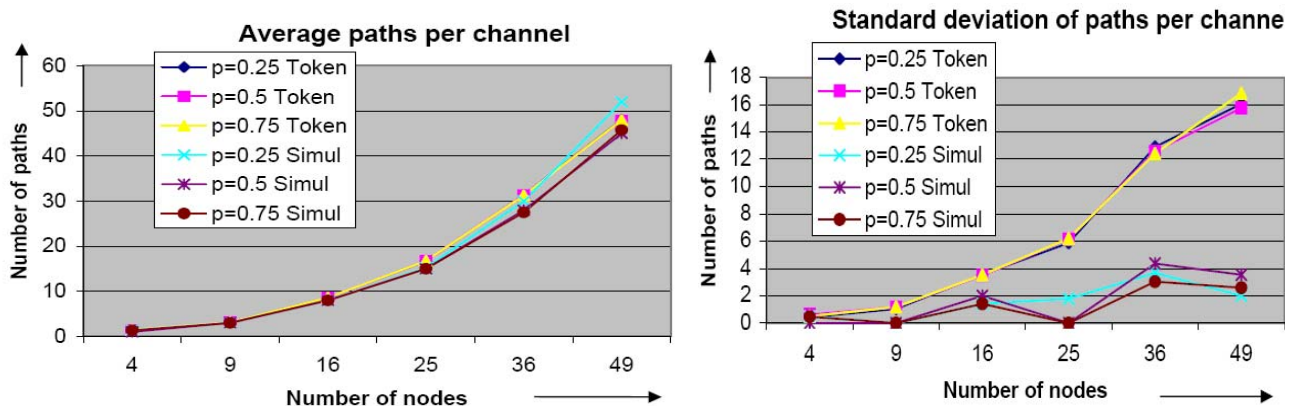


Fig. 5. PPC for constrained random mechanisms.

simultaneous approach is a symmetric algorithm running over a symmetric network, so its probability of using channels uniformly is more than that of token at the cost of larger reconfiguration time. Although the SD of the simultaneous case with probability 0.25 does not follow the trend of better solution with increased probability, it achieves suboptimal solutions in terms of APL and PPC and hence its SD is not indicative of the trend.

4. Effect of number of VCs and faults

Constrained random simultaneous approach with probability 0.25 was chosen for further investigating the effect of number of virtual channels (VCs) and faults on PE. This decision was motivated by the favorable SD of PPC of this approach, a near optimal solution in terms of other performance metrics and more scope for a scalable solution in terms of reconfiguration time.

4.1. Effect of number of VCs

Reconfiguration time increases with decreasing number of VCs (Fig. 6) because of more blocking time experienced with lower numbers of VCs. Moreover, merging is not very effective in simultaneous cases because of heterogeneous traffic. A proportionate

increase in the number of VCs does not lead to a proportionate increase in the performance, and improvement diminishes at further ends of increments. This is because of the inherent asymptotic limitations implied due to the multiplexing of VCs on a physical channel and network capacity.

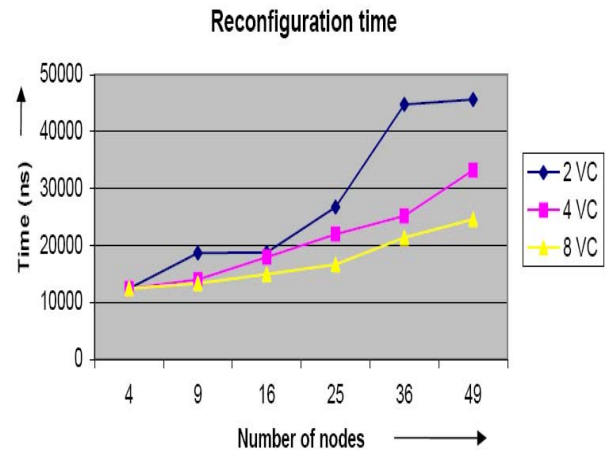


Fig. 6. Effect of VCs on reconfiguration time.

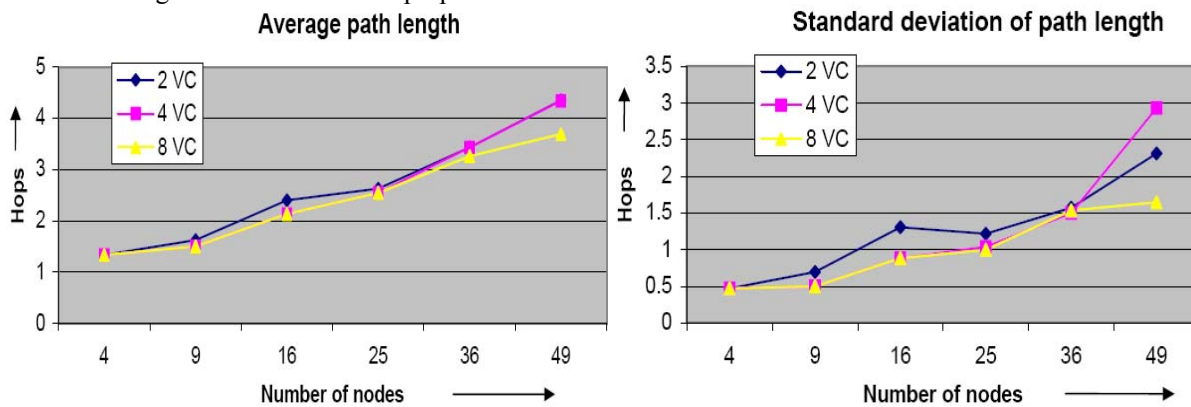


Fig. 7. Effect of VCs on path length.

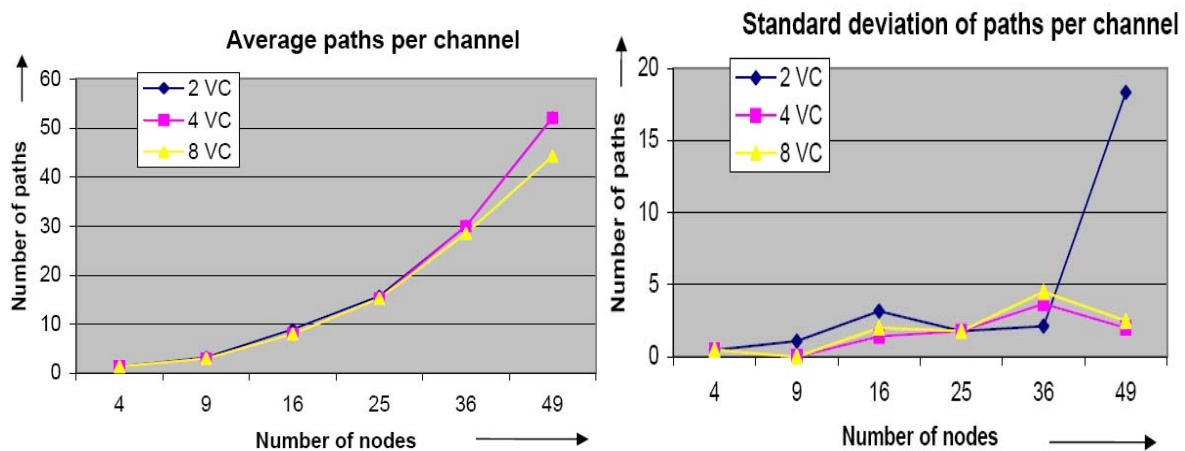


Fig. 8. Effect of VCs on PPC.

APL is better for a larger number of VCs (Fig. 7) because less congestion enables more PE packet options to reach nodes and thus allows PE to converge quickly to a near optimum solution. SD of path length was observed to get better with increasing number of VCs (Fig. 7) because of less congestion during the latter part of reconfiguration, which otherwise resulted in longer paths to arrive earlier and caused APL and SD to increase. The 2 VC case didn't strictly follow the trend and *appears* to be better than the 4 VC case for large network sizes. This may be attributed to its suboptimal solution, a fact more apparent from the SD of PPC curve (where it exhibits a much larger value than other approaches). Increasing the wait interval validated the fact that suboptimal solution is the reason behind the poor quality of paths achieved by the 2 VC case (quantitative results not presented here). Optimal solutions comparable to the 8 VC case were achieved by the 2 VC case at the expense of much larger reconfiguration time. This indicates a trade-off between the network resources used, reconfiguration time and quality of paths.

APPC improved for larger numbers of VCs (Fig. 8) because of the same reason described for variation of APL. SD of PPC was observed to get better with decreasing number of VCs (Fig. 8). But, it is a simple indication of more uniform suboptimal solutions with larger APL and APPC. The 8 VC case, despite its slightly larger SD, appears to have found better solutions by reducing APL and PPC, and thus reducing congestion, zero-load latency and power consumption. The 2 VC case didn't strictly follow the trend and this may be attributed to a suboptimal solution achieved with limited network resources, despite the larger reconfiguration time taken.

4.2. Effect of faults

This evaluation dimension becomes especially important for the proposed routing algorithm given its emphasis on fault-tolerance as a major design parameter. This analysis is done for various percentages of network nodes failing and performance of the remaining connected randomly generated topology. The 4VC configuration was chosen because of reduced hardware requirements as compared to the 8VC configuration.

Reconfiguration time, in general, decreases by increasing the number of faults (Fig. 9), with reconfiguration time highest for a fault-free network. This is because a network with faulty nodes represents an effectively smaller network topology as compared to the fault-free network, so less traffic is generated. Each data point above is a different random network topology (not k -ary 2-cube torus anymore) because of the random nature of faults, so no generic trend laws can be stated to have been observed correctly. However, some observations can be made. One is, reconfiguration time decreases less than proportionally with an increase in the percentage of faults. This is because a reduction in congestion due to reduced

traffic increasingly becomes less important as the asymptotic limitation of static network traversal and processing delay by routers becomes an increasingly significant part of the reconfiguration time.

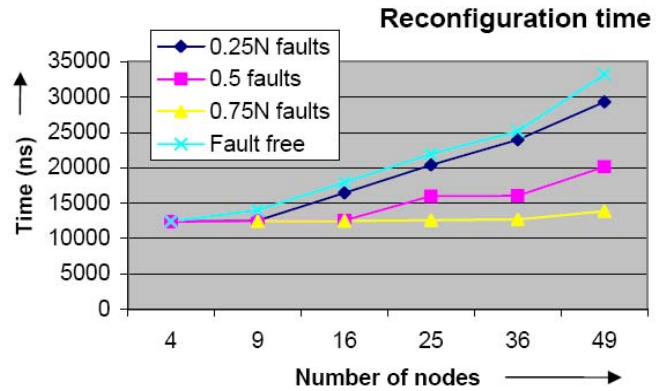


Fig. 9. Effect of faults on reconfiguration time.

APL and its SD do not strictly follow a trend with respect to percentage of faults (Fig. 10), and with respect to network sizes for configurations with 50% faults. This is because each random topology has different asymptotic limits on the achievable minimum APLs because of unavailability of shorter paths for some source-destination pairs. To validate this observation, the ideal optimum APL was calculated for the random topologies used by using the static Dijkstra's algorithm. Results of the proposed chosen algorithm followed the values and trend curves of the ideal optimum solutions for the parameter being optimized (e.g., APL). It is simply noted that the proposed algorithm achieves the optimum solution for the parameter being optimized for a wide range of fault percentages.

Both total numbers of functional source-destination pairs and total number of channels reduce with increasing number of faults in the network. But, number of source-destination pairs reduces at a much faster rate (quadratic relationship to the number of faults) than the reduction in total number of functional channels (linear relationship to the number of faults). So, APPC should reduce with increased number of faults (numerator decreases faster than denominator) compared to its starting point topology. This explains close APPC values (Fig. 11) of the 25% and 50% cases (without this observation, expected APPC values for 50% case should be slightly higher because of some positive correlation between APL and APPC) and much lower APPC of 75% case, irrespective of the randomness implied in the topology. Fault-free topology is regular and symmetric, and hence it uses a greater fraction of available channels than faulty random topologies. SD of PPC is again an indication of randomness in the underlying topology (Fig. 11). Some channels must be used by many paths and others may remain underutilized, leading to large and varying SD in different random topologies. Other explanations can be given on the lines of observations made in the analysis of APL and its SD.

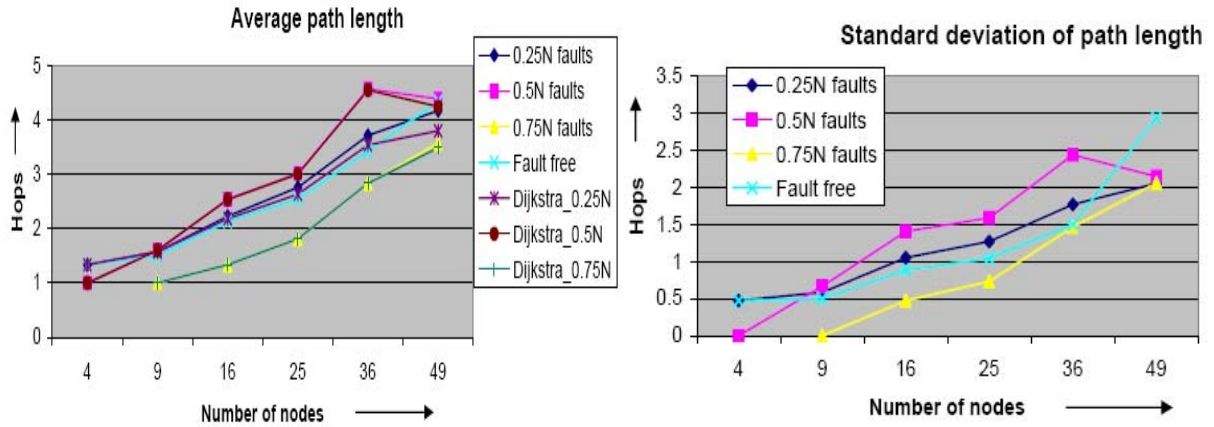


Fig. 10. Effect of faults on path length

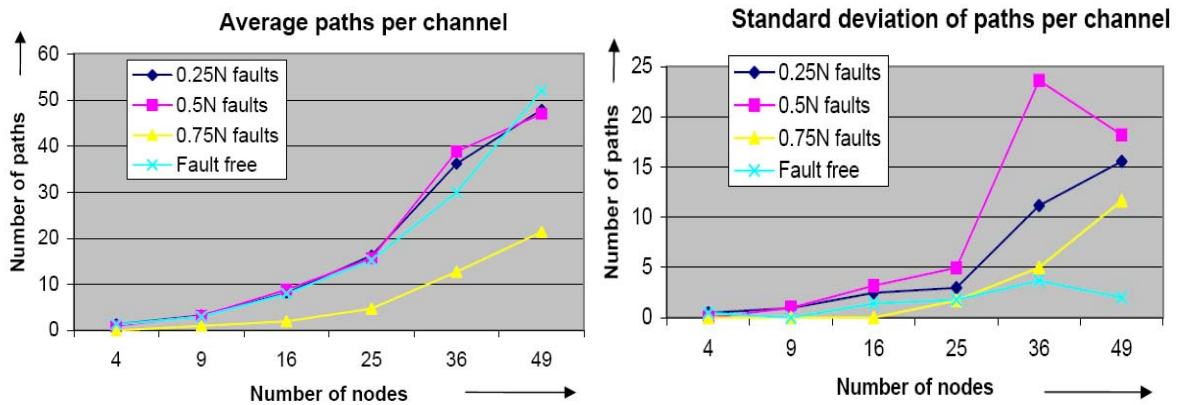


Fig. 11. Effect of faults on PPC.

5. Conclusion

This paper presented various path exploration (PE) approaches for an on-chip generic fault-tolerant routing algorithm and assessed their performance trade-offs for k-ary 2-cube NoCs. In general, the following conclusions are drawn from the presented performance evaluation. Firstly, for low reconfiguration time, the constrained random token mechanism with merging should be used. The constrained random token mechanism displays lower quality of paths, but slightly lower quality of paths may be sufficient for many applications. Secondly, constrained random simultaneous approaches should be used for low APL, SD of APL, APPC and SD of APPC. Constrained random simultaneous mechanisms require a larger reconfiguration time, but higher quality of paths may be more important for certain applications. Thirdly, increasing number of VCs results in achieving optimal results in less time. The resulting trade-off is between network resources, quality of paths and reconfiguration time. Finally, the proposed algorithm achieves optimal results in terms of quality of paths with respect to large variations in the number of faults, and reconfiguration time decreases with more faults.

So, performance of the proposed approach does not degrade with more faults.

6. References

- [1] <http://public.itrs.net>
- [2] T.M. Pinkston, and J. Shin, "Trends toward on-chip networked microsystems", *Journal of High Performance Computing and Networking*, 2005
- [3] S. Mediratta, and J. Draper, "Characterization of a Fault-tolerant NoC Router", *ISCAS*, 2007
- [4] Mediratta, S., *Communication mechanisms for Processing-In-Memory systems*, PhD dissertation, University of Southern California, 2006
- [5] C. McNairy, D. Soltis, "Itanium 2 Processor Microarchitecture.", *IEEE Micro*, 2003
- [6] J.M. Martinez, et al. "Software-Based Deadlock Recovery Technique for True Fully Adaptive Routing in Wormhole Networks", *ICPP*, 97
- [7] B. Shackelford, et al. "High-performance cellular automata random number generators for embedded probabilistic computing systems", *NASA/DoD Conference on Evolvable Hardware*, 2002