Survivable Computer Networks in the Presence of Partitioning

Y.Varoglu[1] and D.R.Avresky[1]

[1] Dept. of Electrical and Computer Eng.
NorthEastern University
Boston, MA 02215
Phone:(617) 373-3051
E-mail: avresky@neu.edu

Abstract

A small number of network components failures can cause high-speed networks (LANs) or System Area Networks (SANs) to be partitioned. This issue has been resolved with the introduction of redundant network components (ie. interfaces) into the network. A cost effective solution can be achieved with switch interconnects. Recently, such networks have been successfully implemented[6] with Myrinet networks.

In this article, we present a "F cycle ring" (FCR) architecture which can withstand up to (F+1) failures without partitioning the switch interconnect. Our simple design is scalable to any ring size, given a lower limit on the number of switches. We focus on switches with a degree of four and computation nodes of degree two. We prove that (F+1) failures can be tolerated by the interconnect with several lemmas. Some examples of FCR are shown as an implementation of a full FCR network.

Index Terms - *Partitioning, rings, meshes, torus, interconnection networks*

1 Introduction

High-Speed Local Area Networks (LANs) or System Area Networks (SANs) [14], [15] consist of hosts connected to a network interface card (NIC). Fault-tolerance becomes an important issue as the network size increases in these distributed systems. The removal of some network components (ie. switches and/or hosts) may result in additional disconnection of other components. especially in cases where the number of ports of the network components is the same throughout the network.

At the software level, many distributed algorithms can not tolerate a large set of node failures due to partitioning[12], [13]. Consequently, the designer of the network may be restricted to connect the network components in a topology that minimizes the partitioning of the network.

Recently, fault-tolerant high-speed LANs have been implemented with Myrinet network elements with constant number of ports per switch. One advantage of Myrinet (or ServerNet [15]) networks is the fact that the switched networking elements are less costly than the host interfaces. Hence, cost effective constructions of networks of switches can be achieved with Myrinet switches.

In this paper, our motivation is to connect compute nodes to switching networks to maximize the network's resistance to partitioning. Given a limit on the number of failures, a partition resistant network is a network that looses only some constant number of network components, with respect to the total number of nodes in the configuration. In case of additional network failures, an indeterminate number of nodes may be disconnected from the network.

Fault tolerant network designs have been comprehensively studied. In some cases, a 0-fault tolerant subgraph is augmented with additional node spares and links to obtain a k-fault tolerant graph [1]. The design algorithm consists of connecting each of the spare nodes to the subgraph with additional links. An example of k-fault tolerant ring is provided in [1], where k is the number of tolerated faults. In this construction, a k-ring would still contain a ring after the presence of k faults. Similar methods of graph augmenting with spare nodes have also been applied to trees [2], [3], [4], rings, meshes and hypercubes [9], [10], [11], [8].

Another method of fault tolerance consists of maintaining connectivity between network components in the presence of faults. In [5], compute nodes are connected by buses for resilience to failures. RAIN [6],[7] maintains connectivity between switches and prevents direct links between compute nodes in a ring. The network resistance to partitioning is k=2d_c-1, where d_c is the number of ports of the computation nodes. Hence the partition is limited by the node degree instead of the network size.

We present a noval approach of a switch interconnect design. Our "F Cycle Ring" switch interconnect (FCR) is a ring of F*F (or more) switches with constant node and switch degrees that can tolerate up to (F+1) faults without partitioning the network. The structure of the paper is as follows: in section 2 and 3, we present our network model and the limitations due to the switch degree of 6. Section 4 defines a FCR switch interconnect. The proof that FCR does not partition in the presence of (F+1) faults is given in section 5. Examples of FCR with computation nodes are shown in section 6. We discuss our validation in section 7 and present our conclusions in the last section.

2 Network Model

2.1 Our Assumptions

In our model, we are concerned with homogeneous distributed scalable systems. Each switch has the same number of connections. The node degree is also the same for all the computation nodes. The distributed network consists of S switches and S compute nodes. However, the number of computation nodes N is not limited to S. All our results still hold for N $\geq$ S.

The traffic in the system is only routed by the interconnected switches. The nodes are passive; they can only generate and drain the messages destined for themselves.

A switch s has the following qualities or restrictions:

1. s has a unique ID (a number).
2. s has a constant degree of 6, $d_s = 6$ (i.e. homogeneous system); The switch degree is the number of network ports of a switch.
3. s fails correctly; if s crashes, all of its links do not receive/forward packets.

A node c has the following qualities or restrictions:

1. c is connected to switches only.
2. c has a constant degree of 2, $d_c = 2$; c's degree is the number of interfaces at a node.
3. c does not forward packets. In other words, if a node c_1 sends a packet to node c_2, the packet travels along the path solely connected through the switching interconnect.

3 Limitations

The switch degree, $d_s = 4$ (2 extra ports reserved for computation nodes), restricts the minimum number of disconnected switches in the network. Failures of 4 switches that are neighbors of switch s_i disconnects s_i from the network. Similarly, failures of 6 switches can remove 2 extra switches from the network.

Lemma 3.1 *In general, as shown in Figure 1, failures of (2f+2) switches removes an additional of f switches from the interconnect, where f=0,1,2,.. and* $f \leq \frac{S-2}{3}$.

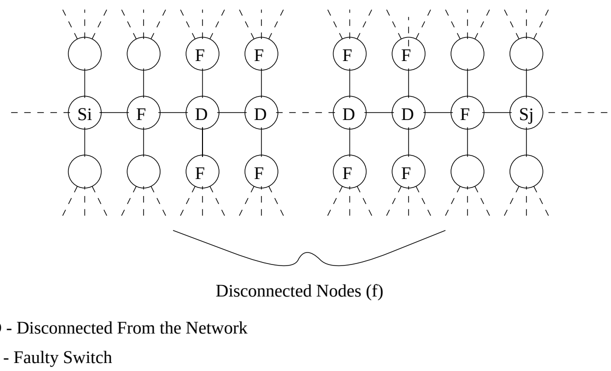

Figure 1: Disconnecting f switches with (2f+2) switch failures.

Theorem 3.1 *For any switch interconnect with F*F switches and* $2F^2$ *edges, where each switch degree is 6 (of which 2 are reserved for computation nodes connections), failures in (2f+2) switches result in the removal of f switches, and hence (3f+2) switches from the network, where f=0,1,2...*

Proof: Proof is by Lemma 3.1. Failures of (2f+2) switches can disconnect an additional f switches.

4 F Cycle Circle (FCR) Approach

In this section, we consider the design of the switch interconnect FCR of S switches of degree d_s=6, connected to S computation nodes of degree d_c=2, while minimizing the partitioning of the network in the presence of switch failures. This FCR consists of F*F switches with cycles of length (F+1) and can tolerate an arbitrary of (F+1) faults in the network without partitioning the nodes into sets of non-constant size (with respect to S), where F=$\sqrt{S}$. Lemma 3.1 is a special case of

(F+1) failures. Hence, the switch interconnect is partition resistant to (F+1)≥(3f+2) failures, where f≤$\lfloor \frac{F-1}{3} \rfloor$ and f=0,1,2,... .

Definition 4.1 *FCR_F is the graph with S=F*F switches and 2F^2 edges constructed as follows. Let (s_1,s_2,...,s_S) be some ordering of switches. For each switch s_i, where 1≤i≤S, introduce an edge connecting s_i to each $s_{(i-1)modS}$ and $s_{(i+1)modS}$. These connections form a single loop system, namely C_S. Additionally, join every switch s_i of C_S to 2 switches that are at a distance of length F from s_i in C_S; that is, every switch s_i connects to the switch $s_{(i-F)modeS}$ and to $s_{(i+F)modeS}$. This last set of connections creates two cycles of length (F+1) between the switches numbered $s_{(i-F)modeS}$ to s_i and s_i to $s_{(i+F)modeS}$.*

Figure 2 shows the general form of connections in a FCR.

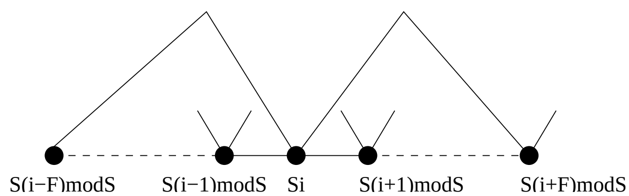

Figure 2: Connections for a switch s_i in C_S, 1≤i≤S.

4.1 Base Case with 5 Switches

A FCR interconnect that tolerates 2 failures without the network partitioning is shown in Figure 3. This topology is a clique.

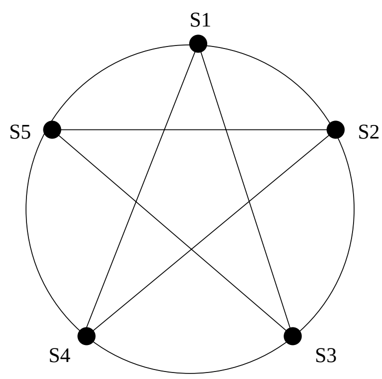

Figure 3: A (2*2)+K, FCR that can tolerate 3 faults, where K=1.

4.2 FCR for F ≥ 3

A F*F FCR topology is constructed by interconnecting all switches in a cycle of length F. Moreover, any size of (F*F+K) FCR can be created, where K≥0. The (F*F+K) FCR will still be (F+1) faults resilient to partitioning and can tolerate (F+1+L) failures without partitioning, where K≥L(2F+1) and L≥1. Some examples of F*F FCRs are shown in Figure 4.

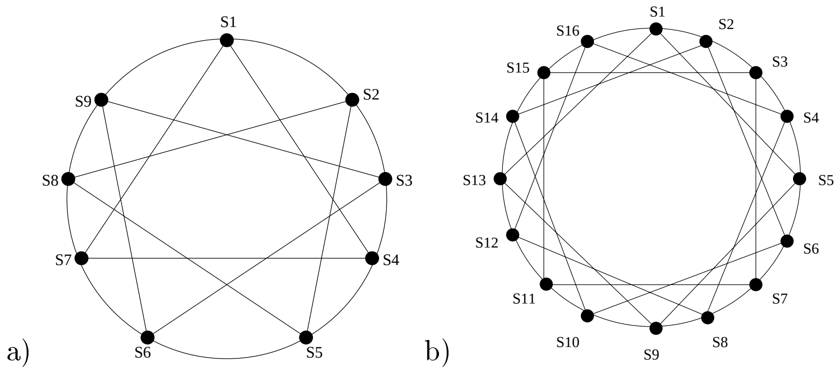

Figure 4: F*F FCR examples with S=9 (a) and S=16 (b).

5 Correctness

Theorem 5.1 *A F*F FCR can tolerate up to (F+1) faults without partitioning the network.*

Proof: We prove the 3 possible cases. Let Q be any (F+1)-fault set in FCR_F. Q consists of subfault sets Q_1, Q_2,..,Q_{F+1}. The following properties describe Q:

1) All faulty switches in each Q_i are a connected set of switches along the perimeter of FCR_F, where 1 ≤ i ≤ (F+1).

2) Each Q_i is a distinct set; that is the switches in Q_i are not neighbors of any switch of Q_j, on the perimeter of FCR_F, $Q_i \neq Q_j$.

Q describes a set of fault subsets Q_1,Q_2,..,Q_{F+1} of FCR_F as shown in Figure 5. There are 3 cases to consider depending whether or not any fault subset Q_i contains (F+1) faults or not.

Case 1: $|Q_i| \leq$ (F-1) for all 1 ≤ i ≤ F. Let s_i and s_j be 2 switches of FCR_F that are adjacent to the switches of Q_i but are not in Q_i and therefore not in Q. There is an "internal" edge between s_i and s_j since there is always an edge between 2 switches at a distance of F. Thus the switches of Q are spanned by this edge. The remaining edges along the perimeter of FCR_F form a closed loop through all the switches of FCR_F as seen in Figure 5. Thus, FCR tolerates (F-1) faults.

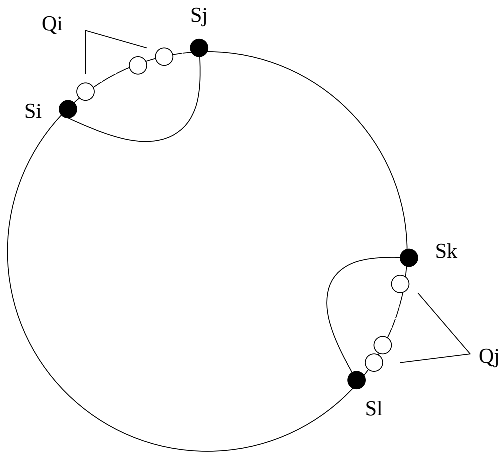

Figure 5: Faulty switches in Q for case 1.

Case 2: (F+1)$\geq|Q_i|\geq$F. This is the case shown in 6 where the disconnected switches due to Q_i are not spanned by any edge of FCR_F. Let s_i and s_j to be the adjacent switches to Q_i but not in Q. We define the sequence Q_i=($s_{(i+1)}$, $s_{(i+2)}$,...,$s_{(j-1)}$). Then, s_i (or s_j) has 2 (or 1 depending if s_k is faulty) functional connections to 2 distinct switches s_k and s_l, where i$\neq$j$\neq$k$\neq$l, k=i-1 (or j-1) and l=i-F (or j-F). Let $|Q^{'}|$ = |Q| - $|Q_i|$. Then $|Q^{'}|$=1 or 0. In case $Q^{'}$=1, the faulty switch in $Q^{'}$ can be s_k or s_l. Let s_f be the faulty switch. If s_f=s_k (or s_l), then s_i (or s_j) is still connected by the internal edge (or by the edge to s_k and internal edges connected to s_k) to the network. Since $s_f \neq s_k$ (or s_l), and since there are no other faults along C_S, then s_f is spanned by an internal edge or an edge along C_S and an internal edge. Thus, the network is still connected with F faults.

In case $Q^{'}$=0, then Q_i=Q and s_i (or s_j) is still connected to the network via its 2 edges; for example, the holes created by Q_i are spanned by the clockwise path between s_i and s_j, along the perimeter of FCR_F. Hence the network is still connected with (F+1) faults.

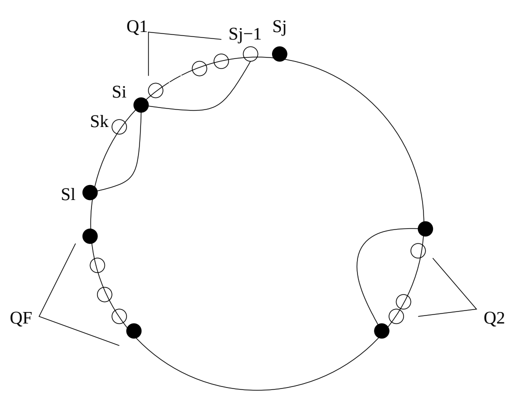

Figure 6: Faulty switches in Q for case 2.

Case 3: |Q|=$|Q_i|$=(3f+2)≤(F+1), where f≤$\lfloor\frac{F-1}{3}\rfloor$ and f=0,1,2... This is the case corresponding to theorem 1 as shown in Figure 7 and in Figure 8. We define 3 sequences Q_v, Q_w and Q_y where Q_v=($s_{vi(+1)}$, $s_{v(i+2)}$,...,$s_{v(j-1)}$), Q_w=($s_{w(i+1)}$, $s_{w(i+2)}$,...,$s_{w(j-1)}$) and Q_y=($s_{y(i+1)}$, $s_{y(i+2)}$,...,$s_{y(j-1)}$), respectively. Similarly, let (s_{vi},s_{vj}), (s_{wi},s_{wj}), and (s_{yi},s_{yj}) be the switch tuples that are adjacent to the faulty switches in Q_v, Q_w and Q_y, respectively. Then, $|Q_v|$=f<F, $|Q_w|$=(f+2)<F and $|Q_y|$=f<F. Then, since there are no more faults in FCR_F, there is a closed loop p bypassing Q_v, Q_w and Q_y, where p=(s_{vi},$s_{v(i-1)}$,...,$s_{v(j-F)}$,s_{vj},$s_{v(j+1)}$,...,s_{wi}, ,$s_{w(i-1)}$,...,$s_{w(j-F)}$,s_{wj},$s_{w(j+1)}$,..., s_{yi},$s_{y(i-1)}$,...,$s_{y(j-F)}$,s_{yj},$s_{y(j+1)}$...,s_{vi}). Hence the interconnect is connected.

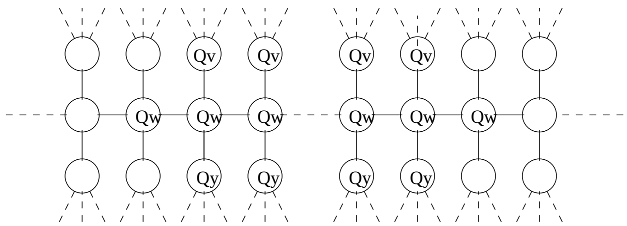

Figure 7: Faulty switches in the network for case 3.

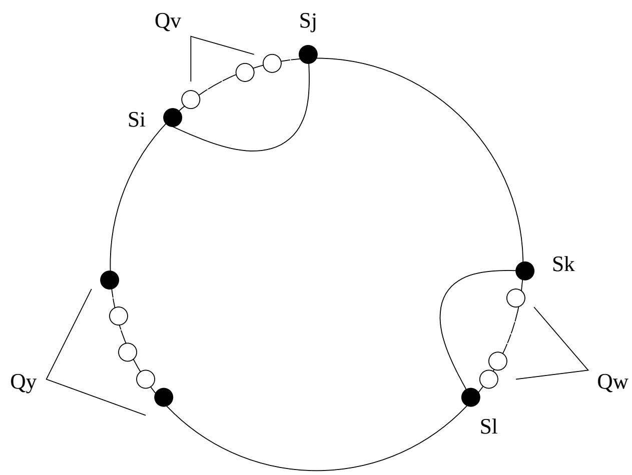

Figure 8: Faulty switches in Q for case 3.

Lemma 5.1 *A (F*F+K) FCR can tolerate up to (F+1) faults without partitioning the network, where K≥0.*

Proof: We prove by induction.

Base case with K=0:

The proof was given in Theorem 5.1.

Case for K+1, K>0

Let FCR_{F_K} be a FCR with (F*F+K) switches that tolerates up to (F+1) faults without partitioning, where K≥0. In FCR_{F_K}, each switch s_i is connected to $s_{(i-1)modeS}$, $s_{(i+1)modeS}$, $s_{(i-F)modeS}$ and $s_{(i+F)modeS}$. Moreover, let a $FCR_{F(K+1)}$ with (F*F+K+1) switches be also constructed in this fashion. Let Q be any (F+1)-fault set in $FCR_{F(K+1)}$ which partitions the network. Since FCR_{F_K} is built in the same fashion as a $FCR_{F(K+1)}$, we can apply Q to FCR_{F_K} to partition the network. However, Q cannot partition FCR_{F_K} with (F+1) faults. Therefore, $FCR_{F(K+1)}$ must be partition resistant to (F+1) faults.

6 Construction

The FCR_F switch interconnect is designed as in section 4.1. The computation node c_i, where 1≤i≤S, is added the following way; connect each node c_i to the switches s_i and $s_{(i+F)modS}$. In other words, the computation nodes are linked parallelly to the internal edges of the FCR_F. We note that the nodes need not exist between every switch. There can be any number of computation nodes between 2 switches. The presence of computation nodes does not affect the resistance to partitioning of the network since the switch interconnect guarantees the partition tolerance. Hence, the designer may add/remove nodes on top of the switch interconnect. Some examples of FCR_F with computation nodes are shown in Figure 9.

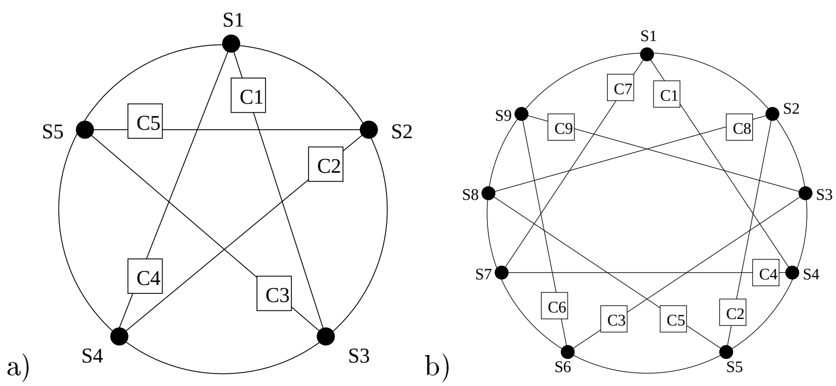

Figure 9: FCR examples with computation nodes with S=5 (a) and S=9 (b).

7 Validation

We implemented a C++ program to validate various size of FCR_Fs. For each FCR_F, our simulation constructs a FCR_F network and fails iteratively some switches. A failed switch is a switch that is completely disconnected from the network and hence, does not receive or forward packets on any of its links. As a first iterative step, the simulation fails (F+1) switches in the FCR_F. Then, the simulation obtains the closure of each partition in the network. The iterative steps are repeated in order to exhaustively apply all switch failure combinations to the FCR_F. Using the above method, we simulated up to 50 nodes FCR_Fs, and verified that each FCR_F does not partition in the presence of (F+1) faults.

Each FCR_F simulation is outputted in column format. The first column contains the faulty switches in a continuous binary stream of length S, where S is the number of switches. Each faulty switch is denoted by a 1 and a non-faulty switch is denoted by 0. The subsequent columns contain the sets of connected switches. All the sets are disjoint. We obtained the following output from our simulation of a 38 node system with seven failed nodes.

<...>

Line 1: 00000000000000000000100001011010100001, $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}, s_{16}, s_{17}, s_{18}, s_{19}, s_{20}, s_{22}, s_{23}, s_{24}, s_{25}, s_{30}, s_{34}, s_{35}, s_{36}, s_{37}\}, \{s_{27}\}, \{s_{32}\}$

00000000000000000000100001010110100001, $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}, s_{16}, s_{17}, s_{18}, s_{19}, s_{20}, s_{22}, s_{23}, s_{24}, s_{25}, s_{29}, s_{34}, s_{35}, s_{36}, s_{37}\}, \{s_{27}\}, \{s_{32}\}$

00000000000000000000100001010010110001, $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}, s_{16}, s_{17}, s_{18}, s_{19}, s_{20}, s_{22}, s_{23}, s_{24}, s_{25}, s_{29}, s_{30}, s_{35}, s_{36}, s_{37}\}, \{s_{27}\}, \{s_{32}\}$

00000000000000000000100001010010101001, $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}, s_{16}, s_{17}, s_{18}, s_{19}, s_{20}, s_{22}, s_{23}, s_{24}, s_{25}, s_{29}, s_{30}, s_{34}, s_{36}, s_{37}\}, \{s_{27}\}, \{s_{32}\}$

00000000000000000000100001010010100101, $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}, s_{16}, s_{17}, s_{18}, s_{19}, s_{20}, s_{22}, s_{23}, s_{24}, s_{25}, s_{29}, s_{30}, s_{34}, s_{35}, s_{37}\}, \{s_{27}\}, \{s_{32}\}$

00000000000000000000100001010010100011, $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}, s_{16}, s_{17}, s_{18}, s_{19}, s_{20}, s_{22}, s_{23}, s_{24}, s_{25}, s_{29}, s_{30}, s_{34}, s_{35}, s_{36}\}, \{s_{27}\}, \{s_{32}\}$

<...>

As expected, by Theorem 3.1, seven failed switches in the system disconnects 2 additional switches. For example, in line 1, the faulty switches are s_{21}, s_{26}, s_{28}, s_{29}, s_{31}, s_{33}, s_{35} and s_{38}. These failed switches remove two additional switches from the network, namely s_{27} and s_{32}.

8 Conclusion

In this article, we proposed a (F+1)-fault partition tolerant ring architecture. FCR is a scalable design with a constant switch (d_s=6) and node (d_c=2) degrees. Unlike earlier partition tolerant architectures, FCR can scale indefinitely while increasing the tolerance to partitioning. With the additional of K$\geq$ L(2F+1) switches, the network can be reconfigured to resist a (F+1+L) faults in the system.

Another advantage of FCR is the way the computation nodes are added to the system. Each node N is attached to 2 switches. The network designer can reduce or increase the number of nodes without affecting the partition resistance of the switch interconnect. This in effect, lowers the cost of the network. Moreover, FCR is complementary to the distributed algorithms which can not tolerate a loss of a large number computation nodes.

References

[1] J. P. Hayes, "A Graph Model for Fault-Tolerant Computing Systems" *IEEE Trans. on Computers,* Vol. 25, No.9, pp. 875-884, 1976.

[2] S. Dutt and J.P. Hayes, "On Desiging and Reconfiguring k-Fault Tolerant Tree Architectures" *IEEE Trans. on Computers,* Vol. 39, No.4, pp. 490-503, 1990.

[3] S. Dutt and J.P. Hayes, "Designing Fault-Tolerant Systems Using Auto-morphisms" *Journal of Parallel and Distributed Computing,* Vol. 12, No.3, pp. 249-268, 1991.

[4] S. Dutt and J.P. Hayes, "Some Practical Issues in the Design of Fault-Tolerant Multiprocessors" *IEEE Trans. on Computers,* Vol. 41, No.5, pp. 588-598, May 1992.

[5] H.K. Ku and J.P. Hayes, "Connective Fault-Tolerance in Multiple-Bus Systems" *Trans. on Parallel and Distributed Systems,* Vol. 8, No.6, pp. 574-586, 1997.

[6] P. LeMahieu, V. Bohossian and J. Bruck, "Fault-Tolerant Switched Local Area Networks" *IEEE Proc. of the International Parallel Processing Symposium* , pp747-751, 1999.

[7] V. Bohossian, C. C. Fan, P. LeMahieu, M. D. Riedel, L. Xu and J. Bruck, "Computing in the RAIN: A Reiable Array of Independent Nodes" *TBD* , .

[8] J. Bruck, R. Cypher and C. Ho, "Fault-Tolerant Meshes and Hypercubes with Minimal Number of Spares" *IEEE Trans. on Computers,* Vol. 42, No.9, pp. 1089-1104, 1993.

[9] F.T. Boesch and A. P. Felzer, "A General Class of Invulnerable Graphs" *Networks,* Vol. 2, pp. 261-283, 1972.

[10] F.T. Boesch and R. Tindell, "Circulants and their Connectivities" *IEEE Journal of Graph Theory,* Vol. 8, pp. 487-499, 1984.

[11] F.T. Boesch and J. F. Wang, "Releiable Circulant Networks with Minimum Transmission Delay" *IEEE Trans. on Circuits and Systems,* CAS-32(12), pp. 1286-1291, 1985.

[12] T.H. Cormen, C.E. Leiserson, R.L. Rivest, "Introduction to Algorithms. " *McGrawHill,* 1991.

[13] N. A. Lynch, "Distributed Algorithms. " *Morgan Kaufmann Publishers, Inc.,* 1997.

[14] N. J. Boden, D. Cohen, R.E. Felderman, A.E. Kulawik, C.L. Seitz, J. Seizovich, and W. Su, "Myrinet - a gigabit per second local area network. " *IEEE Micro,* pp.29-36, February 1995.

[15] D. Garcia, and W. Watson, "Servernet II: Lecture Notes in Computer Science. " In *Proc. of the Workshop on Parallel Computer Routing and Communications*, pp.119-136, Springer, June 1997.

Appendix

A FCR_F is partition tolerant to (F+1) failures and disconnected switches (due to Lemma 3.1). A disconnected switch is considered as a failed switch. There are at most f such switches in (F+1) failed switches. In other words, if (3f+2) $\leq$ (F+1) holds, where f=$\lfloor \frac{F-1}{3} \rfloor$ and f=0,1,2,... . Table 8 shows some values of f and F.

f	(3f+2)	(F+1)	$\lfloor \frac{F-1}{3} \rfloor$
1	5	5	1
2	8	8	2
3	11	11	3
4	14	14	4
5	17	17	5
6	20	20	6
7	23	23	7
8	26	26	8
9	29	29	9

Table 1: Some examples of F and f