Self-Healing in Binomial Graph Networks

Thara Angskun[1], George Bosilca[1], and Jack Dongarra[1,2,3]

[1] Department of Computer Science, The University of Tennessee, Knoxville, USA
[2] Computer Science and Mathematics Division, Oak Ridge National Laboratory, USA
[3] Computer Science and Mathematics Schools, The University of Manchester, UK
{angskun, bosilca, dongarra}@cs.utk.edu

Abstract. The number of processors embedded in high performance computing platforms is growing daily to solve larger and more complex problems. However, as the number of components increases, so does the probability of failure. The logical network topologies must also support the fault-tolerant capability in such dynamic environments. This paper presents a self-healing mechanism to improve the fault-tolerant capability of a Binomial graph (BMG) network. The self-healing mechanism protects BMG from network bisection and helps maintain optimal routing even in failure circumstances. The experimental results show that self-healing with an adaptive method significantly reduces the overhead from reconstructing the networks.

1 Introduction

Recently, several high performance computing platforms have been installed with more than 10,000 CPUs, such as Blue-Gene/L at LLNL, BGW at IBM and Columbia at NASA [1]. However, as the number of components increases, so does the probability of failure. To satisfy the requirements of such a dynamic environment (where the available number of resources is fluctuating), a scalable and fault-tolerant communication framework is needed. The communication framework is important for both runtime environments of MPI libraries and the MPI libraries themselves. In general, the communication framework is based on a logical network topology.

There are several existing logical network topologies that can be used in high performance computing (HPC). Whereas a fully connected topology is good in terms of fault-tolerance and point-to-point performance, it does not exhibit any scalable properties due to its high degree. The bidirectional ring topology is more scalable, but it is hardly fault-tolerant. Hypercube [2] and its variants [3–10], FPCN [11], de Bruijn [12] and its variants [13,14], Kautz [15] and ShuffleNet [16] have a number of node restrictions. They are either not scalable or not fault-tolerant. The Manhattan Street Network (2D Torus) [17] is more flexible (no restriction in number of nodes) than Hypercube-like topologies. However, it has a much higher average hop-distance. Variants of k-ary tree, such as Hierarchical Clique (HiC) [18] and k-ary sibling tree (Hypertree [19]) used in SFTP [20, 21], are scalable and fault-tolerant. They are good for both unicast and broadcast

messages. However, all nodes in their topologies are not equal (i.e. the resulting graph is not regular). Topologies, used in structured, peer-to-peer networking based on distributed hash tables such as CAN [22], Chord [23], SkipNet [24], Kademlia [25], Viceroy [26], Pastry [27] and Tapestry [28], are also scalable and fault-tolerant. They were designed for resource discovery in highly dynamic environments. Hence, they may not be efficiently used in HPC owing to the overhead for managing highly dynamic applications.

Binomial graph (BMG) [29] provides desirable topological properties in terms of both scalability and fault-tolerance for high performance computing such as regular graph (every node has the same degree), low diameter, low cost factor, low message traffic density, low fault-diameter, strongly resilient and good optimal probability in failure cases.

BMG is an undirected graph $G{:=}(V,E)$ where V is a set of nodes (vertices); $|V| = N$; and E is a set of links (edges). Each node i, where $i \in V$ and $i{=}0,1,...,N\text{-}1$, has links to a set of nodes U, where $U{=}\{i{\pm}1, i{\pm}2, ..., {\pm}2^k | 2^k \leq N\}$ in circular space, i.e., node i has links to a set of clockwise (CW) nodes $\{(i{+}1) \bmod N, (i{+}2) \bmod N, ..., (i{+}2^k) \bmod N \mid 2^k \leq N\}$ and a set of counterclockwise (CCW) nodes $\{(N{+}i\text{-}1) \bmod N, (N{+}i\text{-}2) \bmod N, ..., (N{+}i\text{-}2^k) \bmod N \mid 2^k \leq N\}$. The structure of BMG can also be classified in the Circulant graph family[4]. A Circulant graph with N nodes and jumps $j_1, j_2, ..., j_m$ is a graph in which each node i, $0 \leq i \leq n-1$, is adjacent to all the vertices $i \pm j_k \bmod N$, where $1 \leq k \leq m$. BMG is a Circulant graph where j_k is the power of 2 that $\leq N$. For a BMG size N (having N nodes), each node has a degree δ (the number of neighbors) as shown in Equation (1).

$$\delta = \begin{cases} (2 \times \lceil \log_2 N \rceil) - 1 & \text{For } N = 2^k, \text{where } k \in \mathbb{N} \\ (2 \times \lceil \log_2 N \rceil) - 2 & \text{For } N = 2^k + 2^j, \text{where } k, j \in \mathbb{N} \wedge k \neq j \\ 2 \times \lceil \log_2 N \rceil & \text{Otherwise} \end{cases} \quad (1)$$

Fig. 1(a) illustrates an example of a 12-node binomial graph. The lines represent all connections in the network. The other way to look at the binomial graph is that it is a topology, which is constructed from merging all necessary links in order to create binomial trees from each node in the graph. Fig. 1(b) shows an example of a binomial tree when node 0 is the root node. The arrows point in the direction of the leaf nodes.

This paper presents a reliability analysis and self-healing capability of the Binomial graph (BMG) networks. The reliability analysis is done using a discrete event simulation. The result indicates a potential of network bisection when the number of failed nodes is more than or equal to the degree δ (i.e. BMG is $\delta - 1$ node fault-tolerance). The self-healing BMG, introduced in this paper, helps protect BMG from network bisection and maintain optimal routing even in failure circumstances. The structure of this paper is as follows: Section 2 describes the reliability analysis of the BMG. The self-healing BMG algorithm

[4] The family of Circulant graphs includes fully connected, ring, Recursive Circulants [30] and Midimew [31].

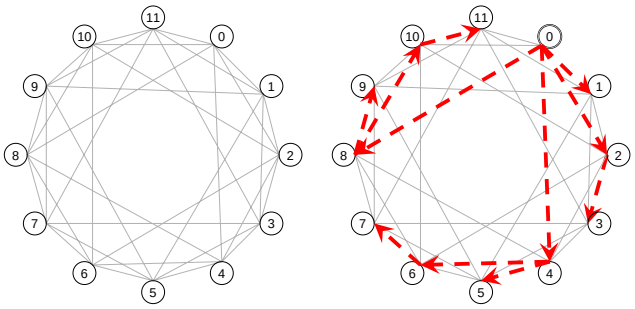


Fig. 1. Binomial graph structure. (a) 12-node BMG. (b) Binomial tree from node 0.

is discussed in section 3. Section 4 presents the experimental evaluations, followed by conclusions and future work in section 5.

2 Reliability Analysis of BMG

The reliability analysis is based on a discrete event simulation. This section presents the description of simulation as well as results from the simulation.

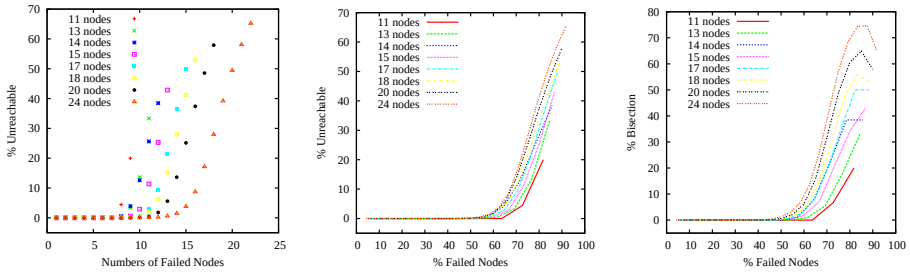
2.1 BMG Reliability Simulation

The reliability of BMG is defined as its ability to maintain an operation over a period of time t, i.e., the reliability $R(t)= Pr$(the network is operational in [0,t]). The BMG is "*operational*", if it can successfully deliver messages from any source to any alive destination even in the case where some intermediary nodes have failed. Due to the fact that multicast and broadcast messages in failure circumstances rely on unicast messages [29], this simulation focuses on reliability of the unicast routing.

The unicast messages in BMG size N are simulated by sending messages from all possible sources (S) to all possible destinations (D), where $S \neq D$. Fortunately, BMG is a vertex symmetric graph (i.e. a graph which looks the same viewed from any node). Thus, the simulation cases for normal circumstances can be reduced from $N \times (N-1)$ to $(N-1)$ cases. During the failure circumstance, the F failed nodes are obtained from combinations of all possible N nodes, i.e., $\binom{N}{F}$ where the source and destination nodes are not one of the failed nodes. Hence, there are $\binom{N-2}{F}$ simulation cases for each unicast transmission.

The total number of simulation cases of unicast message transmission (T) for N nodes of the BMG with F failed nodes is given by

$$T = (N-1) \times \binom{N-2}{F} = \frac{(N-1)!}{(N-F-2)!F!}.$$



(a) (b) (c)

Fig. 2. Percentage of unreachable and bisection in failure circumstances.

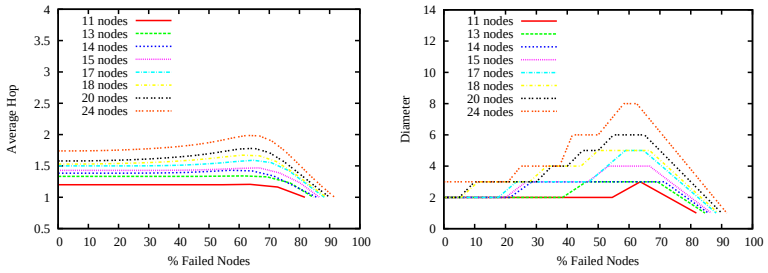
The transmission of unicast messages is considered successful if the messages can reach the destination. This means that the network can deliver messages even in the presence of failures in the routing path. If there are U unreachable cases due to network bisection, the percentage of unreachable cases (P) is defined by

$$P = (\frac{U}{T}) \times 100.$$

2.2 Simulation Results and Analysis

The results were obtained by simulating all possible cases as described in the previous section. The simulation results were obtained from BMG networks of size 11, 13, 14, 15, 17, 18, 20 and 24. All these BMG topologies have the same degree ($\delta = 8$). Fig. 2(a) illustrates that destinations become unreachable when the number of failed nodes is more than or equal to eight. However, a percentage of unreachable cases is significant when there are more than 50% of failed nodes as shown in Fig. 2(b) because the percentage of network bisection rapidly increases when the number of failed nodes is more than 50% as shown in Fig. 2(c). Not only did the failed nodes affect the reliability of the BMG, but they also affect the average hop and the diameter of the BMG. Fig. 3 illustrates the effect of failed nodes on the average hop and diameter for the remaining nodes in the case where the logical topology is constant. It indicates that the failed nodes have an effect on the average hop and the diameter, especially on large number of nodes. The average hop marginally increases when the number of failed nodes increases. While the diameter rapidly increases when the number of failed nodes increases. Eventually, both values will reduce to one when only two nodes are left in the BMG.

These simulations reveal potential problems of network bisection and a decrease in routing performance when the network has a high percentage of failed nodes. Fortunately, these problems can be prevented by a self-healing capability as discussed in the next section.



(a) (b)

Fig. 3. Average hop and diameter in failure circumstances. (a) Average hop. (b) Diameter.

3 Self-healing capability of BMG

This section presents the self-healing capabilities of BMG as a solution to prevent potential problems of network bisection and a decrease in routing performance when the network has a high percentage of failed nodes. Section 3.1 describes methods that are used to recover the BMG, while section 3.2 discusses the appropriate time to perform the recovery.

3.1 Self-Healing Methods

There are two methods presented in this section. The first approach is called the *naive* method. This method destroys the original network and reconstructs the BMG with the remaining nodes. The second method is called *adaptive* method. It only destroys and reconstructs the links that are different between the original BMG and the BMG after excluding of all failed nodes.

Naive Method This is the simplest method to reconstruct the BMG topology. Suppose there are F nodes in BMG size N. There are two steps involved in this method. The first step removes all existing links. The second step establishes all connections of BMG size $N - F$. For each link in the BMG, a node that has a higher ID will initiate the connection to the node that has a lower ID. The total number of removed links in the first step is dependent upon the location of failed nodes. It may vary from $\left[\frac{\delta_N \times N}{2}\right] - \left[(\frac{(F-1)\times F}{2}) + ((\delta_N - (F-1)) \times F)\right]$ in case of connected failed nodes, to $(\frac{\delta_N \times N}{2}) - (\delta_N \times F)$ in case of completely separated failed nodes. The total number of added links of the second step is $\frac{\delta_{N-F} \times (N-F)}{2}$. The δ_N is a degree of BMG size N, while the δ_{N-F} is a degree of BMG size $N - F$. The total number of involved links in this method is the summation between the number of removed links in the first step and the number of added links in the second step. In terms of implementation, this approach requires each node to maintain original <ID, Size>, current <ID, Size> and a list of failed nodes. They are required in the self-healing procedure and the message routing.

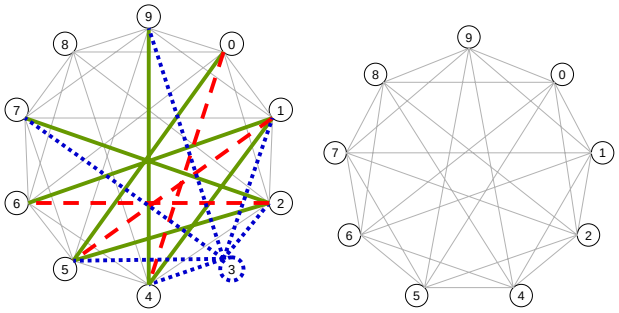


Fig. 4. Self-healing BMG using adaptive method. (a) BMG when node ID 3 failed. (b) BMG after the self-healing procedures.

Adaptive Method This method only removes and adds the links that are different between BMG size N and BMG size $N - F$. Fig. 4(a) illustrates a self-healing procedure of a 10-node BMG using the adaptive method when node 3 failed. When the node 3 failed, all connections that associate with node 3 (represented with dot lines) will be disconnected. A neighbor of node 3 may start the self-healing procedure (dependent upon the frequency as discussed in section 3.2) by broadcasting messages [5] to all nodes. Then each node calculates the different links between BMG size N and BMG size $N - F$. For each connection that needs to be added (represented with thick solid lines), the higher IDs initiate the connections to the lower IDs. All the unnecessary links (represented with dash lines) will be removed. Fig. 4(b) illustrates the BMG topology after the self-healing procedure is completed. All node IDs represented in this figure are IDs according to the original BMG. This method also requires each node to maintain original <ID, Size>, current <ID, Size>, and the list of failed nodes. They are used in the self-healing procedure and the message routing.

A function for calculating added links and removed links of the adaptive method is shown in Algorithm 1. The input variable of this function consists of $\mathbf{N}$, $\mathbf{myID_{org}}$, **dead** and $\mathbf{N_{dead}}$. The $\mathbf{N}$ is the size of the original BMG. The $\mathbf{myID_{org}}$ is a node ID in the original BMG. The **dead** is a list of dead nodes. The size of the dead node list is equal to $\mathbf{N_{dead}}$. There are six steps in this function. The first step is to calculate a new node ID from $\mathbf{myID_{org}}$. The new node ID is an ID in the BMG that excludes all failed nodes. All node IDs after excluding all failed nodes are continuous. The second step is calculating all neighbor IDs of $\mathbf{myID_{org}}$ in BMG size $\mathbf{N}$. The third step is calculating all neighbor IDs of the new node ID (outputs of the first step) in BMG size $\mathbf{n} - \mathbf{n_{dead}}$. The fourth step is to convert outputs of the third step into IDs according to the original BMG.

[5] In case of failure, a broadcast message is encapsulated into a multicast message, and then the message is sent from a parent of the failed node to its children in the binomial spanning tree. The children will de-capsulate the multicast message and continue to forward the initial broadcast message.

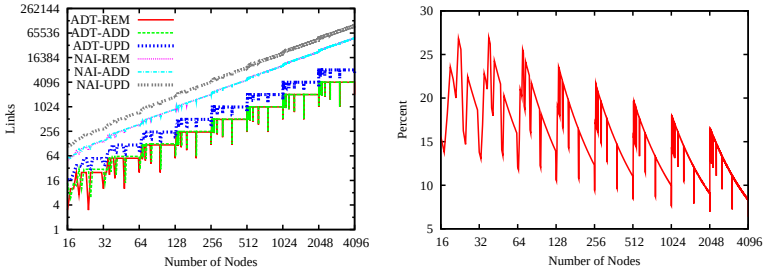


Fig. 5. (a) Number of links used in both methods. (b) % Updated link ratio between adaptive and naive methods.

The fifth step is to calculate the **added** links. All these links are linked to all outputs of the fourth step that do not exist in outputs of the second step. On the other hand, the last step is to calculate **removed** links from outputs of the second step that do not exist in the fourth step. The added and removed links are results of this function.

Algorithm 1 Find added and removed links when some nodes failed.

Require: $\mathbf{N} \in \mathbb{N}, 1 \leq \mathbf{myID_{org}} \leq \mathbf{N}, \{\mathbf{dead} | \forall d \in dead. 1 \leq d \leq \mathbf{N}\}, \mathbf{N_{dead}} = |dead|$

1: $myID_{new} \Leftarrow$ Convert my original IDs to my new IDs.
2: $NeighborID_{org} \Leftarrow$ Get neighbor IDs of $myID_{org}$ in BMG size $\mathbf{N}$.
3: $NeighborID_{new} \Leftarrow$ Get neighbor IDs of $myID_{new}$ in BMG size $\mathbf{N\text{-}N_{dead}}$.
4: Convert all IDs in $NeighborID_{new}$ to IDs in the original BMG.
5: **Added** $\Leftarrow \{\forall a \in Added | a \in NeighborID_{new} \wedge a \notin NeighborID_{org}\}$
6: **Removed** $\Leftarrow \{\forall r \in Removed | r \in NeighborID_{org} \wedge r \notin NeighborID_{new}\}$

Fig. 5(a) illustrates the number of links involved in the self-healing procedures of both methods when there is a failed node. The numbers of added and removed links of each method are roughly the same. The numbers of updated links required by the adaptive method are 10%-30% of the naive method (as shown in Fig. 5(b)), i.e., the adaptive method reduces the overhead from reconstructing the networks up to 90%, especially in a large network.

3.2 Self-Healing Frequency

The frequency of recovering the network may vary from recovering when a node in the BMG failed to recovering when a node had $\delta - 1$ failed neighbors. The frequency is a trade-off between recovery time and an overhead from non-optimal routing. The BMG may be configured such that a node initiates the self-healing procedure when the overhead of routing (in terms of hop number) is more than a threshold. Both the number of failed nodes and the number of hops may also be

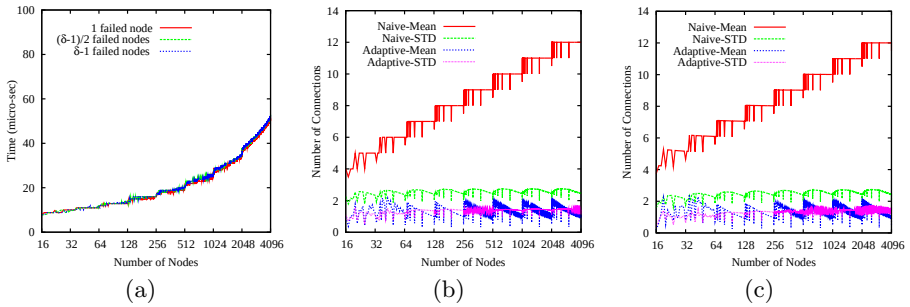


Fig. 6. Experimental Results. (a) Performance of updated link calculation algorithm. (b) Number of added connection per node. (c) Number of removed connection per node.

used as a threshold to start the recovery procedures. No matter which threshold is used, the self-healing procedure must be started before a node had δ failed neighbors. Otherwise the BMG network will become bisectional.

4 Experimental Results

This section presents the evaluation of the algorithm that calculates the added and removed links for the adaptive method (as described in section 3.1). The number of connections that is required by naive and adaptive methods is also evaluated.

The updated-link calculation algorithm has been evaluated with several numbers of failed nodes in BMG sized between 16 and 4096 nodes. The experiments have been conducted on an AMD Athlon^{TM}64 Processor 3500+ 2.2 GHz machine with 1 GB of main memory, running on Linux kernel 2.6.15. Elapsed time of the calling function, that calculates the added and removed links, has been measured. Fig. 6(a) illustrates that the algorithm scales quite well to the number of nodes. The number of failed nodes has marginally affected to the performance of this algorithm.

The number of added and removed connections affects the performance of self-healing procedure. The performance is also dependent on other platform-dependent factors such as time to establish a connection (e.g. three-way handshake in TCP). Fig. 6(b) and Fig. 6(c) present the number of added and removed connections per node in both naive and adaptive methods. The average (mean) number of connections per node of the naive method is significantly higher than that of the adaptive method because the naive method has to update more link than the adaptive method. The standard deviation (STD) of both graphs indicates that the adaptive method has more balance of load than the naive method. Due to the fact that all nodes establish the connection simultaneously, the adaptive method has more parallelism than the naive method.

5 Conclusion and Future Work

This paper presents a self-healing mechanism to improve the fault-tolerant capability of a Binomial graph (BMG) logical topology. A reliability analysis reveals the potential of network bisection, especially when the number of failed nodes is more than 50%. The self-healing mechanism protects BMG from network bisection and helps maintain optimal routing even under failure circumstances. When and how to trigger the self-healing procedures are discussed. The experimental results show that the self-healing with the adaptive method significantly reduces an overhead from re-constructing the networks.

We plan to improve the simulation in near future by using approximation methods (e.g. Monte Carlo) to simulate larger size of BMG. Over the longer term, we hope that BMG will become the basic logical topology of the runtime environments within the FT-MPI and Open MPI libraries.

References

1. Dongarra, J.J., Meuer, H., Strohmaier, E.: TOP500 supercomputer sites. Supercomputer **13** (1997) 89–120
2. Saad, Y., Schultz, M.H.: Topological properties of hypercubes. IEEE Transactions on Computers **37** (1988) 867–872
3. Banerjee, S., Sarkar, D.: Hypercube connected rings: A scalable and fault-tolerant logical topology for optical networks. **24** (2001) 1060–1079
4. Malluhi, Q., Bayoumi, M.: The hierarchical hypercube: A new interconnection topology for massively parallel systems. IEEE Transactions on Parallel and Distributed Systems **05** (1994) 17–30
5. El-Amawy, A., Latifi, S.: Properties and performance of folded hypercubes. IEEE Transactions on Parallel and Distributed Systems **2** (1991) 31–42
6. Kumar, J.M., Patnaik, L.M.: Extended hypercube: A hierarchical interconnection network of hypercubes. IEEE Transactions on Parallel and Distributed Systems **3** (1992) 45–57
7. Tzeng, N.F., Wei, S.: Enhanced hypercubes. IEEE Transactions on Computers **40** (1991) 284–294
8. Preparata, F.P., Vuillemin, J.: The cube-connected cycles: a versatile network for parallel computation. Commun. ACM **24** (1981) 300–309
9. Louri, A., Neocleous, C.: A spanning bus connected hypercube: A new scalable optical interconnection network for multiprocessors and massively parallel systems. IEEE/OSA Journal of Lightwave Technology **15** (1997) 1241–1252
10. Louri, A., Sung, H.: An optical multi-mesh hypercube: A scalable optical interconnection network for massively parallel computing. Journal of Lightware Technology **12** (1994) 704–716
11. Ohring, S., Das, S.K.: Folded petersen cube networks: New competitors for the hypercubes. IEEE Transactions on Parallel and Distributed Systems **7** (1996) 151–168
12. Sivarajan, K.N., Ramaswami, R.: Lightwave networks based on de bruijn graphs. IEEE/ACM Trans. Netw. **2** (1994) 70–79
13. Ganesan, E., Pradhan, D.K.: The hyper-debruijn networks: Scalable versatile architecture. IEEE Transactions on Parallel and Distributed Systems **04** (1993) 962–978

14. Chen, C., Agrawal, D.P., Burke, J.R.: dbcube: A new class of hierarchical multiprocessor interconnection networks with area efficient layout. IEEE Trans. Parallel Distrib. Syst. **4** (1993) 1332–1344
15. Panchapakesan, G., Sengupta, A.: On a lightwave network topology using kautz digraphs. IEEE Transactions on Computers **48** (1999) 1131–1138
16. Karol, M.J.: Optical interconnection using shufflenet multihop networks in multiconnected ring topologies. In: SIGCOMM '88: Symposium proceedings on Communications architectures and protocols, New York, USA, ACM Press (1988) 25–34
17. Maxemchuck, N.F.: Regular mesh topologies in local and metropolitan area networks. AT&T Technical Journal **64** (1985) 1659–1685
18. Campbell, S., Kumar, M., Olariu, S.: The hierarchical cliques interconnection network. Journal of Parallel and Distributed Computing **64** (2004) 16–28
19. Goodman, J.R., Sequin, C.H.: Hypertree: A multiprocessor interconnection topology. IEEE Transactions on Computers **30** (1981) 923–933
20. Angskun, T., Fagg, G.E., Bosilca, G., Pješivac-Grbović, J., Dongarra, J.: Scalable fault tolerant protocol for parallel runtime environments. In: Recent Advances in PVM and MPI. Number 4192 in LNCS, Springer (2006) 141–149
21. Angskun, T., Fagg, G.E., Bosilca, G., Pješivac-Grbović, J., J.Dongarra, J.: Self-healing network for scalable fault tolerant runtime environments. In: Proceedings of 6th Austrian-Hungarian workshop on distributed and parallel systems, Innsbruck, Austria, Springer-Verlag (2006)
22. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content addressable network. Technical Report TR-00-010, Berkeley, CA (2000)
23. Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, H.: Chord: A scalable Peer-To-Peer lookup service for internet applications. In: Proceedings of the 2001 ACM SIGCOMM Conference. (2001) 149–160
24. Harvey, N.J.A., Jones, M.B., a nd Marvin Theimer, S.S., Wolman, A.: Skipnet: A scalable overlay network with practical locality pr operties. In: USENIX Symposium on Internet Technologies and Systems, Seattle, WA, USA, In proceedings of the 4th USENIX Symposium on Internet Technol ogies and Systems (USITS '03) (2003) 113–126
25. Maymounkov, P., Mazieres, D.: Kademlia: A peer-to-peer information system based on the xor metric. In: Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IP TPS), Cambridge, MA, USA (2002)
26. Malkhi, D., Naor, M., Ratajczak, D.R.: Viceroy: A scalable and dynamic emulation of the butterfly. In: Proceedings of the 21st ACM Symposium on Principles of Distributed Comput ing, ACM (2002) 183–192
27. Rowstron, A., Druschel, P.: Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. Lecture Notes in Computer Science **2218** (2001) 329–350
28. Zhao, B.Y., Kubiatowicz, J.D., Joseph, A.D.: Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, UC Berkeley (2001)
29. Angskun, T., Bosilca, G., J.Dongarra, J.: Binomial graph: A scalable and fault-tolerant logical network topology. In: ISPA07. Number 4742 in LNCS, Springer (2007) 471–482
30. Bermond, J.C., Comellas, F., Hsu, D.F.: Distributed loop computer networks: A survey. Journal of Parallel and Distributed Computing **24** (1995) 2–10
31. Lau, F.C.M., Chen, G.: Optimal layouts of midimew networks. IEEE Trans. Parallel Distrib. Syst. **7** (1996) 954–961