Fault Tolerance on Star Graphs

Shuo-Cheng Hu and Chang-Biau Yang
Department of Applied Mathematics
National Sun Yat-sen University
Kaohsiung, Taiwan 804
Republic of China

Abstract

The capability of fault tolerance is one of the advantages of multiprocessor systems. In this paper, we shall prove the fault tolerance of star graphs is $2n-5$ with restriction to the forbidden faulty set. And we shall propose an algorithm for examining the connectivity of star graphs when $2n-4$ faults exist. The algorithm requires $O(n^3 logn)$ time. Besides, we improve the fault-tolerant routing algorithm proposed by Bagherzadeh et al. [5] by calculating the cycle structure of a permutation and the avoidance of routing message to a node without another nonfaulty neighbors. This calculation needs only constant time. And then, we shall propose an efficient fault-tolerant broadcasting algorithm. When no fault occurs, our broadcasting algorithm remains optimal. And the penalty is $O(n^2)$ if at most $n-2$ faults exist.

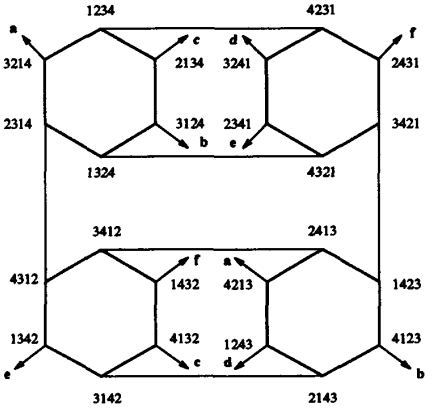


Figure 1: A 4-dimensional star graph, S_4.

1 Introduction

A multiprocessor system consists of a set of processing units and each of them has its own local memory. The processing units in a multiprocessor system are linked in some topology. What we are interested in is a topology proposed by Akers et al. [2, 3], called star graph.

An n-dimensional star graph can be represented by $S_n = (V_n, E_n)$, where V_n consists of $n!$ nodes in which each node is identified by one of the permutations of $\{1, 2, \cdots, n\}$. For any $a, b \in V_n$, $(a, b) \in E_n$ if and only if there exists some i, $2 \leq i \leq n$, such that a and b are identified as $(p_1\ p_2\ \cdots\ p_{i-1}\ p_i\ p_{i+1}\ \cdots\ p_n)$ and $(p_i\ p_2\ \cdots\ p_{i-1}\ p_1\ p_{i+1}\ \cdots\ p_n)$ respectively. We write it as $g_i(a) = b$. The notation S_m^b represents an m-substar in S_n in which the $(m+1)$th position symbol is b. We illustrate S_4 in Figure 1. The star graph is comparable to the hypercube in many aspects. For example, both of them are edge symmetric, node symmetric, strongly hierarchical, bipartite and optimally fault tolerant. But star graphs offer a better degree and diameter than hypercubes. So the star graph structure has been considered as an attractive alternative to the hypercube structure.

The rest of this paper is organized as follows. In Section 2, we shall introduce previous work. In Section 3, we shall propose a generalized measures of fault tolerance. In Section 4, an improved fault-tolerant routing algorithm will be proposed. In Section 5, a new fault-tolerant broadcasting algorithm will be proposed. Finally, this paper will be concluded in Section 6.

0-8186-7038-X/95 $4.00 © 1995 IEEE

2 Previous Work

The optimal broadcasting algorithm on fault-free star graphs has been proposed by Mendia et al. [8]. Their algorithm can be done in the following two phases. First, they route the message to $n-2$ nodes in S_n with the first symbols coinciding with the symbols from second through $(n-1)$th positions of the source node. These nodes are called relay nodes. Second, by applying g_n, each of the $n-2$ nodes will route the message to a unique sub-star. After these two phases, each S_{n-1} of S_n has a copy of the message in one of their nodes. These nodes are called leader nodes which are responsible for broadcasting message in their own sub-stars. The same skill can be recursively applied until the dimension of the sub-stars becomes 1.

The routing algorithm on faulty star graphs is proposed by Bagherzadeh et al. [5], which is based on the greedy routing algorithm combined with depth first search [2,6]. When there are at most $n-2$ faults, the algorithm has penalty $O(\sqrt{n})$.

3 Generalized Measures of Fault Tolerance

The fault tolerance of a star graph is equal to its connectivity minus 1 [1,4]. It is assumed that any subset of nodes in a star graph may be faulty at the same time. We can generalize the measure by restricting some subsets of nodes not to fail at the same time. Similar research on hypercubes has been proposed by Esfahanian [7].

Definition 1 *[7] In a multiprocessor system, a subset of system components is said to be a* **forbidden faulty set** *if the set of nodes does not potentially fail at the same time.*

The connectivity of an S_n is $n-1$ [1,4]. There are $\binom{n!}{n-1}$ node subsets of size $n-1$, only $n!$ of them can disconnect S_n (This will be proved later.). This ratio is very small, and it is getting smaller as n increases. Motived by above, for a node in S_n, we define the forbidden faulty set as the set of the $n-1$ neighbors of that node. For the convenience of representation, we define the following symbols. Let $K'(S_n)$ denote the number of faulty nodes needed to disconnect an S_n with restriction of the forbidden faulty sets. Let $A(G,v)$ denote the set of all nodes adjacent to v in G. And let F^i_{n-1} denote the set of faulty nodes in the S^i_{n-1}.

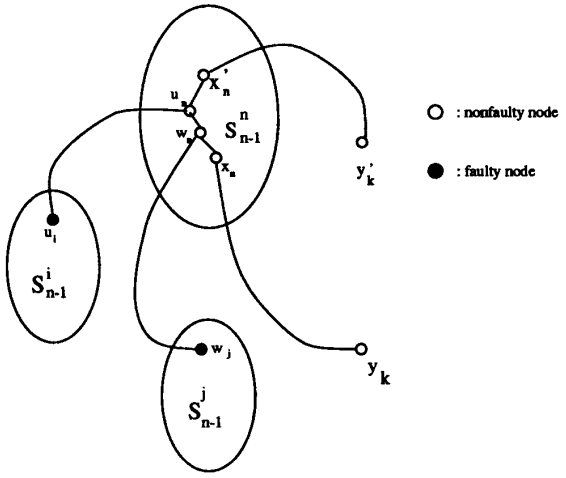


Figure 2: Two kinds of paths connecting u_n to another node y_n in some S_{n-1} through x_n.

Theorem 1 $K'(S_n) = 2n-4$.

Proof: Suppose F is the set of faulty nodes in S_n. If $|F| = 2n-4$. We can derive that if the $2n-4$ nodes adjacent to either node u or v, where u and v are adjacent, then the S_n is disconnected.

Now we want to show that S_n is connected if $|F| \leq 2n-5$. An S_n can be decomposed into n S_{n-1}, say $S^1_{n-1}, S^2_{n-1}, \cdots, S^n_{n-1}$. If there is no such i so that $|F^i_{n-1}| \geq n-2$ ($|F^i_{n-1}| \leq n-3$ for all i), then S_n is connected. So we suppose that there exists i such that $|F^i_{n-1}| \geq n-2$. Without loss of generality, we assume that $i = n$. Now we want to prove that each node in $S^n_{n-1} - F^n_{n-1}$ is connected via a path to a node in $S_n - S^n_{n-1}$.

Let u_n be an arbitrary node in graph $S^n_{n-1} - F^n_{n-1}$ and (u_n, u_i) be an edge from S^n_{n-1} to S^i_{n-1} for some $i \neq n$. If $u_i \notin F$, we have done. So assume that $u_i \in F$. Besides, $A(S_n, u_n)$ is not a subset of F due to the forbidden faulty set. Hence $A(S^n_{n-1} - F^n_{n-1}, u_n)$ is not empty. See Figure 2. Let $w_n \in A(S^n_{n-1} - F^n_{n-1}, u_n)$ and (w_n, w_j) be an edge from S^n_{n-1} to S^j_{n-1} for some $j \neq n$. If $w_j \notin F$, we have done. So assume that $w_j \in F$. Let $X = (A(s^n_{n-1}, u_n) - \{w_n\}) \cup (A(s^n_{n-1}, w_n) - \{u_n\})$. That is, X is the set of neighbors of w_n and u_n in S^n_{n-1} excluding themselves. Since there are $n-2$ neighbors for a node in S^n_{n-1}. So $|X| = 2(n-2) - 2 = 2n-6$. Let $F' = F - \{u_i, w_j\}$. We have $|F'| = 2n-7$. Thus, There must be at least one node in X, say x_n, and (x_n, y_k) is an edge from S^n_{n-1} to S^k_{n-1} for some $k \neq n$, such that both x_n and y_k are nonfaulty. This

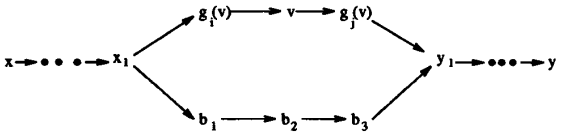


Figure 3: Two paths from x to y.

implies that in $S_n - F$, a node u_n is connected to another node in $S_n - S^n_{n-1}$ via a path of length at most 3. We complete our proof. ∎

In Theorem 1, we have shown that an S_n in which no node has all faulty neighbors can tolerate $2n-5$ faulty nodes. Next, we shall propose an algorithm for determining whether an S_n is connected or not if $2n-4$ faults exist. For identifying the correctness of our algorithm, three lemmas are needed.

Lemma 1 *Let F be the set of faulty nodes in S_n. If $F = A(S_n, v) \cup \{v\}$ for some v in S_n, then graph $S_n - F$ is connected.*

Proof: We shall prove the lemma by showing that for any pair of nonfaulty nodes in S_n , there exists a path connecting them.

Let x and y be two nonfaulty nodes in S_n. We can construct a path from x to y by applying the optimal routing algorithm. If the path does not contain any node in F. We have done. Otherwise, there is at least one fault on the path. It must be one of the following two cases:

$x \rightarrow \cdots \rightarrow x_1 \rightarrow g_i(v) \rightarrow y_1 \rightarrow \cdots \rightarrow y$

and

$x \rightarrow \cdots \rightarrow x_1 \rightarrow g_i(v) \rightarrow v \rightarrow g_j(v) \rightarrow y_1 \rightarrow \cdots \rightarrow y$

where x_1 denotes the last nonfaulty node before the faulty nodes and y_1 denotes the first nonfaulty node after the faulty nodes. In the first case, the path from x_1 to y_1 can be substituted by $x_1 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow y_1$ where x_1, a_1, a_2, a_3, y_1 and $g_i(v)$ form an S_3. And in the second case, the distance between x_1 and y_1 is 4. We can find nodes b_1, b_2 and $b3$ such that the path $x_1 \rightarrow b_1 \rightarrow b_2 \rightarrow b_3 \rightarrow y_1$ can be constructed.(See Figure 3.) We complete our proof. ∎

Lemma 2 *Let F be the set of faulty nodes in S_n. $|F| = n-1$ and $S_n - F$ is disconnected if and only if F is a set of all neighbors of some node v in S_n. $S_n - F$ has exactly two components, one consisting of v, and the other consisting of $S_n - F - \{v\}$.*

Proof: ($\Leftarrow$) It is trivial.
($\Rightarrow$) Suppose there exists u, a neighbor of v, is nonfaulty. It is not hard to see that u connects to all other nodes in $S_n - F$, by an argument similar to the proof of Lemma 1. Thus $S_n - F$ is connected. This is a contradiction. We complete our proof. ∎

Lemma 3 *Let F be the set of faulty nodes in S_n. Suppose that $|F| = 2n-4$ and for every node in S_n, $A(S_n, v)$ is not a subset of F. $S_n - F$ is disconnected if and only if there exists an edge (u, v) in S_n such that $F = (A(S_n, v) - \{u\}) \cup (A(S_n, u) - \{v\})$.*

Proof: ($\Leftarrow$) It is trivial.
($\Rightarrow$) Suppose it is not the case, $F \neq (A(S_n, v) - \{u\}) \cup (A(S_n, u) - \{v\})$. We want to prove that $S_n - F$ is connected. If no S_{n-1} has more than $n-2$ faults, $S_n - F$ is connected. And if only one S_{n-1} has more than $n-2$ faults, applying the same proof technique as that of Theorem 1, we can show that $S_n - F$ is connected. So we suppose that there are two S_{n-1}s having $n-2$ faults. Without loss of generality, we assume the two S_{n-1}s are S^1_{n-1} and S^n_{n-1}. From Lemma 2, the two S_{n-1}s are disconnected if and only if there exist nodes u and v such that the neighbors of both nodes in the same S_{n-1} are faulty. According to our assumption, (u, v) is not an edge in S_n. If both u and v connect to the S_{n-1} without fault, then $S_n - F$ is connected. Otherwise, either u or v connect to each other's S_{n-1} or both. In this case, u or v can start from itself through w (w is a node in $S^n_{n-1} - F - \{v\}$ or $S^1_{n-1} - F - \{u\}$.) via a path to a node in $S_n - S^1_{n-1} - S^n_{n-1}$. So $S_n - F$ is connected. This is a contradiction. We complete our proof. ∎

Now we shall propose an algorithm to determine whether $S_n - F$ is connected when $|F| \leq 2n-4$.

Algorithm 1. Determine if S_n is connected or not when $|F| \leq 2n-4$.

Case 1. $|F| < n-1$. $S_n - F$ is connected.

Case 2. $(n-1) \leq |F| < (2n-4)$. Check if there exists a node $v \in S_n$ such that $v \notin F$ and $A(S_n, v)$ in S_n is a subset of F. If the answer is yes, then $S_n - F$ is disconnected; otherwise $S_n - F$ is connected.

Case 3. $|F| = 2n-4$. Check if there exists an edge $(u, v) \in S_n$ such that $(A(S_n, v) - \{u\}) \cup (A(S_n, u) - \{v\}) = F$. If the answer is yes, then $S_n - F$ is disconnected; otherwise, $S_n - F$ is connected.

The detail of Case 3 in algorithm 1 is as follows.

Algorithm 2. Algorithm of Case 3.

Step 3.1 Select an arbitrary node x from F.

Step 3.2 If F contains a node y whose distance to x is 3, then go to the next step. Otherwise return NO.

Step 3.3 Find one pair of nodes, say u and v, between x and y.

Step 3.4 If $(A(S_n, u) - \{v\}) \cup (A(S_n, v) - \{u\}) = F$ then return YES; otherwise return NO.

Now we can analyze the time complexity of Algorithm 1. In Case 2, there are $(n-1)|F|$ nodes with at least one faulty neighbor. And each of these nodes has $n-2$ neighbors to determine whether they are faulty nodes in F. The label of a node can be viewed as an unsigned real, and F is a sorted list. So it needs $O(((n-1)|F|)(n-2)log\ n)$. It is not difficult to see that Case 3 requires $O(n^2)$ time. We can conclude that the time complexity of Algorithm 1 is $O(((n-1)|F|)(n-2)log\ n) = O(n^3 log\ n)$.

4 An Improved Fault-tolerant Routing Algorithm (IDSR)

Before proposing our algorithm, we define some terminologies and give some assumptions first.

Definition 2 *A node which has a copy of message is said to be* **blocked** *if it cannot forward the message any further, that is, all its neighbor nodes except the one it received message from, is faulty.*

Definition 3 *A node of S_n is* **useless on dimension** *i if the node connected to it by link i is faulty.*

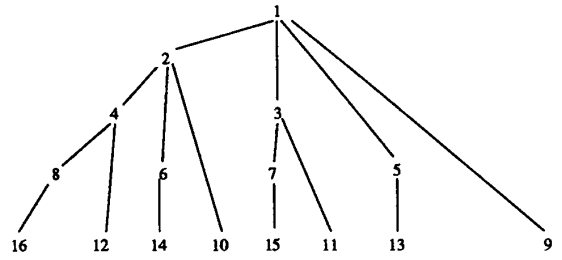
Recall the two phases of the optimal broadcasting algorithm on fault-free star graphs [8]. Any node which receives message from its ith dimension leader will then apply g_i to transmit message to $(i-1)$th dimension leader. If a node is useless on dimension i, then it cannot finish phase 2. Therefore it is useless for us, though it is nonfaulty. Assume that a node has a knowledge of which neighbor node is faulty. And a list of bits, **useless list**, is used to record the status of neighbors of one node.

In our fault-tolerant broadcasting algorithm, there are two basic assumptions: (1) Faults are assumed to be in one or more nodes and with slight modification, we can also take link failures into consideration. (2) A node has only the knowledge of the status of its neighbors. The status includes which one is faulty and which one is nonfaulty. Each node also maintains the useless lists of its neighbors. With the useless list, the blocked neighbors can be found out.

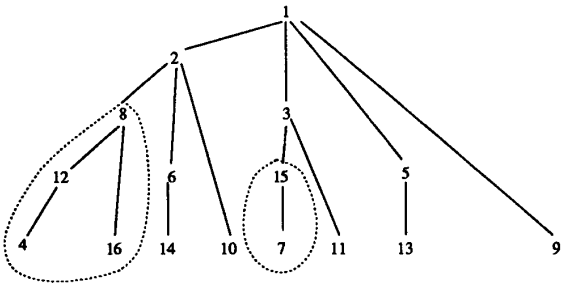
The fault-tolerant routing algorithm proposed by Baghersadeh et al. [5] is based on the depth first search strategy. The major overhead of this kind of algorithm is backtracking. What we focus on is how to decrease backtracking under limited information. This can be done by avoiding routing along blocked nodes. Besides, in their algorithm whenever a node p' of S_n receives a copy of message from another node p, it has to compute the cycle structure of p'. To achieve this purpose, it has to scan all n symbols of p'. In fact, this can be done more efficiently by working on p. In our algorithm, the cycle structure of the next node will be calculated by current node and form a part of the message.

5 A Fault-tolerant Broadcasting Algorithm

In the first phase of the broadcasting algorithm on fault-free star graphs proposed by Mendia et al. [8], they embed the relay nodes on a tree. Applying the rule on an S_{17}, we can derive a broadcasting tree. By preserving the leftmost symbol of each node and omitting the others, the tree is shown in Figure 4(a). These numbers can be viewed as positions. In other words, number i in the tree represents some node whose leftmost symbol is the same as the i-th symbol of the source node. To achieve the purpose of phase one, we have to construct a tree in which all nodes are labeled from 1 to $n-1$ and the root is always 1. We will define a variable l. The l of the source (leader) node will be set to 1. When any other node receives the message, its l will be set to the dimension of the link from which that node received the message. At step i, $1 \le i \le \lceil log(n-1) \rceil$, the nodes storing the message will send the message through the link $g_{l+2^{i-1}}$ if $l + 2^{i-1} \le 2^{\lceil log(n-1) \rceil}$. However the tree is not unique. If any node represented by a number in the original tree is invalid (We will define it in the next paragraph.). It can be substituted if we can construct an isomorphic tree. Therefore, in our mechanism, when

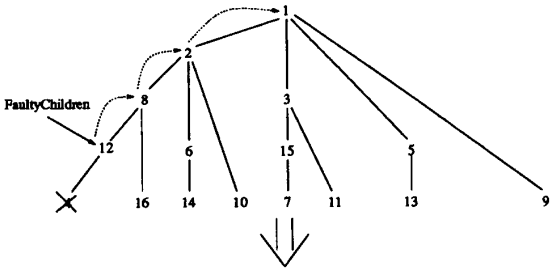


(a)



(b)

Figure 4: Broadcasting trees.(a) A broadcasting tree of S_{17}. (b) A broadcasting tree obtained from (a) after substituting 4 by 8 and 7 by 15.



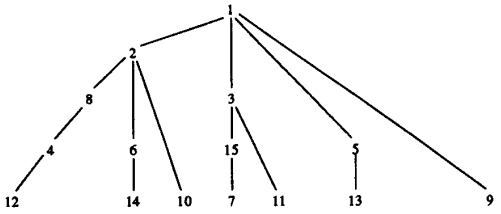


Figure 5: Broadcasting trees of S_{17} and S_{16} respectively.

one or more faults occur, any faulty node can be substituted by any one of its children in the tree. After the substitution, we can do the same calculation but if $l + 2^{i-1} > 2^{\lceil log(n-1) \rceil}$ then the message will be send through $(g_{(l+2^{i-1}) mod (2^{\lceil log(n-1) \rceil})})$ if the neighbor is valid. Otherwise we will do the substitution again. For example, in Figure 4(a), if the node represented by 2 is faulty, any even number smaller than 16 can substitute 2; if the node represented by 3 is faulty, 3 can be substituted by 7,11 and 15, and so on. Figure 4(b) is the tree obtained from Figure 4(a) when nodes represented by 4 and 7 are faulty. Our algorithm can also handle the case when n is not a power of 2.

A node is **valid** if (1) it is nonfaulty, (2) it is not useless on dimension n and (3) at least one of its children in the broadcasting tree are nonfaulty. And a node is **sub-valid** if it only satisfies the first two conditions.

Nevertheless, if the substitution cannot work. For example, if some leaf node is invalid. Since it has no children in the broadcasting tree, there is no substitution can be done. Under the situation, we need to do backtracking. Figure 5 show an example of backtracking: the S_{17} in Figure 5, which is the same as Figure 4(b), if the node represented by 4 is still invalid, the node represented by 12 sends a backtracking message back to its parent node. If the parent node has a valid or subvalid neighbor of link 4, send the broadcasting message to that node. Otherwise, send the backtracking message upward. The same skill can be recursively applied until the root node. Besides, when the node represented by 12 sends a backtracking message, it will let the node represented by 8 know that it has no valid children. Thus, in the next recursion, the node represented by 8 will try to send message along link 4 first as shown the S_{16} in Figure 5.

After the backtracking has been done, the root node will has the collection of all lacking numbers if we still cannot build a complete broadcasting tree. Note that a complete broadcasting tree should have all nodes labeled with 1 through $n-1$. At this moment, the root node has to do routing. Let u be the root node and i be one of the lacking numbers, u will route message to $v(= g_n g_i(u))$ which is the closest node to u in the S^i_{n-1}. The routing to several destinations can be done in pipeline. Since u has the knowledge of useless list of $g_i(u)$. If v is faulty, we will select a neighbor of v, says v', which is in the same smallest sub-star with v. The reason why we select such a v' is, in this way, we can localize the influence of v. Note that v' becomes the leader node if it is nonfaulty, and v' and v are in the same smallest sub-star. If the selected node cannot be reached, u will route message to

$g_{n-1}g_2(v'), g_{n-1}g_3(v'), \cdots, g_{n-1}g_{n-2}(v'), g_{n-1}(v')$ and some neighbor node of v' in the same S_{n-2} in the next recursion. That is, u will take the responsibility of v' in S^i_{n-1}, route message to exactly one node in each S_{n-2} of S^i_{n-1}. Our algorithm can be divided into two phases as follows.

Algorithm FTB (Fault-tolerant Broadcasting Algorithm)

Phase 1 :

1.1 If the source (leader) node has only one nonfaulty neighbor then select the nonfaulty neighbor as a new source (leader) node.

1.2 Construct the broadcasting tree and if invalid nodes are encountered, try to do a substitution if possible.

1.3 If there still is one or more nodes cannot be reached, keep track of the lacking numbers. Note that a complete broadcasting tree should have all nodes labeled with 1 through $n-1$.

Phase 2 :

2.1 For any node which has recorded the lacking numbers, send the numbers and the *FaultyChildren* bit, which is set to 1 if all children of the node are invalid, to its parent. And when a node receives such a message, if there are some neighbor nodes whose leftmost symbol are the same as some lacking numbers, forward the message to those nodes and delete those lacking numbers.

2.2 Each relay node forwards the message to a unique sub-star.

2.3 For every lacking number i, the source (leader) node route the message to a node with the n-th symbol i using IDSR. If the selected node d cannot be reached, the source (leader) node will route message to nodes $g_{n-1}g_2(d), g_{n-1}g_3(d), \cdots, g_{n-1}g_{n-2}(d), g_{n-1}(d)$ and some neighbor node of d in the same S_{n-2} in the next recursion.

If there is no faulty node in the star graph, algorithm FTB is optimal. Since under this condition, FTB is the same as the optimal broadcasting algorithm on fault-free star graphs. Therefore, FTB is optimal.

Theorem 2 *In an S_n, if there is only one fault, FTB has penalty at most $2n+2$.*

Proof: First we consider the case: $n = 2^m + 1$ for some $m \in N$

Recall the tree constructed in phase 1 of FTB. The more children the faulty node has , the less penalty will be caused by it. Note that the only node which cannot be substituted and cannot do backtracking is the one with leftmost symbol $\frac{n+1}{2}$, denoted as v.

If v is the faulty node, $g_n(v)$ and itself cannot be a leader in smaller sub-stars any more. Then from dimension n to $\frac{n+1}{2}$, each has one node needed to route. The distance from the source to those nodes is 2. Routing to these nodes needs 4 hops. So the total penalty is

$$4(\tfrac{n+1}{2}) = 2n+2$$

For the case when n is between 2^m+2 and $2^{m+1}+1$ for some $m \in N$, the penalty is no more than $2n+2$. The proof is similar. ∎

Theorem 3 *In an S_n, if the number of faults is at most $n-2$, FTB has penalty $O(n^2)$.*

Proof: Let r be the total number of nodes to which we should route data.Let d be the maximum distance from the source (leader) node to the destination for routing and p be the maximum number of penalty hops for routing one data element. Since the routing is performed in pipeline, total routing time needed in Step 2.3 is $O(r + n(d+p))$. And the routing is an extra work to broadcasting. Thus, the total penalty of FTB is $O(r + n(d+p))$.

Our IDSR has penalty at most $O(\sqrt{n})$, so $p = O(\sqrt{n})$. The distance of nodes in S_n is $O(n)$. Thus $d = O(n)$. Since there are at most $n-2$ faults, we want to find out the upper bound of r. Let u denote the source (leader) node. Suppose i is one of the lacking number such that $g_i(u)$ and $g_n g_i(u)$ are faulty. In the nth recursion, u originally has to route data to $g_n g_i(u)$. However, u has to route data to $g_2 g_n g_i(u)$ since $g_n g_i(u)$ is faulty. Suppose $g_2 g_n g_i(u)$ is also faulty. In the $(n-1)$th recursion, u has to broadcast the message to $n-1$ leader nodes by our IDSR. With this argument, u has to do extra routing to $O(n)$ nodes when a faulty node exists. It follows that $r = O(n^2)$ since there are at most $n-2$ faults. Thus, the total penalty is $O(n^2)$. ∎

We show an example in Figure 6, in which the source node is 12345. In the first phase of the 5th

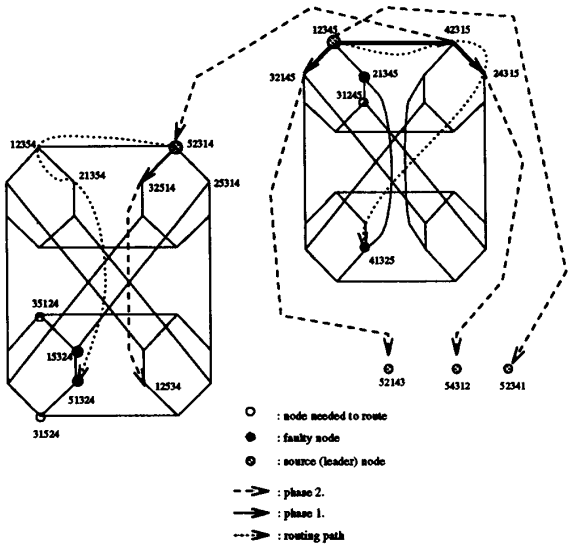


Figure 6: An example of FTB on S_5.

recursion, since 21345 is faulty, 42315 takes its place. In the second phase, applying g_5, each S_4 has a unique leader node. We omit the further progress in S_4 without fault. In the 4th recursion, 12345 is the leader node of S_4^5. Since 21345 is faulty and no substitution can be found, 12345 needs to route message to $g_4g_2(12345) = 41325$. Since the leader node of S_4^4 is 52314, and 25314 is useless on dimension 4, 52314 should route data to $g_2g_4g_2(52314) = 51324$. In 3rd recursion, since 51324 is faulty, 52314 should route to $\{g_3g_2(51324), g_3(51324)\} = \{35124, 31524\}$. Again, since 21345 is faulty, 12345 should route to $g_3(21345) = 31245$.

6 Concluding Remarks

In this paper, we have introduced a generalized fault tolerance measure, the forbidden faulty set, for star graphs. We proved that the fault tolerance of S_n is $2n-5$ with restriction to the forbidden faulty sets. We also proposed an algorithm for determining whether an S_n is connected if a set of $2n-4$ faults is given. The algorithm requires $O(n^3 log\ n)$ time.

Data routing and broadcasting are two basic and important issues in multiprocessor systems. We improved the fault-tolerant routing algorithm proposed by Bagherzadeh et al. [5] to make it more efficient. We also proposed a mechanism to generate a fault-tolerant broadcasting tree when faults are encountered. In this way, we can localize the influence of faulty nodes. Based on that, we proposed an efficient broadcasting algorithm on faulty star graphs. Our algorithm remains optimal when no fault occurs. And the total penalty is $O(n^2)$ if at most $n-2$ faults exist.

Although the star graph structure has been proposed for a couple of years. There are still a lot of open problems in this field. We would be glad to see that more and more researchers pay attention to the nice properties of star graph structure and work on it.

References

[1] S. B. Akers, D. Harel, and B. Krishnamurthy, "On group graphs and their fault tolerance," *IEEE Trans. Computers*, Vol. C-36, pp. 885–888, July 1987.

[2] S. B. Akers, D. Harel, and B. Krishnamurthy, "The star graph: An attractive alternative to the n-cube," *Proc. of the Int'l Conference on Parallel Processing*, pp. 393–400, 1987.

[3] S. B. Akers and B. Krishnamurthy, "A group-theoretic model for symmetric interconnection networks," *IEEE Trans. Computers*, Vol. 38, pp. 555–566, Apr. 1989.

[4] B. Alspach, "Cayley graphs with optimal fault tolerance," *IEEE Trans. Computers*, Vol. 41, No. 10, pp. 1337–1339, Oct. 1992.

[5] N. Bagherzadeh, N. Nassif, and S. Latifi, "A routing and broadcasting scheme on faulty star graphs," *IEEE Trans. Computers*, Vol. 42, No. 11, pp. 1398–1403, Nov. 1993.

[6] D. M. Blough and N. Bagherzadeh, "Near-optimal message routing and broadcasting in faulty hypercubes," *Int. J. Parallel Programming*, Vol. 19, No. 5, pp. 405–423, 1990.

[7] A. H. Esfahanian, "Generalized measures of fault tolerance with application to n-cube networks," *IEEE Trans. Computers*, Vol. 38, No. 11, pp. 1586–1591, Nov. 1989.

[8] V. E. Mendia and D. Sarkar, "Optimal broadcasting on the star graph," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 3, No. 4, pp. 389–396, Jul. 1992.