アルゴリズム 55－2
(1997. 1. 23)

An Adaptive Distributed Fault-Tolerant Routing Algorithm for the Star Graph

Leqiang Bai[1] Hiroyuki Ebara[2] Hideo Nakano[3] Hajime Maeda[1]

[1]Department of Communication, Faculty of Engineering, Osaka University
[2]Faculty of Engineering, Kansai University
[3]Osaka City University

This paper presents an adaptive distributed fault-tolerant routing algorithm for the n-star graph. Based on the local failure information and the properties of the star graph, the algorithm can make routing decisions without deadlock and livelock. After faults are encountered, the algorithm routes messages to a given destination by finding a fault-free $n-1$-star graph. As long as the number f of faults (node faults and/or edge faults) is less than the degree $n-1$ of the n-star graph, the algorithm can adaptively find a path of length at most $d+6f$ to route messages from a source to a destination, where d is the distance between source and destination.

スターグラフにおける耐故障適応ルーディングアルゴリズム

白 楽強[1]　榎原 博之[2]　中野 秀男[3]　前田 肇[1]

[1] 大阪大学工学部通信工学科
[2] 関西大学工学部
[3] 大阪市立大学

この論文では、$n-$ スターグラフに対する耐故障適応ルーディングアルゴリズムを提案する。このアルゴリズムは局所的な故障情報とスターグラフの構造により、デッドロックとライブロックの無いルーディングを見つける。故障に出会ったとき、故障の無い $n-1-$ スターグラフを見つけながら、メッセージを目的頂点に送る。故障の数 f （頂点故障 and/or 辺故障）が $n-$ スターグラフの次数より小さければ、高々 $d+6f$ の長さのルーディングを見つけることができる。ここで、d はソースから目的頂点までの距離である。

1 Introduction

With the advent of massively parallel computers, it has become highly desirable to construct the interconnection network that has as many nodes as possible within a given degree and diameter. The hypercube has been drawn considerable attention from both academic and industrial communities. The star graph in [1] claims to possess topological superiority over the hypercube. Similar to the hypercube, the star graph possesses rich recursive structure, symmetrical properties and simple routing on the fault-free star graph. In addition, it has a smaller diameter and degree, and a lower average diameter for a given size than the hypercube.

Fault-tolerant routing problems have been studied for different interconnection networks. The problems with different fault models have been studied for the hypercube in [5],[9],[13],[14]. Fault tolerance of the star graph was discussed in [1],[7],[12]. Given a set of at most $n-2$ faulty nodes, node-to-node and set-to-set fault tolerant routing algorithms for the star graph have been presented in [5],[6]. Only based on local failure information, fault-tolerant routing algorithms for the star graph have been described subject to faulty edges in [3], [4]. The shortcoming of the algorithms listed above for the star graph is that all these algorithms are only subject to faulty nodes or faulty edges. When the types of faults are unknown, these algorithms cannot insure routing success. Therefore, it is necessary to develop routing algorithms which can tolerate node faults and/or edge faults for the star graph.

In this paper, we present an adaptive distributed fault-tolerant routing algorithm without any deadlock and livelock for the star graph. A special property of the star graph, which is not possessed by the hypercube, is that the number of copies of $n-1$-star graph that makes up the n-star graph is larger than $n-1$ that is the fault tolerance of the n-star graph. We use this property together with the local failure information to develop a fault-tolerant routing algorithm that can route messages to a given destination by finding a fault-free $n-1$-star graph. It is not necessary to judge types of faults when faults are encountered. The algorithm can tolerate at most $n-2$ faults (node faults and/or edge faults) to route messages successfully for an n-star graph.

2 Preliminaries

Let V denote the set of $n!$ permutation of symbols $\{1, 2, ..., n\}$. An n-star graph interconnection network on n symbols, denoted by $S_n = (V, E)$, is an undirected graph with $n!$ nodes. The nodes of S_n are in a 1-1 correspondence with the permutation $p = p_1p_2...p_n$ of $< n >= \{1, 2, ..., n\}$. Two nodes of S_n are connected by an edge if and only if the permutation of one node can be obtained from the other by interchanging the first symbol p_1 with the ith symbol p_i, $2 \leq i \leq n$. Obviously, every node has $n-1$ incident edges, corresponding with $n-1$ symbols which the symbol in the first position can be interchanged with. Thus, S_n is a regular graph of degree $n-1$ and is $(n-1)$-connected. S_n possesses a number of properties that are desired by interconnection networks. These include node and edge symmetric, maximal fault tolerance, and strong resilience. S_n has a high recursive structure, and is made up of n copies of $n-1$-star graph.

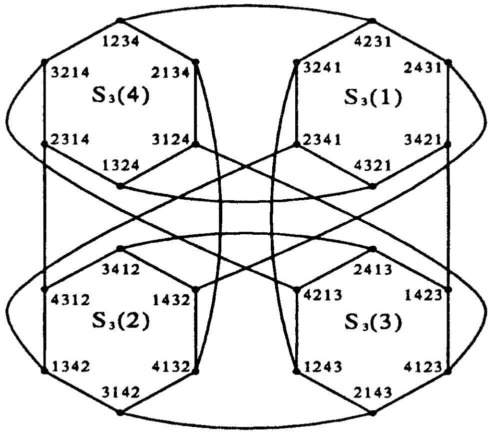

Fig.1: The 4-star graph viewed as four 3-star graphs.

Definition 1 *Given a node t in S_n, $S_{n-1}(t_i)$ denotes an $n-1$-star graph induced by all the nodes with the same last symbol t_i, $t_i \in < n >$, and $S_n = \{S_{n-1}(t_i) | 1 \leq i \leq n\}$.*

Definition 2 *For a node x in S_n, $x^{(i)}$ denotes the node that is obtained by interchanging the first symbol with the ith symbol, specially $x^{(1)} = x$. Given a node t in S_n, $x^{(t_j)}$ for $1 \leq j \leq n$ denotes the node that is obtained by interchanging the first symbol with the symbol t_j, specially $x^{(x_1)} = x$. $x^{(t_j,i)}$ denotes the node that is obtained by interchanging the first symbol with the ith symbol of $x^{(t_j)}$, and $x^{(i,t_j)}$ denotes the node that is obtained by interchanging the first symbol with the symbol t_j of $x^{(i)}$.*

Example 1: Let t = 12345 and x = 54231. Then, $x^{(3)} = \underline{2}4\underline{5}31$, $x^{(t_5)} = \underline{3}42\underline{5}1$, $x^{(t_5,3)} = \underline{2}4\underline{35}1$ and $x^{(3,t_5)} = \underline{3}4\underline{52}1$.

Now, we show some topological properties of S_n that are important in this paper. Let $d(x, y)$ denote the distance of two nodes x and y. Let (t_it_j) be an operator such that $x(t_it_j) = x(t_jt_i) = x_1...t_j...t_i...x_n$ for $x = x_1...t_i...t_j...x_n$, where $x(t_it_j) = x$ if $i = j$. We use the operator (t_it_j) to define a function that can show an image relation from a node in $S_{n-1}(t_i)$ to a corresponding node in $S_{n-1}(t_j)$.

Definition 3 *$\varphi_{ij} : x \rightarrow \varphi_{ij}(x)$, $1 \leq i \leq n$ and $1 \leq j \leq n$, is an image function, and $\varphi_{ij}(x)$ is called the image node of x. For a given node t, $\varphi_{ij}(x)$ is given by:*

$$\varphi_{ij}(x) = \begin{cases} \varphi_{ii}(x) = x & if \quad i = j \\ \varphi_{i1}(x) = x(t_1t_i) & if \quad j = 1, 2 \leq i \leq n \\ \varphi_{1j}(x) = x(t_1t_j) & if \quad i = 1, 2 \leq j \leq n \\ \varphi_{ij}(x) = x(t_it_j)(t_1t_i) & if \quad 2 \leq j \neq i \leq n \end{cases}.$$

Lemma 1 *Let x be in $S_{n-1}(t_i)$, then $\varphi_{ij}(x)$ is in $S_{n-1}(t_j)$ and $d(x, t^{(t_i,n)}) = d(\varphi_{ij}(x), t^{(t_j,n)})$.*

Proof. Let $x = x_1...t_i$, since

$$\varphi_{ij}(x) = \begin{cases} x_1...t_i = x_1...t_j & if \quad i = j \\ x_1...t_1...t_i(t_1t_i) = x_1...t_i...t_1 & if \quad j = 1, 2 \leq i \leq n \\ x_1...t_j...t_1(t_1t_j) = x_1...t_1...t_j & if \quad i = 1, 2 \leq j \leq n \\ x_1...t_1...t_j...t_i(t_it_j)(t_it_1) = x_1...t_i...t_1...t_j & if \quad 2 \leq j \neq i \leq n \end{cases},$$

φ_{ij} is in $S_{n-1}(t_j)$. When $x = t^{(t_i,n)}$,

$$\varphi_{ij}(t^{(t_i,n)}) = \begin{cases} t^{(t_i,n)} = t^{(t_j,n)} & if \quad i = j \\ t_n...t_1...t_i(t_1t_i) = t^{(t_1,n)} & if \quad j = 1, 2 \leq i \leq n \\ t_n...t_j...t_1(t_1t_j) = t^{(t_j,n)} & if \quad i = 1, 2 \leq j \leq n \\ t_n...t_1...t_j...t_i(t_it_j)(t_it_1) = t^{(t_j,n)} & if \quad 2 \leq j \neq i \leq n \end{cases}.$$

Based on node and edge symmetric of the star graph, $d(x,t^{(t_i,n)}) = d(\varphi_{ij}(x),t^{(t_j,n)})$. □

Example 2. As shown in Fig.1, let $t = 1234$. 2.1: Let $x = 1324$, then $\varphi_{41}(1324) = 1324(41) = 4321$. $d(4321,t^{(t_1,4)} = 4231) = d(1324,t = 1234) = 3$. 2.2: Let $x = 3412$, then $\varphi_{23}(3412) = 3412(23)(21) = 1423$. $d(1423,t^{(t_3,4)} = 4213) = d(3412,t^{(t_2,4)} = 4132) = 2$.

Lemma 2 *Given a node t, let $x_n = t_i$ and $y = \varphi_{ij}(x)$, then $d(x,y)) \leq 2$ and $x^{(n)}$ is in the shortest path from x to y when $x_1 = t_j$, and $3 \leq d(x,y) \leq 4$ and $x^{(t_j,n)}$ is in the shortest path from x to y when $x_1 \neq t_j$.*

Proof. It is clear that $d(x,y) = 0$ when $i = j$. When $x_1 = t_j$, $x^{(n)}$ is in the shortest path from x to y if $d(x^{(n)},y) = d(x,y)-1$. When $x_1 \neq t_j$, $x^{(t_j,n)}$ is in the shortest path from x to y if $d(x^{(t_j,n)},y) = d(x,y)-2$. We prove Lemma 2 for $i \neq j$ in three cases.

Case 1. $x = x(t_n) = x_1x_2...t_n$ is in $S_{n-1}(t_n)$:

1.1. $y = \varphi_{n1}(x) = x(t_1t_n)$: When $x_1 = t_1$, $x = t_1...t_n$ and $y = x(t_1t_n) = t_n...t_1$. Then, $d(x^{(t_1,n)},y) = d(x^{(n)},y) = 0$ and $d(x,y) = 1$. When $x_1 \neq t_1$, $x = x_1...t_1...t_n$. $y = x(t_1t_n) = x_1...t_n...t_1$. Then, $d(x^{(t_1,n)},y) = 1$ and $d(x,y) = 3$.

1.2. $y = \varphi_{nj}(x) = x(t_jt_n)(t_1t_n)$, $2 \leq j \leq n-1$: When $x_1 = t_j$, $x = t_j..t_1...t_n$ and $y = x(t_jt_n)(t_1t_n) = t_1...t_n...t_j$. Then, $d(x^{(t_j,n)},y) = d(x^{(n)},y) = 1$ and $d(x,y) = 2$. When $x_1 \neq t_j$ and $i < j$, $x = x_1...t_1..t_j...t_n$ and $y = x(t_jt_n)(t_1t_n) = x_1...t_n...t_1...t_j$. Then, $d(x^{(t_j,n)},y) = 2$ and $d(x,y) = 4$. Similarly, $d(x^{(t_j,n)},y) = 2$ and $d(x,y) = 4$ when $i > j$.

Case 2. $x = x(t_1) = x_1x_2...t_1$ is in $S_{n-1}(t_1)$ and $y = \varphi_{1j}(x) = x(t_1t_j)$, $2 \leq j \leq n$:

When $x_1 = t_j$, $x = t_j...t_1$ and $y = x(t_1t_j) = t_1...t_j$. Then, $d(x^{(t_j,n)},y) = d(x^{(n)},y) = 1$ and $d(x,y) = 1$. When $x_1 \neq t_j$, $x = x_1...t_j...t_1$ and $y = x(t_1t_j) = x_1...t_1...t_j$. Then, $d(x^{(t_j,n)},y) = 1$ and $d(x,y) = 3$.

Case 3. $x = x(t_i) = x_1x_2...t_i$ is in $S_{n-1}(t_i)$, $2 \leq i \leq n$:

3.1. $y = \varphi_{i1}(x) = x(t_1t_i)$, $2 \leq i \leq n$: When $x_1 = t_1$, $x = t_1...t_i$ and $y = x(t_1t_i) = t_i...t_1$. Then, $d(x^{(t_1,n)},y) = d(x^{(n)},y) = 0$ and $d(x,y) = 1$. When $x_1 \neq t_1$, $x = x_1...t_1...t_i$ and $y = x(t_1t_i) = x_1...t_i...t_1$. Then, $d(x^{(t_1,n)},y) = 1$ and $d(x,y) = 3$.

3.1. $y = \varphi_{ij}(x) = x(t_it_j)(t_it_1)$, $2 \leq i \neq j \leq n$: When $x_1 = t_j$, $x = t_j...t_1...t_i$. $y = x(t_it_j)(t_it_1) = t_1...t_i...t_j$. $d(x^{(t_j,n)},y) = d(x^{(n)},y) = 1$ and $d(x,y) = 2$. When $x_1 \neq t_j$, $x = x_1...t_j...t_1...t_i$. $y = x(t_it_j)(t_it_1) = x_1...t_1...t_i...t_j$. $d(x^{(t_j,n)},y) = 2$ and $d(x,y) = 4$. □

Lemma 3 *There is at least one shortest path from x to t that passes through the nodes only in $S_{n-1}(x_n)$ and $S_{n-1}(t_n)$.*

Lemma 4 *Let x and t be two nodes in S_n, then $d(x,t^{(x_n,n)}) \leq d(x,t)$.*

Proof. Without loss of generality, assume $t = 12...n$. If $x_n = n$, $d(x,t^{(x_n,n)}) = d(x,t)$ since $t^{(x_n,n)} = t$. If $x_n = i \neq n$, x is in $S_{n-1}(i)$, From Lemma 3, there is one shortest path through the nodes only in $S_{n-1}(i)$ and $S_{n-1}(n)$. As shown in Fig.2, let $y(n) = x(i)^{(n)}$ and $y(i) = \varphi_{ni}(y(n))$. According to Lemma 2, $x(i) = y(n)^{(n)}$ is in the shortest path from $y(n)$ to $y(i)$. Based on Lemma 1, $d(y(i),t^{(i,n)}) = d(y(n),t)$. Since $y(n)_1 = y(i)_n$, $d(y(i),y(n)) \leq 2$ according to Lemma 2. From $d(x(i),y(i)) = d(y(i),y(n)) - d(x(i),y(n))$, $d(x(i),y(i)) \leq d(x(i),y(n))$. From $d(x,t^{(i,n)}) \leq d(x,x(i))+d(x(i),y(i))+d(y(i),t^{(i,n)})$, then $d(x,t^{(i,n)}) \leq d(x,x(i)) + d(x(i),y(n)) + d(y(i),t^{(i,n)}) = d(x,t)$. □

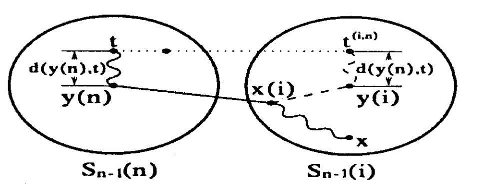

Fig.2: The shortest path through the nodes in $S_{n-1}(i)$ and $S_{n-1}(n)$.

3 Fault-Tolerant Routing Algorithm for S_n

In this section, we not only present an adaptive distributed fault-tolerant routing algorithm, but prove its correctness, and analyze its properties. We make the following assumptions:

A fault can be a node fault and/or an edge fault. If a node is faulty, all the edges incident to it are treated as faulty edges. Each edge is bidirectional. If an edge is faulty, both directions are faulty. The total number of faults is less than degree $n-1$ of S_n. Any node only knows the condition of its incident edges. The source and the destination are fault-free.

3.1 Routing Algorithm for S_n

Since the total number of faults is less than degree $n-1$ of S_n, there are at least two fault-free $n-1$-star graphs in $\{S_{n-1}(t_i)|1 \le i \le n\}$. Given a destination node t, there is at least a fault-free adjacent node of t, which is in or is adjacent to a fault-free $S_{n-1}(t_i)$, $1 \le i \le n$. Let F denote the set of the invalid nodes that is treated as the faulty nodes. When a node x encounters a faulty edge, F is updated as follows.

1. For $2 \le i \le n$, $F := F \cup x^{(x_i)}$ if $x^{(x_i)} \ne t$ and $(x, x^{(x_i)})$ is faulty.
2. For $2 \le i \le n$, $F := F \cup x$ if $x^{(x_i)} = t$ and $(x, x^{(x_i)})$ is faulty.

Let S denote the set of invalid $n-1$-star graphs that is treated as the faulty subgraphs. It is updated as follows.

1. For any $x \in F$, $S := S \cup S(x_n)$ if $x \notin \{t^{(t_i)}|2 \le i \le n-1\}$.
2. For any $x \in F$, $S := S \cup S(x_n) \cup S(x_1)$ if $x \in \{t^{(t_i)}|2 \le i \le n-1\}$.

Let *FT-procedure* be a function for performing the fault-tolerant procedure. The node x in the course of *FT-procedure* updates F and S based on local failure information, and decides a node z with $z \notin F$ that is in or is incident to some $S_{n-1} \notin S$, or returns the node $x^{(n)}$ if the edge (x, t) is faulty and each valid adjacent node of x with $x \in \{t^{(t_i)}|2 \le i \le n-1\}$ is not directly connected to $S_{n-1} \notin S$.

function *FT-procedure* (x, t: node; **var** F: the set of invalid nodes;
                                        var S: the set of invalid $n-1$-star graphs): node;
var i, n : *integer*; y: node; /* t_i, x_i, y_i: symbol of t, x, y */
begin
  /* x is the node that is ready to send M, t is the destination node */
  for $i := 2$ **to** n **do begin** /* update F */
    if $x^{(x_i)} = t$ and $(x, x^{(x_i)})$ is faulty **then** $F := F \cup x$;
    if $x^{(x_i)} \ne t$ and $(x, x^{(x_i)})$ is faulty **then** $F := F \cup x^{(x_i)}$
  end;
  for any $y \in F$ **do begin** /* update S */
    if $y \notin \{t^{(t_i)}|2 \le i \le n-1\}$ **then** $S := S \cup S_{n-1}(y_n)$;
    if $y \in \{t^{(t_i)}|2 \le i \le n-1\}$ **then** $S := S \cup S(y_n) \cup S(y_1)$
  end;
  /* to decide $z \notin F$ that is in or is incident to a $S_{n-1}(t_i) \notin S$ */
  if $S_{n-1}(x_n) \notin S$ **then return** (x);
  if $S_{n-1}(x_1) \notin S$ **then return** $(x^{(x_n)})$;
  for $i := 2$ **to** $n-1$ **do begin**
    if $S_{n-1}(x_i) \notin S$ and $(x, x^{(x_i)})$ is non-faulty **then return** $(x^{(x_i)})$
  end;
  return$(x^{(n)})$
end;

Following two rules described in [1] that insure a path of minimum distance between two nodes, it is easy to develop an optimal routing algorithm for a fault-free S_n. Let $ROUTING(x, t)$ be a routing function that decides the adjacent node of x, which is in the shortest path from x to t for the fault-free S_n, and let $ROUTING(x, t^{(x_i,n)})$ be a routing function that decides the adjacent node of x, which is in the shortest path from x to $t^{(x_i,n)}$ for the fault-free $S_{n-1}(x_i)$, $1 \le i \le n$. Based on *FT-procedure*, $ROUTING(x, t)$ and $ROUTING(x, t^{(x_i,n)})$, we present a routing algorithm called **FT_ROUTING** that can tolerate at most $n-2$ faults. Let $M = \{messages, t, F, S\}$ denote a sending request. By **FT_ROUTING**, a node x that possesses M determines a proper node z for routing the messages based on the following routing rules:

Rule 1. x is an adjacent node of t: If the edge (x, t) is non-faulty, M is sent to t. Otherwise, x sends M to the node decided by *FT-procedure*.

Rule 2. $x \in \{t^{(t_i,n)}|2 \le i \le n-1\}$: If $x^{(n)} \notin F$ and the edge $(x, x^{(n)})$ is non-faulty, M is sent to $x^{(n)}$. Otherwise, x sends M to the node decided by *FT-procedure*.

Rule 3. x is not an adjacent node of t and $x \notin \{t^{(t_i,n)}|2 \le i \le n-1\}$:

3.1. $S = \emptyset$: If the edge (x, z) is non-faulty, where z is decided by $ROUTING(x, t)$, M is sent to z. Otherwise, x sends M to the node decided by *FT-procedure* if x itself is not the node decided by *FT-procedure*. If x is the node decided by *FT-procedure*, go to **3.2.1**.

3.2. $S \neq \emptyset$:

3.2.1. $S_{n-1}(x_n) \notin S$: If the edge (x, z) is non-faulty, where z is the node decided by $ROUTING(x, t^{(x_i,n)})$, M are sent to z. Otherwise, the edge (x, z) is faulty, x sends M to the node decided by *FT-procedure*.

3.2.2. If $S_{n-1}(x_n) \in S$, x sends M to the node decided by *FT-procedure*.

```
Algorithm FT_ROUTING;
Input: A sending request $M$.
Output: A node $z$ to which a sending request $M$ is sent.
var $i, n$ : integer; $x, t, y, z$: node; /* $t_i, x_i$: symbol of $t$, $x$ /*
var $F$: the set of invalid nodes; $S$: the set of invalid $n-1$-star graphs;
begin
  /* $x$ is the node that is ready to send $M$, $z$ is the node to which $M$ is sent */
  if $x \in \{t^{(t_i)} | 2 \leq i \leq n\}$ then begin /* Rule 1 */
    for $i := 2$ to $n$ do begin
      if $(x, t)$ is non-faulty then begin $z := t$; goto SENDING end
      else begin $z :=$ FT-procedure$(x, t, F, S)$; goto SENDING end
    end
  end; /* Rule 1 */
  if $x \in \{t^{(t_i,n)} | 2 \leq i \leq n-1\}$ then begin /* Rule 2 */
    for $i := 2$ to $n-1$ do begin
      if $t^{(t_i)} \notin F$ and $(x, t^{(t_i)})$ is non-faulty then begin $z := t^{(t_i)}$; goto SENDING end
      else begin $z :=$ FT-procedure$(x, t, F, S)$; goto SENDING end
    end
  end; /* Rule 2 */
  /* begin Rule 3 */
  if $S = \emptyset$ then begin $y := ROUTING(x, t)$; /* Rule 3.1 */
    if $(x, y)$ is non-faulty then begin $z := y$; goto SENDING end
    else begin $z :=$ FT-procedure$(x, t, F, S)$;
      if $x \neq z$ then goto SENDING /* if $x = z$, Rule 3.2.1 */
    end
  end; /* Rule 3.1 */
  if $S \neq \emptyset$ then begin /* Rule 3.2 */
    if $S_{n-1}(x_n) \notin S$ then begin $y := ROUTING(x, t^{(x_i,n)})$; /* Rule 3.2.1 */
      if $(x, y)$ is non-faulty then begin $z := y$; goto SENDING end
      else begin $z :=$ FT-procedure$(x, t, F, S)$; goto SENDING end
    end; /* Rule 3.2.1 */
    if $S_{n-1}(x_n) \in S$ then $z :=$ FT-procedure$(x, t, F, S)$ /* Rule 3.2.2 */
  end /* Rule 3.2 */
  /* end Rule 3 */
  SENDING: if $x \neq t$ then send $M$ to $z$ /* if $x$ is not $t$, route $M$ from $x$ to $t$ */
end;
```

3.2 Performance Analysis of FT_ROUTING

Given a destination t, each of its adjacent nodes can solely determine a $S_{n-1}(t_i)$ of S_n, and the number of $S_{n-1}(t_i)$, $1 \leq i \leq n-1$, is equal to $n-1$ that is the fault tolerance of S_n. This special property of the star graph is the basis for us to develop **FT_ROUTING**. By updating F and S dynamically, the algorithm needs not to judge that faults are node faults or edge faults and can get necessary information for finding a $S_{n-1} \notin S$. At a time, each node in routing sends messages to one node only. Messages are always routed to the destination t through a $S_{n-1} \notin S$ and an adjacent node of t that is not in F.

Let s be a source, and let $f \leq n-2$ denote the number of faults in S_n. We give and prove the lemmas about *FT-procedure*, $ROUTING(x, t)$ and $ROUTING(x, t^{(x_n,n)})$.

Lemma 5 *For S_n, $|F| \leq n-2$ and $|S| \leq n-1$. If $|S| = n-1$, then $|F| = n-2$.*

Lemma 6 *FT-procedure can always find a $S_{n-1} \notin S$ if $f \leq n-2$ in S_n.*

Proof. Without loss of generality, assume $n \geq 3$. Firstly, we prove Lemma 6 in the special case: The edge (x,t) is faulty and each valid adjacent node of x with $x \in \{t^{(t_i)} | 2 \leq i \leq n-1\}$ is not directly connected to $S_{n-1} \notin S$. We prove that the edge $(x, x^{(n)})$ is non-faulty and there are no faults in $S_{n-1}(x_1)$ by contradiction. Assume that the edge $(x, x^{(n)})$ is faulty. Since $|F| = n-2$ before the faulty edge $(x, x^{(n)})$ is encountered, $f = |F| + 1 = n-1$ in contradiction with $f \leq n-2$. Similarly, we can prove by contradiction that there are no faults in $S_{n-1}(x_1)$. Since $x \in F$ for $x^{(n)}$, $x^{(n)}$ cannot send M to x and must process *FT-procedure*. From Lemma 5, there is at least a node $y \notin F$ incident to the non-faulty edge $(x^{(n)}, y)$, where y is directly connected to $S_{n-1}(y_1) \notin S$. Since $n-2$ faults have been encountered, the edge from y to $S_{n-1}(y_1)$ is non-faulty. *FT-procedure* can always find a $S_{n-1} \notin S$.

We prove Lemma 6 by induction. It is clear that Lemma 6 is correct when there is only a fault in S_n. Assume Lemma 6 is correct when there are $k (\leq n-3)$ faults in S_n. It shows that *FT-procedure* can find a node that is directly connected to a $S_{n-1} \notin S$ after k faults are encountered. When $f = k+1$, let x be the node that receives M after k faults are encountered, then $|F| = k$ and $|S| \leq k+1$ in M. If x encounters $(k+1)$th faults, $|F| = k+1 \leq n-2$ and $|S| \leq k+1 \leq n-1$. If the condition of the special case is satisfied in x, Lemma 6 holds as shown in the proof in the special case. Otherwise, since $|F| = k+1 \leq n-2$ and $|S| \leq k+1 \leq n-1$, there is at least an non-faulty edge $(x, x^{(x_i)})$ incident to x, where $x^{(x_i)}$ is directly connected to $S_{n-1}(x_i) \notin S$. Since $k+1$ faults have been encountered before M is sent to $x^{(x_i)}$, the edge $(x^{(x_i)}, x^{(x_i,n)})$ is non-faulty. *FT-procedure* can always find a $S_{n-1} \notin S$ in S_n. □

Lemma 7 *$ROUTING(x,t)$ is optimal before encountering faults in S_n.*

Lemma 8 *$ROUTING(x, t^{(x_n,n)})$ is optimal before encountering faults in $S(x_n)$.*

Now, we prove **FT_ROUTING**, analyze its properties, and give the length of the path that is decided by **FT_ROUTING** as well as the message complexity of **FT_ROUTING**.

Theorem 1 **FT_ROUTING** *can adaptively find a fault-free path to route messages from a source s to a destination t in S_n with less than $n-2$ faults.*

Corollary 1 **FT_ROUTING** *is deadlock-free and livelock-free.*

For convenience, we call a pair of sending and receiving a step, and denote the length of path by the number of steps here.

Theorem 2 **FT_ROUTING** *can take at most $d(s,t) + 6f$ steps to route messages from a source s to a destination t in S_n with f faults.*

Proof. Let x, y and z be the nodes in the course of **FT_ROUTING**, and let *extra* denote the number of the extra steps induced by faults. When x in $S_{n-1}(t_i)$ encounters faults, it always tries to send M to $x^{(t_j,n)}$ in $S_{n-1}(t_j) \notin S$, where $x^{(t_j,n)} = x^{(n)}$ if $t_j = x_1$. Since $d(\varphi_{ij}(x), t^{(t_j,n)}) = d(x, t^{(t_i,n)})$ and $d(x^{(t_j,n)}, t^{(t_j,n)}) \leq d(x^{(t_j,n)}, \varphi_{ij}(x)) + d(\varphi_{ij}(x), t^{(t_i,n)})$ based on Lemma 1 and 2, $d(x^{(t_j,n)}, t^{(t_j,n)}) \leq d(x^{(t_j,n)}, \varphi_{ij}(x)) + d(x, t^{(t_i,n)})$. Assume that the edge $(x^{(t_j)}, x^{(t_j,n)})$ is non-faulty and faults are not encountered in $S_{n-1}(t_j)$. Let x' be a node in the shortest path from $x^{(t_j,n)}$ to $t^{(t_j,n)}$ and $d(x', t^{(t_j,n)}) = d(\varphi_{ij}(x), t^{(t_j,n)})$, then it takes at most $d(x, x^{(t_j,n)}) + d(x^{(t_j,n)}, x')$ steps to send M from x to x' based on Lemma 1, 2 and 8. Since $d(x^{(t_j,n)}, x') \leq d(x^{(t_j,n)}, \varphi_{ij}(x))$, $d(x, x^{(t_j,n)}) + d(x^{(t_j,n)}, x') \leq d(x, x^{(t_j,n)}) + d(x^{(t_j,n)}, \varphi_{ij}(x)) = d(x, \varphi_{ij}(x))$. It shows that it takes at most $d(x, \varphi_{ij}(x))$ steps to send M to x'. Without loss of generality, assume that $x' = \varphi_{ij}(x)$ in the course of **FT_ROUTING**. We prove *extra* $\leq 6f$ in three cases:

Case 1: x is an adjacent node of t except $t^{(t_n)}$, i.e., $x \in \{t^{(t_i)} | 2 \leq i \leq n-1\}$.

Case 2: x is a destination node of $S_{n-1}(t_i)$ except t, i.e., $x \in \{t^{(t_i,n)} | 2 \leq i \leq n-1\}$.

Case 3: x is a node except the nodes in case 1 and 2.

Let f_1 denote the number of faults that are encountered after M is sent to some node that belongs to case 1, and let f_2 denote the number of faults that are encountered after M is sent to some node that belongs to case 2 and before M is sent to some node that belongs to case 1, and let f_3 denote the number of faults that are encountered before M is sent to some node that belongs to case 2. Then, $f = f_1 + f_2 + f_3$.

Case 1. We prove it by induction. Assume $f_2 = f_3 = 0$. When $f = 1$, let $x = t^{(t_i)}$, $2 \le i \le n-1$, and the edge (x, t) be faulty. **FT_ROUTING** takes 4 steps to route M from x to $\varphi_{nj}(x)$ in $S_{n-1}(t_j) \notin S$ based on Lemma 2, 6 and 8, and 1 steps from $\varphi_{nj}(x)$ to $t^{(t_j,n)}$ based on Lemma 1 and 8, and 1 step from $t^{(t_j,n)}$ to $t^{(t_j)}$. Since $d(t^{(t_j)}, t) = d(t^{(t_i)}, t)$, **FT_ROUTING** takes 6 extra steps to complete routing.

When $f = f_1 - 1 \le n-4$, assume that **FT_ROUTING** takes $6(f_1 - 1)$ extra steps to route M to $t^{(t_k)} \notin F$. It means that M can be routed to a node $y = t^{(t_j)}$ in $6(f_1 - 2)$ extra steps. When the faulty edge (y, t) that is the $(f_1 - 1)$th fault is encountered in y, M is routed in 6 steps from y to $t^{(t_k)}$ through $S_{n-1}(t_k) \notin S$. Let y^1, y^2 and y^3 be in the path from y to $\varphi_{nk}(y)$, and let $y^0y^1y^2y^3y^4y^5y^6$ denote the path from y to $t^{(t_k)}$ as shown in Fig.3, where $y^0 = y$, $y^4 = \varphi_{nk}(y)$, $y^5 = t^{(t_k,n)}$, and $y^6 = t^{(t_k)}$. **FT_ROUTING** can take m steps to route M from y to y^m.

Fig.3 shows the routing procedure from y to y^m, and from y^m to its image node in a fault-free $S_{n-1}(t_\alpha)$, $1 \le \alpha \le n-1$, as well as from this image node to $t^{(t_\alpha,n)}$, and from $t^{(t_\alpha,n)}$ to t. When the f_1th fault is encountered in y^m, M is routed to t through a fault-free $S_{n-1}(t_\alpha)$. Let $\varphi_{\beta\alpha}(y^m)$ be the image node of y^m in $S_{n-1}(t_\alpha)$, where $\beta = n$ if m =0, 1 or 6 and $\beta = k$ if $2 \le m \le 5$. Then, $extra = 6(f_1 - 2) + m + d(y^m, \varphi_{\beta\alpha}(y^m)) + d(\varphi_{\beta\alpha}(y^m), t^{(t_\alpha,n)}) + 1$ based on Lemma 1, 2, 6 and 8, where $d(y^m, \varphi_{\beta\alpha}(y^m)) \le 4$ from Lemma 2. When $0 \le m \le 1$, since y^m is in $S_{n-1}(t_n)$ and $d(y^m, t) = 1 + m \le 2$, $d(\varphi_{n\alpha}(y^m), t^{(t_\alpha,n)}) = d(y^m, t) \le 2$ from Lemma 1. Therefore, $extra \le 6(f_1 - 1) + 2 < 6f_1$. When $2 \le m \le 5$, since $m + d(\varphi_{k\alpha}(y^m), t^{(t_\alpha,n)}) = m + d(y^m, y^5) = 5$, $extra \le 6(f_1 - 1) + 4 < 6f_1$. It is clear that $extra = 6f_1$ when $m = 6$. For any y^m, $0 \le m \le 6$, $extra \le 6f_1$.

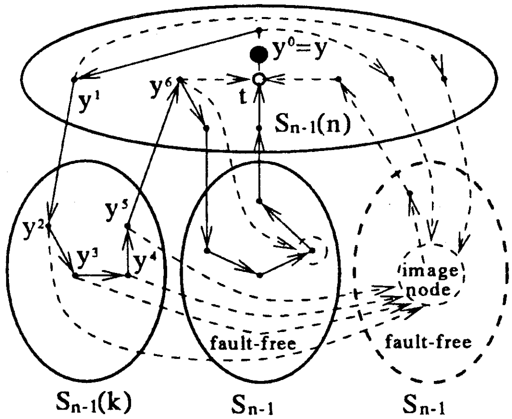

Fig.3: The fault-tolerant routing procedure of **FT_ROUTING**.

When $y = t^{(t_j)}$ with $j \ne 1$ sends M to $y^{(n)} = t_n...t_1...t_j$ in $S_{n-1}(t_j)$, $f = n-2$. If only $S_{n-1}(t_1) \notin S$, $y^{(n)}$ sends M to $z = y^{(n,t_1)} = t_1...t_n...t_j$ in $S_{n-1}(t_j)$, and z sends M to $z^{(t_j)} = t_j...t_n...t_1$ in $S_{n-1}(t_1)$. By $ROUTING(z^{(n)}, t^{(n)})$, $z^{(t_j)}$ sends M to $t^{(n)} = z^{(t_j,n)} = t_n...t_j...t_1$. It takes 4 steps from $t^{(t_j)}$ to $t^{(n)}$, and $extra = 6(n-3) + 4 = 6f - 2$. If $S_{n-1}(t_1) \cup S_{n-1}(t_\alpha) \notin S$ with $\alpha \ne 1$ in y, there are at least two faulty edges incident to y, which have been not encountered before M is sent to y. It shows that $|F| \le n-4$ when y received M and M was sent to y in at most $6(n-4)$ steps. Since it takes one step from y to $y^{(n)}$ and 5 steps from $y^{(n)}$ to $t^{(t_\alpha,n)} \notin F$ through $S_{n-1}(t_\alpha) \notin S$, $extra = 6(n-3) < 6f$. Thus, Theorem 2 holds in case 1.

Case 2. Similarly, we can prove case 2 by induction. Assume $f_1 = f_3 = 0$. When $f = 1$, let $x = t^{(t_i,n)}$ and the edge $(t^{(t_i,n)}, t^{(t_i)})$ is faulty. By *FT-procedure*, $S_{n-1}(x_1) \cup S_{n-1}(x_n) \in S$, where $S_{n-1}(x_1) = S_{n-1}(t_n)$ and $S_{n-1}(x_n) = S_{n-1}(t_i)$ since $x_1 = t_n$ and $x_n = t_i$. Let M be sent to $S_{n-1}(t_j) \notin S$. Based on Lemma 2, it takes at most 4 steps for x to route M to $t^{(t_j,n)}$. When $x = t^{(t_i,n)} = t^{(n)}$, we can prove that it takes 4 steps from $t^{(t_1,n)} = t^{(n)}$ to $t^{(t_j,n)}$ for $j \ne 1$. It is just the reverse procedure from $t^{(t_j,n)}$ to $t^{(n)}$ as shown in the proof of case 1. Assume that it can take $4(f_2 - 1)$ extra steps to route M to t when $f = f_2 - 1$. As same as the proof of case 1, $extra = 4f_2$ when $f = f_2$. When $f_1 \ne 0$ and $f_3 = 0$, $extra = 6f_1 + 4f_2 \le 6f$. Theorem 2 holds in case 2.

Case 3. Let *length* denote the length of the path from s to t, which is decided by **FT_ROUTING**. Assume $f_1 = f_2 = 0$. When $f = 1$, only one of $S_{n-1}(x_1)$ and $S_{n-1}(x_n)$ is in S. Let x be in $S(t_i)$, and let M be sent to $S_{n-1}(t_j) \notin S$. Let $y = \varphi_{ij}(x)$, then $S_{n-1}(y_n) = S_{n-1}(x_1)$ if $y_n = x_1$ or $S_{n-1}(y_n) = S_{n-1}(x_n)$ if $y_n = x_n$. Based on Lemma 2 and 8, it takes at most 2 steps to route M from x to y in $S_{n-1}(y_n) = S_{n-1}(t_j)$. Therefore, $length = d(s,x) + d(x,y) + d(y, t^{(t_j,n)}) + 2$. Since $d(x,y) \le 2$ and $d(s,x) + d(y, t^{(t_j,n)}) = d(s,x) + d(x, t^{(t_i,n)}) \le d(s,t)$ based on Lemma 1 and 4, $length \le d(s,t) + 4$. It is easy to prove by induction that $length \leq d(s,t) + 4f_3$. So, $extra \leq 4f_3$. When $f_1 \neq 0$ and $f_2 \neq 0$, $extra \leq 6f_1 + 4f_2 + 4f_3 \leq 6f$. Theorem 2 holds in case 3. □

Corollary 2 **FT_ROUTING** *is optimal if no faults are encountered in routing.*

Corollary 3 *The number of M transmitted from s to t in the course of* **FT_ROUTING** *is less than or equal to* $d(s,t) + 6f$ *in* S_n *with f faults.*

4 Conclusions

In this paper, we presented an adaptive distributed fault-tolerant routing algorithm without deadlock and livelock for the star graph. It is simple and easy for implementation. The message header only needs to be updated when faults are encountered. The upper bound of the message header is $2n-2$ for the n-star graph. This algorithm can always route messages to a given destination by finding a fault-free $n-1$-star graph based on the local failure information and the topological property of the star graph. Thus, it insure that the routing procedure is deadlock-free and livelock-free. If there are $f \leq n-2$ faults(node faults and/or edge faults) in the n-star graph, it can find a path of length at most $d(s,t) + 6f$ to route messages from a source s to a destination t.

References

[1] S. B. Akers, D. Harel, and B. Krishnamuthy. "The star graph: An attractive alternative to the n-cube". *Proc. Int. Conf. Parallel Proceeding*, pages 393–400, 1987.

[2] S. B. Akers and B. Krishnamurthy. "A Group-Theoretic Model for Symmetric Interconnection Networks". *IEEE Trans. Computs.*, 38(4):555–566, 1989.

[3] N. Bagherzadeh, N. Nassif, and S. Latifi. "A Routing and Broadcasting Scheme on Faulty Star Graphs". *IEEE Trans. Computs.*, 42(11):1398–1403, 1993.

[4] L. Chungti, B. Sourav, and T. Jack. "Performance Evaluation of Fault-tolerant Routing on Star Networks". *Proceedings of the Scalable High Performance Computing Conference*, pages 650–657, 1994.

[5] Qian-Ping Gu and Shietung Peng. "Linear Time Algorithm for Fault Tolerant Routing in Hypercubes and Star Graphs". *IEICE Trans. Inf. Systs.*, E78-D(9):1171–1177, 1995.

[6] Qian-Ping Gu and Shietung Peng. "Set-To-Set Fault Tolerant Routing in Star Graphs". *IEICE Trans. Inf. Systs.*, E79-D(4):282–289, 1996.

[7] Z. Jovanovic and J. Misic. "Fault Tolerance of the Star Graph Interconnection Network". *Information Processing Letters*, 49:145–150, 1994.

[8] J. S. Jwo, S. Lakshmivarahan, and S. K. Dhall. "Embeding of cycles and grids in star graphs". *J. Circuits, Syst., Comput.*, 1(1):43–74, 1991.

[9] Youran Lan. "An Adaptive Fault-tolerant Routing Algorithm for Hypercube Multicomputers". *IEEE Trans. Parallel. Distrib Systs.*, 6(11):1147–1152, 1995.

[10] Latifi-Shahram. "Parallel dimension permutations on star-graph". *Proceedings of the IFIP WG10.3 Working on Architectures and Compilation Techniques for Fine and Medium Grain Parallelism. Orlando, FL, USA*, pages 191–201, 1993.

[11] James A. McHugh. *"Algorithmic Graph Theory"*. Prentice-Hall Inc., 1990.

[12] Y. Rouskov, S. Latify, and P. K. Srimani. "Conditional Fault Diameter of Star Graph Networks". *Journal of Parallel and Distributed Computing.*, 33(1):91–98, 1996.

[13] Ramaraghavan Srinivasan, Vipin Chaudhary, and Syed M Mahmud. "Contention Sensitive Fault-tolerant Routing Alrorithm For Hypercubes". *International Symposium on Parallel Architectures, Algorithms and Networks*, pages 197–204, 1994.

[14] Chien-Chun Su and K. G Shin. "Adaptive Fault-Tolerant Deadlock-Free Routing in Mesh and Hypercubes". *IEEE Trans. Computs.*, 45(6):666–683, 1996.