A Fault-Tolerant Routing Algorithm with Safety Vectors on the (n, k)-star Graph

Chiao-Wei Chiu, Chang-Biau Yang, Kuo-Si Huang, Chiou-Ting Tseng
Department of Computer Science and Engineering
National Sun Yat-sen University, Kaohsiung, Taiwan
Email: cbyang@cse.nsysu.edu.tw

***Abstract*—In this paper, we focus on the design of the fault-tolerant routing algorithm for the (n, k)-star graph. We apply the idea of collecting the limited global information used for routing on the n-star graph to the (n, k)-star graph. First, we build the probabilistic safety vector (PSV) with modified cycle patterns. Then, our routing algorithm decides the fault-free routing path with the help of PSV. The performance is judged by the average length of routing paths. Compared with distance first search and safety level, we get the best performance in the simulations.**

***Keywords*-interconnection network; (n, k)-star graph; probabilistic safety vector; fault-tolerant routing**

I. Introduction

Interconnection networks play an important role in illustrating the performance of a parallel multi-computer system. Hypercubes are one of the famous interconnection networks. The n-star graph [1], denoted as S_n, has been recognized as an alternative to the hypercube. There is a large gap between the number of nodes from S_n to S_{n+1}. The (n, k)-star graph, denoted as $S_{n,k}$, the generalized version of S_n, was proposed by Chiang and Chen [2]. $S_{n,k}$ has better scalability and it preserves many attractive properties of S_n, such as *node symmetry*, *distance*, *diameter*, *hierarchical structure*, and *faultless shortest routing* [1], [2]. Recent research results on *broadcasting* [3], *topological properties* [4], *hamiltonian connectivity* [5] and *weak-vertex-pancyclicity* [6] demonstrate that $S_{n,k}$ is a very powerful network.

In interconnection networks, *routing* refers to the process of selecting a path of *unicast*, which sends a message from a single source to a single destination through some intermediate nodes. For the reason of efficiency, the length of the path decided by the routing algorithm should be as short as possible. A node is said to be *faulty* if it malfunctions. It can be viewed as that this node and the connected edges are removed from the graph. When some nodes are faulty in $S_{n,k}$, the faultless shortest routing may not be available. The main challenge in this paper is to design a near-optimal routing algorithm on $S_{n,k}$ with low complexity. Several fault-tolerant routing algorithms on S_n have been proposed [7]–[9]. We need some new approaches to record the accessibility of each node to assist us for routing on $S_{n,k}$.

The rest of this paper is organized as follows. In Section II, we give an introduction to the properties of $S_{n,k}$. In Section III, we give a basic routing method on $S_{n,k}$ and introduce previously proposed routing algorithm for S_n, *safety level* algorithm. Section IV is the main body of our method. We extend the probabilistic safety vector of S_n by adding new *cycle patterns* to perform routing on $S_{n,k}$. In Section V, we show the experimental results of our method and some comparisons with other methods. Finally, we conclude this paper in Section VI.

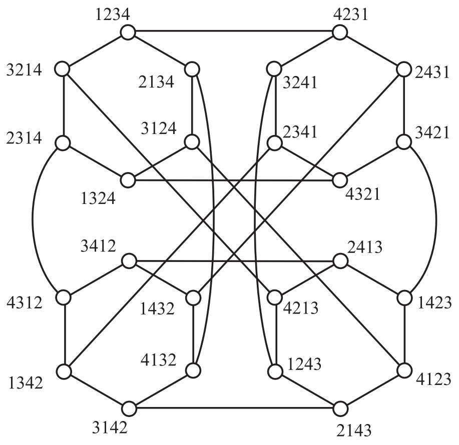

Figure 1. Diagram of 4-star graph S_4.

II. Preliminaries

A. The n-star and (n, k)-star Graphs

The definition of n-star graph (S_n) is proposed in [1]. Each node in S_n is identified by a distinct permutation of the set $\{1, 2, \cdots, n\}$. Nodes $u = u_1u_2 \cdots u_n$ and v are adjacent if we can get v by swapping the first symbol of u with another symbol, that is $v = u_iu_2 \cdots u_{i-1}u_1u_{i+1} \cdots u_n$. Figure 1 shows S_4 as an example.

The (n, k)-star graph [2], denoted as $S_{n,k}$, is a generalized version of the n-star graph. Similarly, each node of $S_{n,k}$ is identified by a distinct permutation with length k selected from $\{1, 2, \cdots, n\}$, where n and k, $1 \le k \le n-1$, are the numbers of available symbols for selection and being selected, respectively. A *node* u of $S_{n,k}$ is denoted by label $u_1u_2 \cdots u_k$, where $u_i \in \{1, 2, \cdots, n\}$ for $1 \le i \le k$, and $u_i \ne u_j$ for $1 \le i < j \le k$. In a node u, the symbols belonging to u are *internal symbols*, and the others are

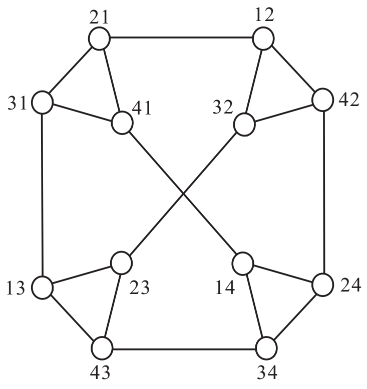

Figure 2. Diagram of (4,2)-star graph $S_{4,2}$.

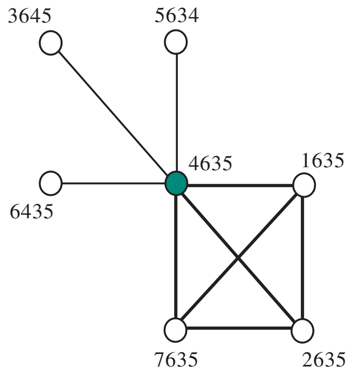

Figure 3. Neighbors of $u = 4635$ in $S_{7,4}$.

external symbols. For example, permutations 7352 and 1583 are two nodes in $S_{8,4}$. In node 7352, internal symbols are 2, 3, 5 and 7; external symbols are 1, 4, 6 and 8. The adjacency in $S_{n,k}$ is more generalized than that in S_n.

In $S_{n,k}$, nodes $u = u_1u_2 \cdots u_k$ and v are adjacent if it satisfies one of following cases. Case 1: $v = u_iu_2u_3 \cdots u_{i-1}\ u_1u_{i+1} \cdots u_k$; Case 2: $v = \beta u_2u_3 \cdots\ u_k$ where $\beta \in \{1, 2, \cdots, n\} \setminus \{u_1, u_2, \cdots, u_k\}$.

In the above definition of edges, Case 1 is similar to S_n. In Case 1, since all symbols of u are the same as those of v, edge (u, v) is called an *internal edge*. In Case 2, since the first symbol of v is an external symbol of u, edge (u, v) is called an *external edge*. For example, a diagram of $S_{4,2}$ is shown in Figure 2. The edge from 42 to 24 is an internal edge, while the edge from 42 to 32 is an external edge.

A graph is called a *regular graph* if the degrees of all nodes are identical. S_n and $S_{n,k}$ are both regular graphs with degree $n-1$. If node u is adjacent to node v, we say that node u is a *neighbor* of node v. For the completeness, we denote the neighbor of node u as $u^{(\alpha)}$, where $\alpha \in \{1, 2, \cdots, n\} \setminus \{u_1\}$. For example, in Figure 3, consider $u = 4635$ in $S_{7,4}$. The neighbors of u are $u^{(1)} = 1635$, $u^{(2)} = 2635$, $u^{(3)} = 3645$, $u^{(5)} = 5634$, $u^{(6)} = 6435$ and $u^{(7)} = 7635$. Note that for $u = 4635$, there is no $u^{(4)}$ since 4 is the first symbol of u. In this figure, the thin and bold lines represent internal and external edges of u, respectively.

According to the definition of $S_{n,k}$, the following properties have been proved [2].

Property 1: [2] There are $n!$ and $\frac{n!}{(n-k)!}$ nodes in S_n and $S_{n,k}$, respectively. $S_{n,k}$ has better scalability than S_n.

Property 2: [2] $S_{n,1}$ is isomorphic to the complete graph, n-clique, and $S_{n,n-1}$ is isomorphic to S_n.

B. Permutation Cycles of (n, k)-star Graphs

In $S_{n,k}$, the paths can be represented by *cycles* [2]. For source $u = u_1u_2 \cdots u_k$ and destination v, a cycle is defined as $\gamma = \{s_1, s_2, \cdots, s_l\}$, where $\gamma \subset \{u_i | 1 \leq i \leq k\}$ and l is the length of the cycle. If all symbols in γ are internal symbols of v, we say γ is an *internal cycle* and denote it with a pair of parentheses, $(s_1s_2 \cdots s_l)$. If γ has an external symbol of v, we put it after s_l, and we say that γ is an *external cycle*. There must exist a desired symbol t of s_l, where t belongs to v. We denote γ with a pair of angle brackets, $\langle s_1s_2 \cdots s_l, t\rangle$. Note that there is no more than one external symbol in a cycle.

For example, consider source $u = 21745$ and destination $v = 14526$ in $S_{7,5}$, the cycles are $\binom{14526}{21745} = (241)\langle 57, 6\rangle = \langle 57, 6\rangle(241)$. The *faultless shortest routing* rule follows the order of the symbols in the cycles to route the message. For example, in $S_{7,5}$, two of the shortest paths from 21745 to 14526 are given as follows.

Path 1:21745 $\rightarrow$ 41725 $\rightarrow$ 14725 $\rightarrow$ 54721 $\rightarrow$ 74521 $\Rightarrow$ 64521 $\rightarrow$ 14526

Path 2:21745 $\rightarrow$ 51742 $\rightarrow$ 71542 $\Rightarrow$ 61542 $\rightarrow$ 21546 $\rightarrow$ 41526 $\rightarrow$ 14526

It takes 6 steps from 21745 to 14526, and we say that the path length from 21745 to 14526 is 6.

C. The Distance Computation of (n, k)-star Graphs

The *distance* between two nodes u and v, denoted as $d(u, v)$, is the length of the shortest path from u to v. The distance of two nodes in a faultless $S_{n,k}$ is computable. The computation of $d(u, v)$ is given as follows [2].

$$d(u, v) = \begin{cases} c + m + e, & \text{if } u_1 = v_1; \\ c + m + e - 2, & \text{if } u_1 \neq v_1, \end{cases} \qquad (1)$$

where c is the number of cycles, m is the number of misplaced symbols, and e equals 0 or 1 if there exists no external cycle or otherwise, respectively.

For example, in $S_{7,5}$, from 21745 to 14526, $\binom{14526}{21745} = (241)\langle 57, 6\rangle$, we get $c = 2$, $m = 5$, $e = 1$, and $u_1 \neq v_1$. Then we get $d(21745, 14526) = 6$ by Equation (1). Note that the distance of two nodes in a faultless S_n can also be computed by Equation (1) with $e = 0$.

The longest distance for all pairs of nodes in $S_{n,k}$ is called the *diameter* of $S_{n,k}$, which is denoted as $D(S_{n,k})$ and it

Table I
THE COMPARISON OF S_n AND $S_{n,k}$.

Graph	Size	Degree	Diameter
$S_{4,2}$	12	3	3
$S_{5,2}$	20	4	3
$S_4 = S_{4,3}$	24	3	4
$S_{6,2}$	30	5	3
$S_{5,3}$	60	4	5
$S_5 = S_{5,4}$	120	4	6
$S_{6,3}$	120	5	5
$S_{6,4}$	360	5	6
$S_6 = S_{6,5}$	720	5	7
$S_{n,k}$, $1 \leq k \leq \lfloor \frac{n}{2} \rfloor$	$\frac{n!}{(n-k)!}$	$n-1$	$2k-1$
$S_{n,k}$, $\lfloor \frac{n}{2} \rfloor + 1 \leq k \leq n-2$	$\frac{n!}{(n-k)!}$	$n-1$	$k + \lfloor \frac{n-1}{2} \rfloor$
$S_n = S_{n,n-1}$	$n!$	$n-1$	$\lfloor \frac{3n-3}{2} \rfloor$

can be computed as follows [2].

$$D(S_{n,k}) = \begin{cases} 2k-1, & \text{if } 1 \leq k \leq \lfloor \frac{n}{2} \rfloor; \\ k + \lfloor \frac{n-1}{2} \rfloor, & \text{if } \lfloor \frac{n}{2} \rfloor + 1 \leq k \leq n-1. \end{cases} \quad (2)$$

By Property 2, diameter $D(S_n) = D(S_{n,n-1})$.

For routing from nodes u to v in a faultless $S_{n,k}$, the neighbors of a node u can be classified into three kinds as follows. The neighbors of u on the shortest paths to v are called *preferred neighbors*. The distance from preferred neighbors of u to v is $d(u,v)-1$. The neighbors of u with the same distance to v are called *semi-preferred neighbors*. The neighbors of u with distance $d(u,v)+1$ to v are called *non-preferred neighbors*. For example, in Figure 2, for routing from nodes 14 to 23, node 34 is a preferred neighbor, nodes 41 and 24 are semi-preferred neighbors; for routing from nodes 12 to 31, node 21 is a preferred neighbor, nodes 32 and 42 are non-preferred neighbors. Table I shows the comparison of S_n and $S_{n,k}$ for the properties of size, degree and diameter, where size indicates the number of nodes contained in the graph.

III. PREVIOUS WORK

In this paper, we try to design an algorithm to perform routing on a faulty $S_{n,k}$. When node u wants to send a message to destination v or receives from node t a message which is sent to destination v, this routing algorithm will decide which neighbor to route the message through. Then u will send the message to this neighbor in the next step. To avoid the routing loop between nodes, we only consider the neighbors not visited. If there exists no available neighbor, u will ignore the message or send the message back to t. If destination v is reachable, the routing path is composed of all the nodes that the message travels. The length of the routing path is one of the criteria for measuring the performance of routing algorithm.

A routing algorithm can make the decision based on one of the three types of information, *local information*, *global information* and *limited global information*. Local information is that each node only knows the states of its neighbors. We give a greedy algorithm, *distance first search* (DFS), based on local information for $S_{n,k}$. Global information is to take care of the state of the whole graph. With the global information, the *Floyd-Warshall* algorithm [10] can solve the all-pairs shortest path problem on a graph with time complexity $O(|V|^3)$, where V is the node set, and perform the optimal routing for each pair of nodes. However, it is not efficient while the size of the graph is large. Limited global information considers a limited amount of global information. Each method requires a process to collect the fault information of the neighbors. We introduce the previous method, *safety level* [11] for routing on S_n.

A. Routing on $S_{n,k}$ with Distance First Search (DFS)

Because the distance of two nodes can be computed immediately by Equation (1), the heuristic is to route the message through the nearest neighbor. With the local information of the neighbors, we can design a greedy *distance first search* (DFS) algorithm to perform routing for each node in $S_{n,k}$, which returns the decision of each node. In this DFS algorithm, a non-visited neighbor is selected according to its distance to destination. That is, the neighbor selecting shall follow the order: preferred neighbors, semi-preferred neighbors, and non-preferred neighbors.

In our observations, the performance of DFS becomes worse when the number of faulty nodes increases. The local information of neighbors may not be sufficient for deciding a good routing path.

B. Routing on S_n with Safety Levels (SL)

We allow the nodes to communicate their information with their neighbors before routing, and collect the limited global information as the routing ability of a node. The *safety level* can be used to represent the ability to perform routing on hypercubes [11]. With the same concept, the safety level can also be used to represent the routing ability on S_n [9]. The safety level of a node is an integer σ, which ensures that all the nodes of the distance less than or equal to σ can be reached with the shortest path.

The safety level of node u, denoted as $\sigma(u)$, is computed iteratively as follows [9].

For i from 1 to diameter $D(S_n)$, compute

$$\sigma(u) = \begin{cases} 0, & \text{if } u \text{ is faulty;} \\ min\{\sigma(u^{(\alpha)}) | \alpha \neq u_1\} + 1, & \text{otherwise.} \end{cases} \quad (3)$$

The routing algorithm [9] with safety levels for S_n based on the guarantee of safety level. If there exists a neighbor in the optimal path guaranteed by safety level, this neighbor shall be selected; otherwise we select the neighbor with the least distance and the maximal safety level.

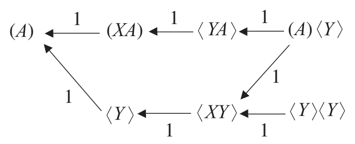

Figure 4. Connection structure of cycle patterns in $S_{4,2}$.

IV. Fault-Tolerant Routing on the (n,k)-star Graph

A. The Modified Cycle Pattern of the (n,k)-star Graph

The cycle patterns in $S_{n,k}$ are different from that in S_n. The original representations used in S_n are not enough to represent the external cycle in $S_{n,k}$. We add a new symbol Y to describe the external cycles in $S_{n,k}$. A *modified cycle pattern* is represented by one A, some Xs and some Ys, where A, X, and Y denote the first symbol, any other internal symbol and any other external symbol of the destination, respectively. An internal cycle $\gamma_i = (s_1 s_2 \cdots s_l)$ belongs to either $(XXX \cdots XX)$ or $(XXX \cdots XA)$. An external cycle $\gamma_e = \langle s_1 s_2 \cdots s_l, t\rangle$ belongs to either $\langle XXX \cdots XXY\rangle$ or $\langle XXX \cdots XYA\rangle$. For example, in $S_{7,5}$, $\binom{12345}{34561} = (351)\langle 46,2\rangle$ and $\binom{14526}{21745} = (241)\langle 57,6\rangle$ both belong to $(XXA)\langle XY\rangle$; $\binom{12345}{72634} = \langle 7,1\rangle\langle 436,5\rangle$ belongs to $\langle Y\rangle\langle XXY\rangle$, and $\binom{12345}{43276} = (23)\langle 6,5\rangle\langle 47,1\rangle$ belongs to $(XX)\langle Y\rangle\langle XY\rangle$.

We say that cycle pattern P_1 has an edge to cycle pattern P_2 if P_1 can be changed to P_2 by a single swap. The definition of *preferred cycle patterns* (PCP) of a cycle pattern is the same as that in S_n. If P_1 has an edge to P_2 and $d(P_2) = d(P_1) - 1$, we say that P_2 is a *preferred cycle pattern* of P_1, and the relationship can be denoted as $P_2 \in PCP(P_1)$. The connection structures of cycle patterns in $S_{4,2}$ and $S_{5,3}$ are shown in Figures 4 and 5, respectively, where the arrow head indicates that the distance of the cycle pattern in the head side is less than the other one, and the number aside each arrow describes the number of ways to change the cycle pattern into its corresponding preferred cycle patterns. For example, $\langle YA\rangle$ and $\langle XY\rangle$ are both preferred cycle patterns of $(A)\langle Y\rangle$, and we have $PCP((A)\langle Y\rangle) = \{\langle YA\rangle, \langle XY\rangle\}$. In fact, Figure 4 is a subgraph of Figure 5 since $S_{4,2}$ is a subgraph of $S_{5,3}$.

Given a node u and a cycle pattern P, the *neighbor pattern set* consists of the neighbors of u associated with the preferred cycle patterns of P. Let $\Omega(u,P)$ denote the set of the possible neighbor pattern sets of node u and cycle pattern P. A possible neighbor pattern is denoted as $u^{(\alpha)} \ominus P'$ where $u^{(\alpha)}$ and P' are a neighbor node and a PCP of P, respectively. For example, consider node $u = 12$ in $S_{4,2}$ and the connection structure of cycle patterns in Figure 4. The neighbors of node u are $u^{(2)} = 21$, $u^{(3)} = 32$, $u^{(4)} = 42$. Since $PCP((A)\langle Y\rangle) = \{\langle YA\rangle, \langle XY\rangle\}$, $\Omega(u,(A)\langle Y\rangle) = \{\{21 \ominus \langle YA\rangle,\ 32 \ominus \langle XY\rangle\},\ \{21 \ominus \langle YA\rangle,\ 42 \ominus \langle XY\rangle\},$ $\{32 \ominus \langle YA\rangle,\ 21 \ominus \langle XY\rangle\},\ \{32 \ominus \langle YA\rangle,\ 42 \ominus \langle XY\rangle\},$ $\{42 \ominus \langle YA\rangle,\ 21 \ominus \langle XY\rangle\},\ \{42 \ominus \langle YA\rangle,\ 32 \ominus \langle XY\rangle\}\}$. Since $PCP(\langle Y\rangle\langle Y\rangle) = \{\langle XY\rangle\}$, $\Omega(u,\langle Y\rangle\langle Y\rangle) = \{\{21 \ominus \langle XY\rangle\},\ \{32 \ominus \langle XY\rangle\},\ \{42 \ominus \langle XY\rangle\}\}$.

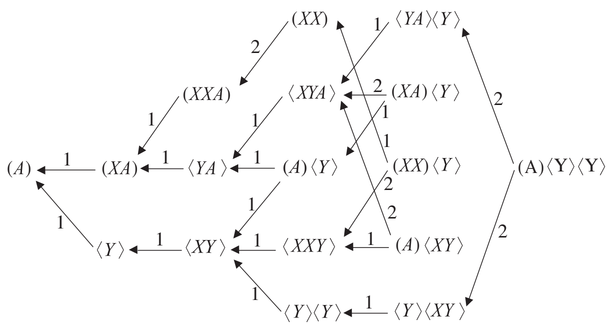

Figure 5. Connection structure of cycle patterns in $S_{5,3}$.

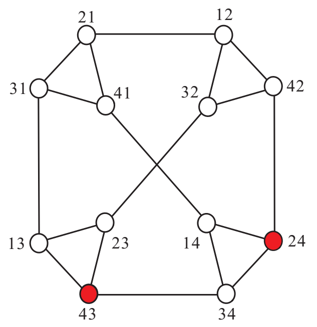

Figure 6. $S_{4,2}$ with two faulty nodes 24 and 43.

B. The Computation of PSV in $S_{n,k}$

In $S_{n,k}$, similar to S_n, nodes with the same cycle pattern have similar faultless shortest paths. The nodes exchange the information of their routing ability of cycle patterns with their neighbors before routing. The probabilistic safety vector (PSV) is used to store the routing abilities of all cycle patterns for each node. Similar to S_n, the value of cycle pattern P in the PSV of node u is denoted as $PSV(u)[P]$, which is computed by following steps.

Step 1: Initially,

$$PSV(u)[P] = \begin{cases} 1, & \text{if } P = (A) \text{ and } u \text{ is faultless;} \\ 0, & \text{if } u \text{ is faulty.} \end{cases}$$

Step 2: For each P, $d(P)$= 1 to $D(S_{n,k})$, compute $PSV(u)[P] = \frac{1}{|\Omega(u,P)|} \sum_{S \in \Omega(u,P)} \max\{PSV(u^{(\alpha)})[P'] | u^{(\alpha)} \ominus P' \in S\}$.

Table II shows the PSV of each node in $S_{4,2}$ with two faulty nodes, 24 and 43, as shown in Figure 6.

C. Routing on $S_{n,k}$ with Probabilistic Safety Vectors

In S_n, one node may have two kinds of neighbors, preferred and non-preferred. However, in $S_{n,k}$, one more kind of neighbor, semi-preferred, exists. Thus, the routing algorithm with PSV for $S_{n,k}$ is redesigned as follows.

Table II
PROBABILISTIC SAFETY VECTORS OF FIGURE 6.

Label of node	**Probabilistic safety vector**					
	$\langle XA\rangle$	$\langle Y\rangle$	$\langle XY\rangle$	$\langle YA\rangle$	$\langle Y\rangle\langle Y\rangle$	$\langle A\rangle\langle Y\rangle$
12	1	1	0.89	0.89	0.81	0.93
13	0.67	0.67	0.56	0.56	0.48	0.78
14	0.67	0.67	0.44	0.44	0.37	0.67
21	1	1	1	1	0.89	0.89
23	0.67	0.67	0.56	0.56	0.44	0.7
24	0	0	0	0	0	0
31	1	1	0.89	0.89	0.81	0.96
32	1	1	0.78	0.78	0.7	0.81
34	0.33	0.33	0.22	0.22	0.15	0.3
41	1	1	0.89	0.89	0.78	0.96
42	0.67	0.67	0.67	0.67	0.56	0.85
43	0	0	0	0	0	0

```
Input: Node $u$, destination $v$, threshold $\theta$, and PSV
Output: $u^{(\alpha)}$
  if $u = v$ then
    stop.
  else if $\exists u^{(\alpha)} \in$ {preferred neighbors} \ {visited neighbors} such that $PSV(u^{(\alpha)})[P(u^{(\alpha)}, v)] \geq \theta$ then
    return $u^{(\alpha)}$. {/* Check optimal paths */}
  else if $\exists u^{(\alpha)} \in$ {semi-preferred neighbors} \ {visited neighbors} such that $PSV(u^{(\alpha)})[P(u^{(\alpha)}, v)] \geq \theta$ then
    return $u^{(\alpha)}$. {/* Check suboptimal paths */}
  else if $\exists u^{(\alpha)} \in$ {non-preferred neighbors} \ {visited neighbors} such than $PSV(u^{(\alpha)})[P(u^{(\alpha)}, v)] \geq \theta$ and $P(u^{(\alpha)}, v) \neq (X \cdots XA)$ then
    return $u^{(\alpha)}$. {/* No shorter path via $(X \cdots XA)$ */}
    {No guarantee}
  else if $\exists u^{(\alpha)} \in$ {preferred neighbors} \ {visited neighbors} such that $PSV(u^{(\alpha)})[P(u^{(\alpha)}, v)]$ is maximum then
    return $u^{(\alpha)}$.
  else if $\exists u^{(\alpha)} \in$ {semi-preferred neighbors} \ {visited neighbors} such that $PSV(u^{(\alpha)})[P(u^{(\alpha)}, v)]$ is maximum then
    return $u^{(\alpha)}$.
  else if $\exists u^{(\alpha)} \in$ {non-preferred neighbors} \ {visited neighbors} such that $PSV(u^{(\alpha)})[P(u^{(\alpha)}, v)]$ is maximum then
    return $u^{(\alpha)}$.
  end if
```

First, we guarantee the optimal path by checking the routing abilities of cycle patterns in preferred neighbors, and then check the semi-preferred and the non-preferred neighbors sequentially. Because nothing can be sure in the final situation, we select the neighbor with the shortest distance and the best routing ability as a heuristic.

For example, consider the routing from node 14 to node 23 in Figure 6, and suppose that the threshold $\theta = 0.5$. The process of the PSV routing is shown in Table III. The neighbors of node 14 are nodes 34, 41, and 24, which are the preferred neighbor, semi-preferred neighbor, and a faulty node, respectively. $P(34, 23) = \langle XY\rangle, P(41, 23) = \langle Y\rangle\langle Y\rangle$. First, we check $PSV(34)[\langle XY\rangle] = 0.22$, which is less than θ and then it is not selected. Second, we check $PSV(41)[\langle Y\rangle\langle Y\rangle] = 0.78$, which is greater than θ, so we route the message to node 41. In Table III, the neighbor marked with an underline is chosen by the PSV routing algorithm. Collecting the chosen neighbors, we obtain the routing path (14, 41, 31, 13, 23) and its length is 4.

Table III
THE NEIGHBORS OF NODES IN THE PSV ROUTING PATH FROM NODE 14 TO NODE 23 OF FIGURE 6, WHERE THE UNDERLINED NEIGHBOR IS SELECTED BY EACH NODE.

Label of node	**Non-visited neighbors**					
	preferred		semi-preferred		non-preferred	
14	34	$\langle XY\rangle$	$\underline{41}$, 24	$\langle Y\rangle\langle Y\rangle$		
41	$\underline{31}$	$\langle XY\rangle$	21	$\langle Y\rangle\langle Y\rangle$		
31	$\underline{13}$	$\langle Y\rangle$			21	$\langle Y\rangle\langle Y\rangle$
13	$\underline{23}$	$\langle A\rangle$	43	$\langle Y\rangle$		

V. EXPERIMENTAL RESULTS

We provide the simulations of routing ability in some (n, k)-star graphs for the three methods, distance first search (DFS), safety level (SL) and probabilistic safety vector (PSV), and then compare the results with the optimal solution (OPT) obtained by using the Floyd-Warshall algorithm [10], whose time complexity is $O(|V|^3)$, where V is the node set in $S_{n,k}$. Note that the Floyd-Warshall algorithm can find the shortest paths between all pairs of nodes, but it needs the global information of the graph. We set the threshold $\theta = 0.5$ for the PSV value, which means the probability of successful routing with the faultless shortest path is 0.5.

In the first simulation, we focus on $S_{5,3}$, which contains 60 nodes. We consider the cases of 0, 5, 10, 15, 20, 25 and 30 faulty nodes, and randomly generate 500 graphs for each number of faulty nodes. We discard the unreachable paths in this simulation. The result is shown in Table IV(a). For each graph, the average routing length of method M between all pairs of nodes is recorded, which is denoted as $L(M)$. The error rate $\frac{L(M)-L(OPT)}{L(OPT)}$ is shown in the parenthesis following $L(M)$, which makes it easier for us to compare the routing abilities of these methods.

The second simulation considers $S_{6,3}$, which contains 120 nodes. We consider the cases that the number of faulty nodes varies from 0 to 40 with step 5, and randomly generate 200 graphs for each number of faulty nodes. The result is shown in Table IV(b). When the percentage of faulty nodes is less than 12% in $S_{6,3}$, $L(PSV)$ is very close to $L(OPT)$.

In the third simulation, we consider $S_{7,4}$, which contains 840 nodes. We consider the cases of 0, 20, 40, 60, 80, 100, 150, 200 and 250 faulty nodes, and randomly generate 100 graphs for each number of faulty nodes. For each case of method M, we randomly select 1000 pairs of nodes, and

Table IV
COMPARISON OF AVERAGE ROUTING LENGTH OF EACH METHOD.

Number of	Average routing length			
faulty nodes	OPT	DFS	SL	PSV
0 (0%)	3.136	3.136 (0%)	3.136 (0%)	3.136 (0%)
5 (8%)	3.304	3.650 (10%)	3.492 (5%)	3.399 (2%)
10 (16%)	3.514	4.339 (23%)	4.047 (15%)	3.859 (9%)
15 (25%)	3.810	5.423 (42%)	5.085 (33%)	4.706 (23%)
20 (33%)	4.206	6.926 (64%)	6.727 (59%)	6.065 (44%)
25 (41%)	4.697	8.727 (85%)	8.610 (83%)	7.926 (68%)
30 (50%)	4.661	8.139 (74%)	8.067 (73%)	7.504 (61%)

(a) $S_{5,3}$

Number of	Average routing length			
faulty nodes	OPT	DFS	SL	PSV
0 (0%)	3.429	3.429 (0%)	3.429 (0%)	3.429 (0%)
5 (4%)	3.513	3.719 (5%)	3.624 (3%)	3.579 (1%)
10 (8%)	3.602	4.051 (12%)	3.890 (7%)	3.778 (4%)
15 (12%)	3.698	4.434 (19%)	4.205 (13%)	4.021 (8%)
20 (16%)	3.814	4.890 (28%)	4.628 (21%)	4.373 (14%)
25 (20%)	3.945	5.464 (38%)	5.139 (30%)	4.835 (22%)
30 (25%)	4.113	6.208 (50%)	5.915 (43%)	5.519 (34%)
35 (29%)	4.273	6.968 (63%)	6.696 (56%)	6.262 (46%)
40 (33%)	4.520	8.279 (83%)	8.075 (78%)	7.784 (72%)

(b) $S_{6,3}$

Number of	Average routing length			
faulty nodes	OPT	DFS	SL	PSV
0(0%)	4.585	4.585 (0%)	4.585 (0%)	4.585 (0%)
20(2%)	4.650	4.985 (7%)	4.792 (3%)	4.706 (1%)
40(4%)	4.711	5.351 (13%)	5.046 (7%)	4.843 (2%)
60(7%)	4.778	5.667 (18%)	5.304 (10%)	4.994 (4%)
80(9%)	4.825	6.065 (25%)	5.600 (16%)	5.178 (7%)
100(11%)	4.922	6.501 (32%)	5.994 (21%)	5.404 (9%)
150(17%)	5.048	7.279 (44%)	6.751 (33%)	5.936 (17%)
200(23%)	5.271	9.025 (71%)	8.420 (59%)	7.139 (35%)
250(29%)	5.552	10.705 (92%)	10.261 (84%)	9.704 (74%)

(c) $S_{7,4}$

Table V
TIME COMPLEXITIES FOR VARIOUS METHODS ON $S_{n,k}$.

For each node	DFS	SL	PSV	OPT
preprocessing	none	$O(nD)$	$O(n\psi(n,k))$	$O(\|V\|^2)$
routing	$O(n)$	$O(n)$	$O(n)$	$O(1)$

then compute the average length $L(M)$ of reachable paths. The result is shown in Table IV(c). Although the size of $S_{7,4}$ is larger, PSV still gets better result than DFS and SL.

Let $\psi(n,k)$ denote the length of the probabilistic safety vector in $S_{n,k}$. The time required for requesting the information of neighbors to compute the probabilistic safety vector of each node is $O(n\psi(n,k))$, and the space to store the probabilistic safety vector for each node is $O(\psi(n,k))$. In our observation, $\psi(n,k)$ grows slowly when the size of $S_{n,k}$ becomes large. The time complexity for our algorithm to route through one path of length L is $O(nL)$, which is the same as that for DFS and SL. The time complexity of each method on $S_{n,k}$ is shown in Table V, where D and V denote the diameter and the node set of $S_{n,k}$, respectively.

VI. CONCLUSION

In this paper, we apply the idea of collecting the limited global information used for routing on the n-star graph to the (n,k)-star graph. We modifies the cycle patterns of star graph S_n, and propose the routing algorithm for $S_{n,k}$ based on the probabilistic safety vector (PSV). Compared with the distance first search algorithm (DFS) and the safety level algorithm (SL), the average routing length of our method is the best. In the future, it is worth to discuss the relationship between faulty $S_{n,k}$ and the probabilistic safety vector model. It may guide a better way for the modification of the PSV model to get better routing performance.

REFERENCES

[1] S. B. Akers, D. Horel, and B. Krishnamurthy, “The star graph: An attractive alternative to the n-cube,” *Proceeding of the International Conference on Parallel Processing*, pp. 393–400, 1987.

[2] W.-K. Chiang and R.-J. Chen, “The (n,k)-star graph : A generalized star graph,” *Information Processing Letters*, vol. 56, pp. 259–264, Dec. 1995.

[3] J. Li, M. Chen, Y. Xiang, and S. Yao, “Optimum broadcasting algorithms in (n,k)-star graphs using spanning trees,” *IFIP International Federation for Information Processing*, pp. 220–230, 2007.

[4] W.-K. Chiang and R.-J. Chen, “Topological properties of the (n,k)-star graph,” *International Journal of Foundations of Computer Science*, vol. 9, no. 2, pp. 235–248, 1998.

[5] H. Hsu, Y. Hsieh, J. Tan, and L. Hsu, “Fault hamiltonicity and fault hamiltonian connectivity of the (n,k)-star graphs,” *Networks*, vol. 42, no. 4, pp. 189–201, Apr. 2003.

[6] Y.-Y. Chen, D.-R. Duh, T.-L. Ye, , and J.-S. Fu, “Weak-vertex-pancyclicity of (n,k)-star graphs,” *Theoretical Computer Science*, vol. 396, pp. 191–199, 2008.

[7] N. Bagherzadeh, M. Nassif, and S. Latifi, “A routing and broadcasting scheme on faulty star graphs,” *IEEE Transactions on Computers*, vol. 42, no. 11, pp. 1398–1403, Nov. 1993.

[8] Q. P. Gu and S. Peng, “Node-to-node cluster fault routing in star graph,” *Information Processing Letters*, vol. 56, pp. 29–35, 1995.

[9] S.-I. Yeh, C.-B. Yang, and H.-C. Chen, “Fault-tolerant routing on the star graph with safety vectors,” in *Proc. of the Sixth Annual International Symposium on Parallel Architectures, Algorithms, and Networks, I-SPAN02*. Manila, Philippines, May 2002, pp. 301–306.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. MIT Press and McGraw-Hill, 2001.

[11] J. Wu, “Unicasting in faulty hypercubes using safety levels,” *IEEE Transactions on Computers*, vol. 46, no. 2, pp. 241–244, Feb. 1997.