

# A FAULT-TOLERANT RECONFIGURATION SCHEME IN THE FAULTY STAR GRAPH

YUH-SHYAN CHEN<sup>†</sup> and JANG-PING SHEU<sup>‡</sup>

<sup>†</sup>Department of Computer Science

Chung-Hua Polytechnic Institute, Hsin-Chu, 30067, Taiwan

<sup>‡</sup>Department of Computer Science and Information Engineering

National Central University, Chung-Li, 32054, Taiwan

**ABSTRACT** In this paper, we propose a scheme to identify the maximal fault-free substar-ring. This is the first result to derive a reconfiguration scheme with high processor utilization in the faulty  $n$ -star graph. The maximal fault-free substar-ring is connected by a ring of fault-free *virtual substars* with dilation 3 and maximal length of the ring is  $n(n-1)(n-2)$ . Our proposed scheme can tolerate  $n-3$  faults such that the processor utilization is  $\frac{n^2-2n+3}{n^2-n}$ . This is a near optimal result since the maximal fault-free substar-ring is constructed by using all of the possible fault-free  $(n-2)$ -substars. Moreover, our algorithm can still work when number of faults exceeds  $n-3$ . The simulation results also show that the processor utilization is more than 50% if the number of faults is less than  $\frac{n^2-n-1}{2}$  in the  $n$ -star graph.

## I. INTRODUCTION

Recently, one new interconnection network that has attracted lots of attention is the star graph [5]. The star graph [5], being a member of the class of Cayley graphs, has been shown to possess appealing features including low degree of the node, small diameter, partitionability, symmetry, and high degree of fault tolerance. Especially when the size of the star graph system increases, fault tolerance is an important issue for such a large system to continue operations after failure of one or more processors/links. In this paper, we study how algorithms that are originally designed for fault-free star graph can be implemented on star graph containing faults with high processor utilization. To measure the efficiency of fault-tolerant approach, we use *processor utilization ratio* or PUR, which is the total number of fault-free processors used by our reconfiguration scheme divided by the total number of processors of  $n$ -star graph.

Much research recently has directed toward studying the aspects of fault-tolerant computing on the hypercube and star graph [9, 11, 12, 19, 22, 23, 24, 25]. The hierarchy of hypercube and star graph allow the assignment of its special subgraph, subcube and substar, which have the same topological features as the original graph. Most of the fault-tolerant strategies address the issue of *reconfiguration* once the faulty processors/links are identified. One effective approach of the reconfiguration strategies

in hypercubes is to identify the largest fault-free subcube and use the subcube to emulate the entire hypercube [11]. An  $n$ -star graph can be recursively decomposed into  $n$   $(n-1)$ -substars. The largest fault-free substar can be easily identified and used to emulate the entire star graph. It is unreasonable to using the largest fault-free substar approach as reconfiguration scheme since its PUR becomes at most  $1/n$  even when an  $n$ -star graph contains only one fault. A different but related research is studying how to allocate tasks in the complete star graph [15]. Note that this approach is more suitable to accommodate multiple jobs on different size of substars. Therefore, the motivation of this paper is to provide a novel fault-tolerant reconfiguration scheme in  $n$ -star graph so that higher PUR and lower diameter is obtained.

The major contribution of this paper is to propose an efficient reconfiguration scheme, which can tolerate  $f \leq n-3$  faults in  $n$ -star graph, such that PUR is  $\frac{n^2-2n+3}{n^2-n}$  and diameter is  $O(n^3)$ . Furthermore, our scheme with reasonable PUR can still work when number of faults exceeds  $n-3$ . Based on our simulation results, it shows that our algorithm keeps PUR more than 50% if the number of faults is less than  $\frac{n^2-n-1}{2}$  in the  $n$ -star graph. In addition, a novel communication pattern with constant time cost will be presented in this paper for the sake of easily performing algorithm on the maximal fault-free substar-ring. To demonstrate the application capability, we describe how to apply sorting and matrix-multiplication algorithms on the reconfiguration scheme. In this paper, we only consider node faults and an edge fault is assumed that one of the nodes incident upon it is faulty. We also assume that faulty nodes can neither perform calculations nor route data.

## II. PRELIMINARIES

An  $n$ -dimensional star graph, also referred to as  $n$ -star or  $S_n$ , is an undirected graph consisting of  $n!$  nodes (or vertices) and  $(n-1)n!/2$  edges. Each node is uniquely assigned a label  $x_1x_2 \cdots x_n$ , which is the concatenation of a permutation of  $n$  distinct symbols  $\{x_1, x_2, \dots, x_n\}$ . Without loss of generality, let these  $n$  symbols be  $\{1, 2, \dots, n\}$ . Given any node label  $x_1 \cdots x_i \cdots x_n$ , let permutation function  $g_i, 2 \leq i \leq n$ , be such that  $g_i(x_1 \cdots x_i \cdots x_n) = x_i \cdots x_1 \cdots x_n$  (i.e., swap  $x_1$  and  $x_i$

and keep the rest symbols unchanged). In  $S_n$ , for any node  $x$ , there is an edge joining  $x$  and node  $g_i(x)$ , and this edge is said the dimension  $i$ . Each node in  $S_n$  is connected to  $n - 1$  adjacent nodes by  $n - 1$  edges. Let  $F$  denote as a set of faulty nodes in a faulty  $S_n$ . An  $S_n$  with set  $F$  is denoted as  $S_n^F$ . For instance, a star graph  $S_5$  with set  $F = \{52341, 43152\}$ , or  $S_5^{\{52341, 43152\}}$ , is shown in Fig. 1.

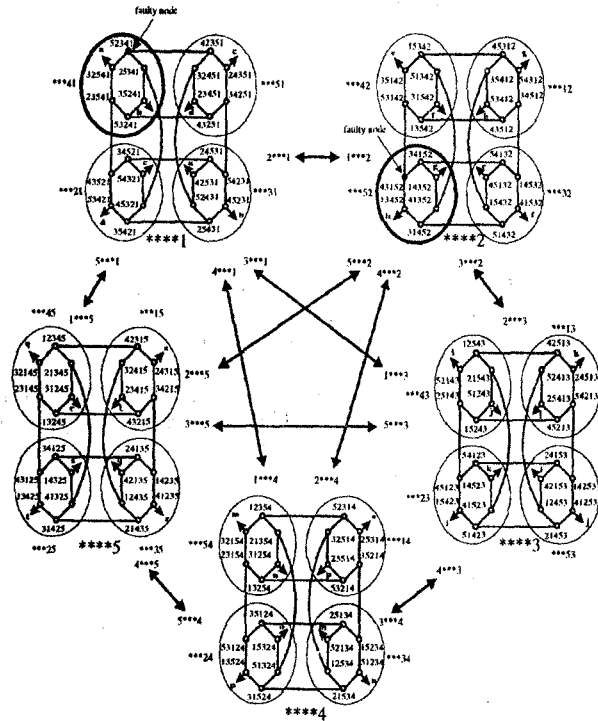


Figure 1: A star graph  $S_5$  with set  $F = \{52341, 43152\}$ .

Each  $S_n$  contains  $n$  disjoint  $S_{n-1}$ 's. Let  $\Gamma = \{1, 2, \dots, n, *\}$ , where  $*$  denotes a *don't care* symbol. Every substar of  $S_n$  can be uniquely labeled by a string of symbols in  $\Gamma$  such that only repeated symbol is  $*$ . Formally, a  $k$ -dimensional substar,  $S_k$  or  $k$ -substar, is denoted as a string  $G = x_1x_2 \dots x_n$  and number of  $*$  symbols in string  $G$  is  $k$ , where  $x_1 = *$  and  $x_i \in \Gamma, 2 \leq i \leq n$ . The substar represented by  $G$  is a subgraph of  $S_n$  containing all vertices obtained from  $G$  by replacing each  $*$  with digits  $\{1, 2, \dots, n\}$ . These vertices are connected by the original links in  $S_n$ . For instance, the  $**3*1$  is a 3-dimensional substar and contains the set of nodes  $\{54321, 45321, 52341, 25341, 42351, 24351\}$ . Throughout this paper, a  $k$ -substar is said to be faulty if there exist at least one faulty node in the  $k$ -substar, where  $1 \leq k \leq n$ . Otherwise, the  $k$ -substar is said as fault-free. For example in Fig. 1, substars  $**41$  and  $**52$  are faulty substars.

### Definition 1: $j$ -split and $D$ -split

Let  $G = x_1x_2 \dots x_j \dots x_n$  be a  $k$ -substar with  $x_j = *$ . The  $j$ -split on  $G, 1 \leq j \leq n$ , is to partition  $G$  along  $j$ -dimension into  $k$  number of  $(k - 1)$ -substars, each obtained from  $G$  by replacing  $x_j$  with a legal non- $*$  symbol. Let  $D = \{d_1, d_2, \dots, d_m, 1\}, m \leq k$ , be a set of dimensions such that the  $x_{d_i} = *, i = 1..m$ . Then the  $D$ -split on  $G$  is to first apply a  $d_1$ -split on  $G$ , whose result is then applied a  $d_2$ -split, whose result is then applied a  $d_3$ -split, etc, until there is  $k(k-1) \dots (k-m+1)$  number of  $(k-m)$ -substars. The final result of the  $D$ -split on  $G$  is obtained by applying a 1-split on each of the  $(k-m)$ -substars.  $\square$

In above definition, if  $j = 1$  then the partitioning result does not remain substars, which is defined as a *virtual substar* in the following. An  $S_n$  can be decomposed into  $n(n-1) \dots (k+1)$  copies of  $k$ -substar after applying  $D'$ -split on  $S_n$ , where  $D' = \{d_1, d_2, \dots, d_i, \dots, d_{n-k}\}$ , for all  $d_i \neq 1, 1 \leq i \leq n-k$ , and  $k \leq n$ . Given a  $k$ -substar  $S_k$ , 1-split on  $S_k$  is to decompose  $S_k$  into  $k$  *virtual substars* represented as  $X_i, 1 \leq i \leq k$ . The *virtual substar*  $X_i = x_1x_2 \dots x_n$  is subgraph of  $S_k$  containing all vertices obtained from  $S_k$ , where  $x_1$  is a non- $*$  symbol and number of  $*$  symbol of  $X_i$  is  $k - 1$ . These vertices are connected as follows. Assume that  $d'_1 < d'_2 < d'_3, \dots, < d'_{k-1} \in \{1, 2, \dots, n\} - \{d_1, d_2, \dots, d_{n-k}\}$ , where  $x_{d'_1} = x_{d'_2} = \dots = x_{d'_{k-1}} = *$ . For any node  $x$  in  $X_i$ , there is a *virtual edge* joining  $x$  and  $g_{d'_j}(g_{d'_j}(g_{d'_j}(x)))$ ,  $2 \leq j \leq k-1$ , and this *virtual edge* is said the *virtual dimension*  $j$ . Each node in  $X_i$  is connected to  $k-2$  *virtual adjacent nodes* by  $k-2$  *virtual edges*. For a fixed *virtual dimension*  $j, 1 \leq j \leq k$ , each disjoint node of  $X_i$ , in parallel, performs the same permutation functions  $g_{d'_j}, g_{d'_j}$ , and  $g_{d'_j}$  simultaneously. Under the assumption of the bidirectional link, all paths from nodes of  $X_i, 1 \leq i \leq k$ , to one of its *virtual adjacent nodes* are edge-disjoint. Moreover, each node of  $X_i, 1 \leq i \leq k$ , sending data to one of its *virtual adjacent nodes* needs 3 time steps.

Consider any pair of *virtual substars*  $X_i$  and  $X_j$  obtained from 1-split on  $S_k$ , where  $i \neq j$ . Each node  $rx_2x_3 \dots x_n$  of  $X_i$  has a direct link to node  $sy_2y_3 \dots y_n$  of  $X_j$  along dimension  $u$  such that  $x_u = s$  and  $y_u = r$ , where  $r$  and  $s$  are non- $*$  symbols. That is, if we say the relation of direct link is represented by lines connecting *virtual substar*. Then, each *virtual substar*  $X_i$  of  $S_k$  are fully connected with each other  $X_j$  of  $S_k$ .

For instance, given an  $S_6$ , a 5-split on  $G$  is to partition  $G$  into six 5-substars  $*****1, *****2, *****3, *****4, *****5$ , and  $*****6$ . Consider substar  $*****56$  is one of substars after applying  $D'$ -split on  $G$ , where  $D' = \{5, 6\}$ . The 1-split on  $*****56$  is to decompose  $*****56$  into *virtual substars*  $1***56, 2***56, 3***56$ , and  $4***56$  as shown in Fig. 2. Therefore,  $d'_1 = 2, d'_2 = 3$ , and  $d'_3 = 4$ . Fig. 2 indicates that *virtual edges* joining  $x$  and  $g_2(g_3(g_2(x)))$  along *virtual dimension* 2, where  $x$  in *virtual substars*  $1***56, 2***56, 3***56$ , and  $4***56$ . Since performing permutation function, the number of nodes with differ-

ent color is always equal to 6, so all paths from  $x$  to  $g_2(g_3(g_2(x)))$  are edge-disjoint. Node 123456 of virtual substar 1\*\*\*56 has direct link to 213456, 321456, and 423156 of virtual substars 2\*\*\*56, 3\*\*\*56, and 4\*\*\*56, respectively. The other nodes of 1\*\*\*56 also have links connecting to nodes in 2\*\*\*56, 3\*\*\*56, and 4\*\*\*56. Therefore, the virtual substar 1\*\*\*56 are fully connected with 2\*\*\*56, 3\*\*\*56, 4\*\*\*56. Similarly, virtual substars 2\*\*\*56, 3\*\*\*56, and 4\*\*\*56 are fully connected with each other.

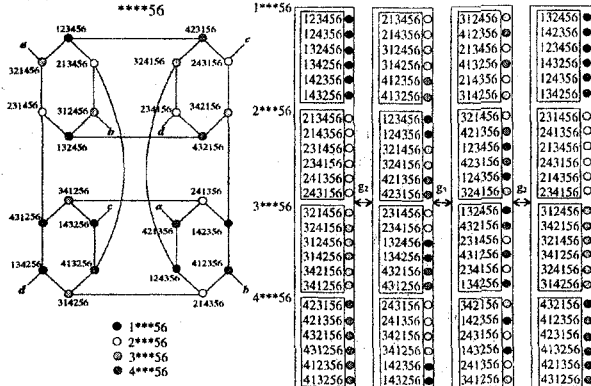


Figure 2: The virtual edges joining  $x$  and  $g_2(g_3(g_2(x)))$  along virtual dimension 2, where  $x$  in virtual substars 1\*\*\*56, 2\*\*\*56, 3\*\*\*56, and 4\*\*\*56.

### Definition 2 : Adjacent substars

Given any two  $k$ -substars  $G = x_1x_2 \dots x_j \dots x_n$  and  $H = y_1y_2 \dots y_i \dots y_n$  are said to be adjacent if and only if the labels of  $G$  and  $H$  differ in exactly one dimension, where  $1 < i \leq n$ .

For instance, 3-substar  $G = ***31$  are adjacent to  $H = ***41$ , but not adjacent to  $H' = ***13$ . In the following, we will describe the adjacent relation under applying the  $j$ -split operation. For given two adjacent  $k$ -substars  $G = x_1x_2 \dots x_j \dots x_n$  and  $H = y_1y_2 \dots y_j \dots y_n$  such that  $x_j = y_j = *$ . If we apply the  $j$ -split on  $G$  and  $H$ , we will obtain  $k$  substars (dimension  $k - 1$ ) from each of  $G$  and  $H$ . One can easily see that all  $k$  substars in  $G$  are adjacent to each other, and so are those in  $H$ . If the adjacent relations are represented by lines connecting substars, each  $(k - 1)$ -substars of  $G$  (or in  $H$ ) are fully connected with each other substars of  $G$  (or in  $H$ ). Furthermore, among these substars,  $k - 1$  substars in  $G$  are adjacent to  $k - 1$  substars in  $H$  in a one-to-one manner. For example, if we apply 5-split on  $G = S_5$  as shown in Fig. 1, \*\*\*\*1 are fully connected with \*\*\*\*2, \*\*\*\*3, \*\*\*\*4, and \*\*\*\*5. If  $G = ****3$ ,  $H = ****4$  and we apply 4-split on \*\*\*\*3 and \*\*\*\*4, we can see that each substar of \*\*\*13, \*\*\*23, \*\*\*43, and \*\*\*53 is fully connected with each other. Similarly, substars

\*\*\*14, \*\*\*24, \*\*\*34, and \*\*\*54 are also fully connected. Moreover, there are 3 pairs of substars (\*\*\*13, \*\*\*\*14), (\*\*\*23, \*\*\*\*24), and (\*\*\*53, \*\*\*\*54) are adjacent. Given a sequence of  $k$ -substars  $[G_0, G_1, \dots, G_{t-1}]$ , a  $(k, t)$ -ring is defined here before constructing our reconfiguration scheme.

### Definition 3 : $(k, t)$ -ring

A sequence of  $k$ -substars  $[G_0, G_1, \dots, G_{t-1}]$  is denoted as an  $(k, t)$ -ring if substars  $G_i$  is adjacent to its neighboring  $G_{(i-1) \bmod t}$  and  $G_{(i+1) \bmod t}$  for any  $i = 0..t - 1$ .

Before defining our final reconfiguration, we give the following lemma.

**Lemma 1 :** Given a  $(k, t)$ -ring  $= [G_0, G_1, \dots, G_{t-1}]$ , a feasible  $(k - 1, kt)$ -ring can be constructed from the  $(k, t)$ -ring.

**Proof:** If the  $j$ -th symbol of all  $G_i$ ,  $0 \leq i \leq t - 1$ , in  $(k, t)$ -ring is  $*$ , each  $G_i$  is applied the  $j$ -split to obtain  $k$   $(k - 1)$ -substars. All the  $(k - 1)$ -substars are fully connected by the adjacent relation, and there are  $k - 1$  connections between  $G_i$  and  $G_{i-1}$  (and  $G_{i+1}$ ). It can be easily to derive a feasible  $(k - 1, kt)$ -ring by visiting all  $(k - 1)$ -substars of  $G_i$  and one of  $k - 1$  connections between  $G_i$  and  $G_{i-1}$  (and  $G_{i+1}$ ).

The 1-split is performed on each  $G_i$  of a  $(k, t)$ -ring  $= [G_0, G_1, \dots, G_i, \dots, G_{t-1}]$  to obtain a sequence of virtual substars  $[X_0, X_1, \dots, X_i, \dots, X_{h-1}]$ . As mentioned earlier, each  $X_i$  is a virtual substar. Based on Lemma 1, our final reconfiguration scheme, namely fault-free substar-ring  $R_s(k - 1, h)$ , is defined as follows.

### Definition 4 : Fault-free substar-ring $R_s(k - 1, h)$

Let  $R_s(k - 1, h) = [X_0, X_1, \dots, X_i, \dots, X_{h-1}]$  denote a feasible fault-free substar-ring, where  $[X_0, X_1, \dots, X_i, \dots, X_{h-1}]$  is a sequence of disjoint fault-free virtual substars. The  $R_s(k - 1, h)$  is constructed by each node in  $X_i$  connecting to a node in  $X_{(i-1) \bmod h}$  and  $X_{(i+1) \bmod h}$  with dilation 3 at most, for all  $0 \leq i \leq h - 1$ . Therefore, the  $R_s(k - 1, h)$  is

$$X_0 \longleftrightarrow X_1 \longleftrightarrow X_2 \longleftrightarrow \dots \longleftrightarrow X_{h-2} \longleftrightarrow X_{h-1} \longleftrightarrow X_0.$$

If the connection  $X_{h-1} \longleftrightarrow X_0$  does not exist, then a fault-free substar-chain, denoted by  $C_s(k - 1, h)$ , is constructed. Obviously, a fault-free substar-ring  $R_s(k - 1, h)$  can be treated as a fault-free substar-chain  $C_s(k - 1, h)$  with the same size of virtual substars. The processor utilization of  $R_s(k - 1, h)$  and  $C_s(k - 1, h)$  are  $(k - 1) \times h$ . The diameter of  $R_s(k - 1, h)$  and  $C_s(k - 1, h)$  are at most  $\lfloor \frac{9}{2}(k - 2) \rfloor + \lfloor \frac{3h}{2} \rfloor$  and  $\lfloor \frac{9}{2}(k - 2) \rfloor + 3h$ , where  $h = n(n - 1) \dots (k - 1)$ . In this paper, we will only focus our attention on constructing a feasible fault-free substar-ring  $R_s(n - 3, h)$  in a faulty star graph.

### III. IDENTIFYING ALGORITHM

#### A. Construction of $R_s(n-3, h)$ with $f \leq n-3$ Faults

In the following, we describe a centralized algorithm to identify the maximal fault-free substar-ring  $R_s(n-3, h)$  (IMSR) for tolerating  $n-3$  faults in a faulty  $n$ -star graph. Besides, we also state how to apply ASCEND/DESCEND algorithms on the  $R_s(n-3, h)$ .

The IMSR algorithm is divided into three steps. First, we recognize all maximal fault-free  $S_{n-2}$  substars from an  $S_n^F$ . Second, a  $(n-2, t)$ -ring is constructed from the  $S_{n-2}$  substars, where  $n(n-1) - (n-3) \leq t \leq n(n-1) - 1$ . Third, the maximal fault-free substar-ring  $R_s(n-3, h)$ ,  $h \leq (n-2)t$ , is constructed by applying 1-split on each  $S_{n-2}$  substars of  $(n-2, t)$ -ring. The detail steps of IMSR algorithm are described in the followings.

In the beginning, we apply  $D$ -split on  $S_n^F$  to obtain  $n(n-1)$   $S_{n-2}$  substars based on the best selection of set  $D$ . If possible, all faulty nodes may be located in one  $S_{n-2}$  under a best selection of a set  $D$ . Then there are at most  $n^2 - n - 1$  fault-free  $S_{n-2}$  substars can be used. Finding the best set  $D$  for recognizing the maximal number of  $S_{n-2}$  substars is described as follows. Given a set of faulty nodes  $F$ ,  $f = |F|$ , in an  $n$ -star. Consider a node or substar  $x = x_1 x_2 \cdots x_n$ ,  $x_i \in \{*, 1, 2, \dots, n\}$  and  $i = 1..n$ , an extraction function is defined by  $e_i(x_1 x_2 \cdots x_i \cdots x_n) = x_i$ . A predicate function [2] is defined by

$$P(x) = \begin{cases} 1 & \text{if } x = \text{TRUE} \\ 0 & \text{if } x = \text{FALSE} \end{cases}$$

Let  $t_i^d$  be the occurrences of  $e_d(y_j) = i$  under a fixed value  $d$ ,  $1 \leq i \leq n$  and  $y_j \in F$ . That is,  $t_i^d = \sum_{j=1}^f P(e_d(y_j) = i)$ , where " $e_d(y_j) = i$ " is a boolean expression. Let  $m_d$  denote by  $\max_{i=1}^n (t_i^d)$ . The best set  $D$  is obtained by finding dimensions  $k$  and  $k'$  such that  $m_k$  and  $m_{k'}$  are the first and second largest values among  $m_d$ , where  $d = 1..n$ . For example, assume a faulty star  $S_5$  with  $F = \{12435, 32451, 52134\}$ , since  $e_2(12435) = e_2(32451) = e_2(52134) = 2$ , we have  $t_2^2 = 3$  and  $t_1^2 = t_3^2 = t_4^2 = t_5^2 = 0$ . Therefore,  $m_2 = \max_{i=1}^5 (t_i^2) = 3$ . Similarly, we can also obtain  $m_1 = 1$ ,  $m_3 = 2$ ,  $m_4 = 2$ , and  $m_5 = 1$ . Thus set  $D$  is  $\{2, 3\}$  or  $\{2, 4\}$  since both  $m_3$  and  $m_4$  are equal to 2. If we choose  $D = \{2, 3\}$ , the minimal number of faulty substars is 2. That is,  $*24**$  and  $*21**$  are faulty substars and  $*12**$ ,  $*13**$ ,  $*14**$ ,  $*15**$ ,  $*23**$ ,  $*25**$ ,  $*31**$ ,  $*32**$ ,  $*34**$ ,  $*35**$ ,  $*41**$ ,  $*42**$ ,  $*43**$ , and  $*45**$  are fault-free substars.

The next step is to construct a  $(n-2, t)$ -ring from  $S_n^F$  under set  $D = \{k, k'\}$ . Intuitively, the  $k$ -split on faulty  $S_n$  is to partition the  $S_n^F$  into  $n$   $S_{n-1}$  substars so that we can construct a  $(n-1, n)$ -ring. We then apply  $k'$ -split on all  $S_{n-1}$  substars of the  $(n-1, n)$ -ring to obtain  $n(n-1)$   $S_{n-2}$  substars. In our scheme, we withdraw all faulty substars from  $n(n-1)$   $S_{n-2}$  substars. Therefore, a  $(n-2, t)$ -ring is constructed from all non-faulty  $S_{n-2}$

substars, where  $t \leq n(n-1) - f$ .

**Lemma 2 :** Given a  $(n-1, n)$ -ring, if  $f \leq n-3$ , it is possible to construct a  $(n-2, t)$ -ring from the  $(n-1, n)$ -ring, where  $n^2 - 2n + 3 \leq t \leq n^2 - n - 1$ .

**Proof:** Given a set  $D = \{k, k'\}$ , we apply the  $D$ -split on the  $S_n^F$ . A  $(n-1, n)$ -ring is first obtained by applying the  $k$ -split on the  $S_n^F$ . If the  $k'$ -th symbol of  $G_i$ ,  $0 \leq i \leq n-1$ , in  $(n-1, n)$ -ring is  $*$ , each  $G_i$  is applied the  $k'$ -split to obtain  $n-1$   $S_{n-2}$  substars. As mentioned earlier, all the  $S_{n-2}$  substars are fully connected (by the adjacent relation), and there are  $n-2$  connections between  $G_i$  and  $G_{i-1}$  and  $G_{i+1}$ . Since there are  $n-3$   $S_{n-2}$  substars of  $G_i$  are faulty at most, there exists at least one connection between  $G_{i-1}$  and  $G_{i+1}$ . By the similar reason of Lemma 1, it is trivial to derive a feasible  $(n-2, t)$ -ring by visiting all non-faulty substars of  $G_i$  even when there are at most  $(n-3)$  faulty substars in  $G_i$ , where  $n(n-1) - (n-3) \leq t \leq n(n-1) - 1$ .  $\square$

Therefore, a  $(n-2, t)$ -ring, is constructed, where  $n^2 - 2n + 3 \leq t \leq n^2 - n - 1$ . There are  $n(n-1)$   $S_{n-2}$  substars and  $n-3$  of them are faulty at most. In the case, the  $(n-2, t)$ -ring uses  $n(n-1) - (n-3)$   $S_{n-2}$  substars if  $f \leq n-3$ , therefore the PUR is  $\frac{n^2 - 2n + 3}{n^2 - n}$  and the diameter is  $\lfloor \frac{9}{2}(n-4) \rfloor + \lfloor \frac{3n(n-1)(n-3)}{2} \rfloor$ . For example, if set  $D = \{4, 5\}$ , then  $***1 \leftrightarrow ***2 \leftrightarrow ***4 \leftrightarrow ***3 \leftrightarrow ****5 \leftrightarrow ****1$  is a  $(4, 5)$ -ring as indicated in Fig. 3. Note that substars  $***41$  and  $***52$  are faulty. Therefore, a  $(3, 18)$ -ring is shown in Fig. 3.

Given a  $(n-2, t)$ -ring  $= [G_0, G_1, \dots, G_{t-1}]$ , all  $(n-2)!$  nodes of  $G_i$  exchange the contents with its corresponding node of adjacent substars  $G_{(i-1) \bmod t}$  or  $G_{(i+1) \bmod t}$ . Using GROUP-COPY procedure [20], the task is achieved by 3 time steps if there is no fault occurring. But if there exists faults, GROUP-COPY procedure can't work, so the task cost  $O(n-2)$  time by only using  $(n-3)!$  direct links. A scheme to reduce the communication cost is proposed here. Our scheme is to apply 1-split on each  $S_{n-2}$  substar  $G_i$  of a  $(n-2, t)$ -ring to obtain virtual substars  $X_j$ , where  $0 \leq j \leq (n-2)t$ . Each  $X_j$  is worked as a processing unit. These  $X_j$  can finally construct a ring with 3-dilation links as shown in Theorem 1.

**Theorem 1 :** Given a  $(n-2, t)$ -ring  $= [G_0, G_1, \dots, G_{t-1}]$ , it is possible to construct a  $R_s(n-3, h) = [X_0, X_1, \dots, X_j, \dots, X_{h-1}]$  such that each pair of neighboring  $X_j$  and  $X_{j+1}$  is connected at most 3-dilation links, where  $h = (n-2)t$ .

**Proof:** First, if  $X_j$  and  $X_{j+1}$  are located in same  $G_l$  of the  $(n-2, t)$ -ring,  $1 \leq l \leq t$ ,  $X_j$  directly links to  $X_{j+1}$ . Second, if  $X_j$  and  $X_{j+1}$  are respectively located in neighboring  $G_l$  and  $G_{l+1}$  of the  $(n-2, t)$ -ring, there exists a pair of  $X'$  (in  $G_l$ ) and  $X''$  (in  $G_{l+1}$ ) such that  $X'$  direct links to  $X''$ . The 3-dilation links are  $X_j \leftrightarrow X' \leftrightarrow X'' \leftrightarrow X_{j+1}$ . Specially, if  $X'' = X_{j+1}$ , then links between  $X_j$  and  $X_{j+1}$  are only 2-dilation links. As a result, each pair

of neighboring  $X_j$  and  $X_{j+1}$  of  $R_s(n-3, h)$  is connected at most 3-dilation links.  $\square$

Recall above example, Fig. 3 shows a feasible  $R_s(2, 54) = [5^{**21}, 4^{**21}, 3^{**21}, 2^{**51}, 4^{**51}, 3^{**51}, \dots, 1^{**35}, 4^{**35}, 2^{**35}, 3^{**25}, 4^{**25}, 1^{**25}]$ . Note that, *virtual substars*  $3^{**12}$  and  $2^{**14}$  are respectively located in  $^{***}12$  and  $^{***}14$ , edges between  $3^{**12}$  and  $2^{**14}$  are  $3^{**12} \leftrightarrow 4^{**12} \leftrightarrow 2^{**14}$ .

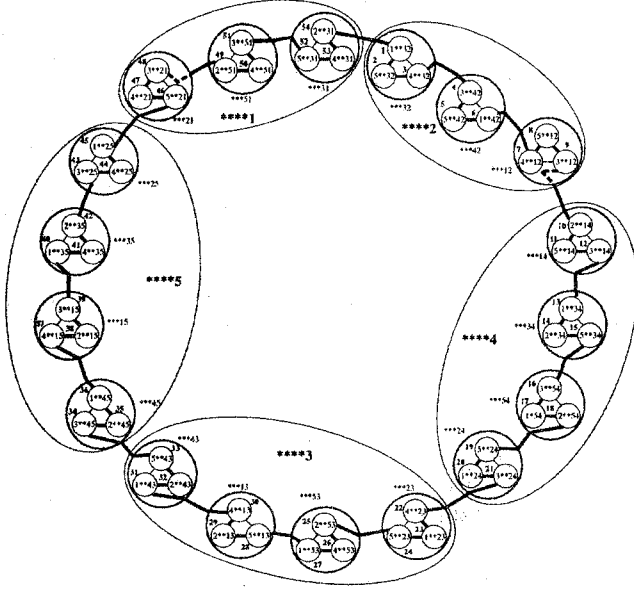


Figure 3: A maximal fault-free substar-ring  $R_s(2, 54)$ .

The IMSR algorithm is outlined as follows.

**Algorithm:** Identifying maximal fault-free substar-ring (IMSR)

**Input:** An  $S_n$  with faulty nodes set  $F$ , where  $1 \leq f \leq n-3$ .

**Output:** Substars sequence  $[X_0, X_1, \dots, X_{h-1}]$  is obtained, where  $(n-2)(n^2-2n+3) \leq h \leq (n-2)(n^2-n-1)$ .

**Step 1:** Find the best set  $D = \{k, k'\}$ . The maximal number of fault-free substars is obtained by partitioning  $S_n^F$  into disjoint  $S_{n-2}$  substars along dimensions  $k$  and  $k'$ .

**Step 2:** Identify a  $(n-2, t)$ -ring based on Lemma 2 among all fault-free  $S_{n-2}$  substars, where  $n^2 - 2n + 3 \leq t \leq n^2 - n - 1$ .

**Step 3:** Construct the maximal fault-free substar-ring  $R_s(n-3, h)$  by Theorem 1, where  $(n-2)(n^2 - 2n + 3) \leq h \leq (n-2)(n^2 - n - 1)$ .

The total time cost  $T_{IMSR}$  of IMSR algorithm is analyzed as follows. In step 1, time cost  $O(n^2 f)$  is needed to check all addresses of faulty nodes to determine the value of set  $D$ . When  $f = n-3$ , the time cost is  $O(n^3)$ . Step 2 only needs time cost  $O(n)$  to construct the  $(n-2, t)$ -ring. In step 3, time cost  $O(n^3)$  is needed to split each  $S_{n-2}$  of  $(n-2, t)$ -ring to obtain  $R_s(n-3, h)$ , where  $h \leq (n-2)t$ . Consequently, the total time cost of  $T_{IMSR}$  can be measured by the following equation.

$$T_{IMSR} = O(n^3) + O(n) + O(n^3) = O(n^3)$$

Therefore, our algorithm is an effective algorithm.

## B. Construction of $R_s(n-3, h)$ with $f > n-3$ Faults

We now describe how to construct  $R_s(n-3, h)$  when  $n-3 < f < n^2 - n$ . It is known that an  $N$ -node ring can be one-to-one embedded with dilation 3 in any connected  $N$ -node network [4]. Given a  $S_n^F$  and its best set  $D = \{k, k'\}$ . Initially, we construct a tree, namely *substar-tree*  $T$ , among  $n$   $S_{n-1}$ 's, which are obtained by performing  $k$ -split on  $S_n^F$ . Each node of *substar-tree*  $T$  is an  $S_{n-1}$ . Then all possible fault-free *virtual substars* can be obtained by performing  $k'$ -split and 1-split operations on each node of *substar-tree*  $T$ . These fault-free *virtual substars* can still form a connected network. Therefore, a ring of the *virtual substars* with 3-dilation links is obtained. The ring is denoted as  $R_s(n-3, h)$ . In the following, we describe the modified IMSR' algorithm to construct *substar-tree*  $T$  and then to obtain the final  $R_s(n-3, h)$ .

The detail algorithm of modified IMSR' algorithm is described as follows. We apply  $k$ -split on  $S_n^F$  to produce  $n$   $S_{n-1}$ 's and collect them into set  $\Psi$ , where  $D = \{k, k'\}$ . The *substar-tree*  $T$  is constructed based on set  $\Psi$ . As stated earlier, each node of *substar-tree*  $T$  is an  $S_{n-1}$  substar. The total nodes of *substar-tree*  $T$  is at most  $n$ . Before describe how to construct *substar-tree*  $T$ , we define a function  $AD(G, G')$  to represent the adjacent relation of a pair of  $S_{n-1}$ 's  $G$  and  $G'$ , where  $G$  and  $G'$  belong to set  $\Psi$ . The  $k'$ -split on  $G$  and  $G'$  is to decompose  $G$  and  $G'$  into  $2(n-1)$   $S_{n-2}$ 's, each of them is fault-free or not. Let function  $AD(G, G')$  denote the number of pairs of adjacent fault-free  $S_{n-2}$  substars  $x$  and  $y$ , where  $x$  and  $y$  are located in  $G$  and  $G'$ , respectively. If  $AD(G, G') > 0$ , there exists at least one pair of adjacent fault-free  $S_{n-2}$ 's between  $G$  and  $G'$ . Otherwise, no fault-free  $S_{n-2}$ 's are adjacent if  $AD(G, G') = 0$ . For example, if set  $D = \{4, 5\}$ , substars  $^{***}41$ ,  $^{***}51$ , and  $^{***}52$  are faulty substars as shown in Fig. 4, there is no fault-free adjacent substar  $S_3$ 's between  $^{***}1$  and  $^{***}2$ , so  $AD(^{***}1, ^{***}2) = 0$ . Equation  $AD(^{***}1, ^{***}3) = 1$  holds because  $^{***}21$  and  $^{***}23$  are the only pair of fault-free adjacent substars. Similarly, equations  $AD(^{***}1, ^{***}4) = 2$ ,  $AD(^{***}1, ^{***}5) = 2$ , and  $AD(^{***}3, ^{***}2) = 2$  are obtained.

Continually, *substar-tree*  $T$  is constructed as follows. First, root of tree  $T$  is a substar selected from set  $\Psi$  randomly. By using the Breadth-First-Searching method, we can expand branches of *substar-tree*  $T$  as follows.

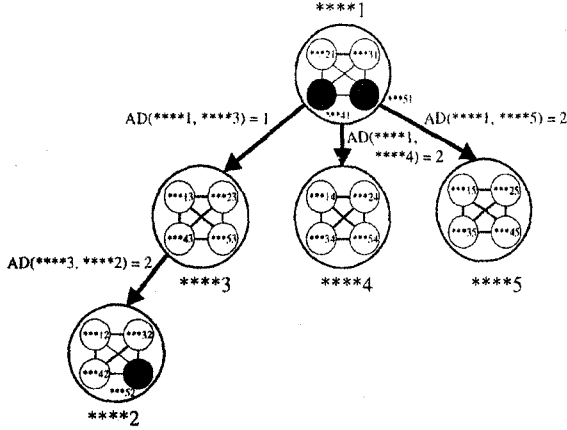


Figure 4: Constructing a *substar-tree*  $T$ .

Branches of tree  $T$  represent the possible connecting substars. Each node  $u$  of *substar-tree*  $T$  probes each of remaining substar  $v$  from set  $\Psi$ . If the condition  $AD(u, v) > 0$  exists, then node  $u$  connects to the substar  $v$  and eliminate  $v$  from set  $\Psi$ . Repeatedly performing above probing operations until set  $\Psi$  is empty or no connecting substar can be further found, *substar-tree*  $T$  is constructed. Since there exists at least one branch of each node of *substar-tree*  $T$ , so *substar-tree*  $T$  is a connected network. The 1-split is performed on each  $S_{n-2}$  substar of *substar-tree*  $T$  to obtain all possible *virtual substars*. Each *virtual substars* is treated as a processing unit, and these *virtual substars* still form a connected network. Therefore, a maximal fault-free substar-ring  $R_s(n-3, h)$  with 3-dilation links can be obtained [4], where  $h \leq n^3 - 4n^2 + n + 6$ . Recall above example, let root of *substar-tree*  $T$  be \*\*\*\*1, branches of root are \*\*\*\*3, \*\*\*\*4, \*\*\*\*5, and branch of \*\*\*\*3 is \*\*\*\*2. The *substar-tree*  $T$  is shown in Fig. 4. After applying 1-split on all  $S_3$ 's of *substar-tree*  $T$ , a  $R_s(2, 36)$  with 3-dilation links is obtained.

Finally, we analyze the total time cost  $T_{IMSR'}$  of modified IMSR algorithm. In step 1 of identifying all  $S_{n-2}$ , time cost  $O(n^2f)$  is needed. When  $f = n^2 - n - 1$ , the time cost is  $O(n^4)$ . During constructing *substar-tree*  $T$ , there are  $O(n^2)$  times to perform AD operation each one needs  $O(n^2)$  time. It takes  $O(n^4)$  to construct *substar-tree*  $T$ . Time cost  $O(n^3)$  is needed to split  $n(n-1)$   $S_{n-2}$ 's into  $n(n-1)(n-2)$  *virtual substars*. Consequently, the total time complexity of  $T_{IMSR'}$  can be measured by the following equation.

$$T_{IMSR'} = O(n^4) + O(n^4) + O(n^3) = O(n^4)$$

Therefore, our algorithm keeps an effective algorithm even when  $f < n^2 - n - 1$ .

### C. Application Algorithms on the $R_s(n-3, h)$

For demonstrate the application capability, we indicate

how to execute sorting operation on  $R_s(n-3, h) = [X_0, X_1, \dots, X_j, \dots, X_{h-1}]$ . First, an efficient sorting algorithm on star graph [17] can be performed on each *virtual substar*  $X_j$  of  $R_s(n-3, h)$  such that unsorted elements on each  $X_j$  are sorted. Second, if each *virtual substar*  $X_j$  of  $R_s(n-3, h)$  is worked as a processing unit, the Odd-Even Transposition Sort algorithm [1] can be performed on ring of these processing units. After performing above operations, in final, data elements in all *virtual substars* will be sorted. The detail algorithm can refer the similar sorting operations on the maximal fault-free subcube-ring in faulty hypercubes [23]. Furthermore, using the similar matrix-multiplication operations on maximal fault-free subcube-ring [23], we can also perform the matrix-multiplication algorithm on  $R_s(n-3, h)$ . Similarly, lots of scientific algorithms on star graph [8] [20] can be tailored onto our  $R_s(n-3, h)$ .

## IV. PERFORMANCE ANALYSIS

In this section, we analyze the distributed percentage of processor utilization of the maximal fault-free substar-ring under two cases of number of faulty processors are assumed. First, if  $f$  is less than  $n-2$ , the PUR is at most  $\frac{n^2-2n+3}{n^2-n}$ . Second, we will depict the PUR of our reconfiguration scheme even when  $n-2 \leq f \leq n^2-n-1$ .

The percentage of processor utilization of  $R_s(n-3, h)$  is analyzed as follows. In our simulation, the addresses of faulty processors are randomly generated on each of 10000 simulations for fixed  $n$  and  $f$ . For illustrating the capability of fault tolerance, we consider the worst case to simulate the PUR. An  $S_n$  is partitioned into  $n(n-1)$   $S_{n-2}$  by step 1 of IMSR algorithm. Here, we denote the number of faulty  $S_{n-2}$  by  $r$ . The factor of the value of  $r$  presents the degree of occurring faults. That is, the larger value  $r$  is, the more number of occurring faults will be. In the following, we will use the factor of value of  $r$  to analyze the PUR. It is known that if percentage of processor utilization of  $R_s(n-3, h)$  is larger than 50%, the slowdown factor of computation has more opportunity to reduce to be smaller than 2. In the case of  $f \leq n-3$ , the PUR is at most  $\frac{n^2-2n+3}{n^2-n}$  and is always larger than 50%. All possible maximal fault-free substar-ring  $R_s(n-3, h)$  and percentage distribution of processor utilization in a faulty  $S_5$ , where  $1 \leq r \leq 5^2-5-1 (= 19)$  and  $3 \leq h \leq 57$ , are shown in Table I. For instance, when  $n = 5$  and  $r = 1, 2$ , and  $3$ , 100% cases of  $S_5$  can be identified into  $R_s(2, 57)$  with 95% processor utilization, 100% cases are identified into  $R_s(2, 54)$  with 90% processor utilization, and 100% cases are identified into  $R_s(2, 51)$  with 85% processor utilization, respectively. When  $n = 5$  and  $r = 4$ , 99.57% cases of  $S_5$  are identified into  $R_s(2, 48)$  with 80% processor utilization and 0.43% cases are identified into  $R_s(2, 45)$  with 75% processor utilization. This indicates that the smaller the value of  $r$  is, the maximal fault-free substar-ring with high PUR generally be determined. As shown in Table I, all more than 73% cases to exploit

Table 1: Percentage distribution of processor utilization of maximal fault-free substar-ring  $R_s(2, h)$  in a  $S_5$ , where the number of faulty 3-substars is from 1 to 19 and  $3 \leq h \leq 57$ .

PUR	h	The number of faulty 3-substars																		
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
5%	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10%	6	0	0	0	0	0	0	0	0	3.21	11.45	26.40	45.58	65.38	74.96	66.72	43.31	15.85	0	0
15%	9	0	0	0	0	24	65	2.73	7.24	15.05	23.79	33.35	37.03	32.85	24.55	15.03	6.14	1.08	0	0
20%	12	0	0	0	0	0	0	0	0	65	1.23	2.63	4.67	5.61	5.18	3.51	2.14	.64	0	0
25%	15	0	0	0	0	0	0	0	0	4	1.9	4.35	8.36	9.47	8.7	6.65	4.19	1.54	0	0
30%	18	0	0	0	0	.37	.6	2.12	5.69	10.29	11.62	10.20	6.84	2.37	0	0	0	0	0	0
35%	21	0	0	0	0	0	0	1.10	3.31	8.07	11.00	12.12	7.82	2.96	0	0	0	0	0	0
40%	24	0	0	0	0	0	1.58	2.12	5.26	8.19	12.39	9.61	4.24	0	0	0	0	0	0	0
45%	27	0	0	0	0	.40	.12	1.28	5.57	12.17	12.89	7.71	0	0	0	0	0	0	0	0
50%	30	0	0	0	0	.06	6.21	10.02	16.29	15.35	0	0	0	0	0	0	0	0	0	0
55%	33	0	0	0	0	0	4.70	4.96	15.14	28.96	0	0	0	0	0	0	0	0	0	0
60%	36	0	0	0	0	1.52	.87	2.88	48.79	0	0	0	0	0	0	0	0	0	0	0
65%	39	0	0	0	0	0	0.45	5.44	70.73	0	0	0	0	0	0	0	0	0	0	0
70%	42	0	0	0	0	0	2.02	46.21	0	0	0	0	0	0	0	0	0	0	0	0
75%	45	0	0	0	0	43	95.32	0	0	0	0	0	0	0	0	0	0	0	0	0
80%	48	0	0	0	99.57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
85%	51	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
90%	54	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
95%	57	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

the more than 50% processor utilization in an faulty  $S_5$  when  $1 \leq r \leq 8$ . This also indicates that the percentage of processor utilization of maximal fault-free substar-ring  $R_s(n-3, h)$  is always larger than 50% when  $r < n(n-1)/2$ .

Continually, we discuss the average PUR as follows. The average PUR is defined by the sum of percentage of processor utilization of each  $R_s(n-3, h)$  \* PUR of the  $R_s(n-3, h)$ . For instance in Table I, when  $n = 5$  and  $r = 4$ , 99.57% of  $R_s(2, 48)$  with PUR = 80% and 0.43% of  $R_s(2, 45)$  with PUR = 75% are identified, so the average PUR is 99.57% \* 80% + 0.43% \* 75% = 79.9785%. In our simulation, we estimate the average PUR under the cases of  $5 \leq n \leq 12$ . The simulation results of the average PUR with different value of  $r$  are depicted in Fig. 5. The average PUR with value of  $r$  being larger than  $n(n-1)/2$  are always larger than 50% as depicted in Fig. 5. The average PUR is inverse proportional to the value of  $r$ , i.e., the larger value of  $r$  is, the higher average PUR will be. For instance, when the number of faulty  $S_{n-2}$  substars are  $n(n-1)/10$ ,  $2n(n-1)/10$ , and  $3n(n-1)/10$ , the average PUR are about 90%, 80%, and 70%, respectively. Consequently, two factors, the value of  $f$  and locations of faulty nodes, mainly effect number of faulty  $S_{n-2}$  substars. The smaller the value of  $f$  is, the smaller number of faulty  $S_{n-2}$  substars will be. Therefore, the smaller number of faulty  $S_{n-2}$  substars is, the high average PUR will be obtained.

Note that, when the number of faulty  $S_{n-2}$  are smaller than  $n(n-1)/2$ , more than 50% of average PUR are obtained by our simulation results. This is indicated that our reconfiguration scheme has reasonable average PUR. This also concludes that our method is a truly effective reconfiguration scheme with high processor utilization.

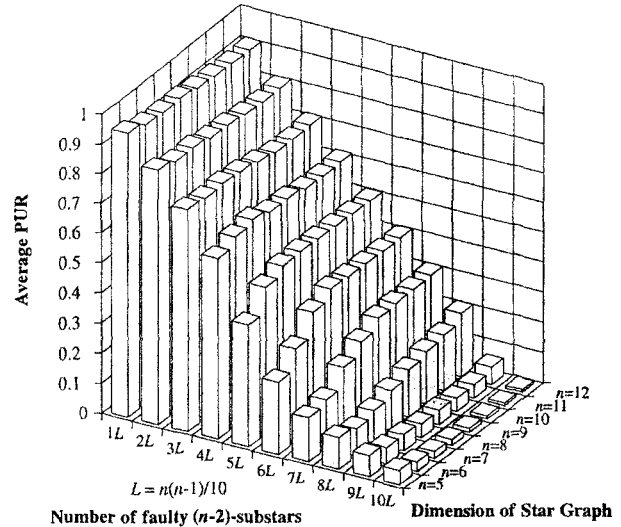


Figure 5: The average PUR of  $R_s(n-3, h)$  of  $S_n$ , for  $5 \leq n \leq 12$ .

## V. CONCLUSIONS

In this paper, we propose a reconfiguration scheme to identify the maximal fault-free substar-ring for tolerating faults in faulty  $n$ -dimensional star graph. The fault-free substar-ring is connected by a ring of fault-free *virtual substars* with dilation 3. This is the first result to propose a reconfiguration scheme in the faulty star graph. Our proposed scheme can tolerate  $n-3$  faults such that the processor utilization is  $\frac{n^2-2n+3}{n^2-n}$  and the diameter is  $\lfloor \frac{3}{2}(n-4) \rfloor + \lfloor \frac{3n(n-1)(n-3)}{2} \rfloor$ . This is a near optimal result since the maximal fault-free substar-ring is constructed by using all of the possible fault-free  $(n-2)$ -substars. For demonstrate application capability, we describe how to apply sorting algorithm onto our reconfiguration scheme. Moreover, our reconfiguration scheme can work when faults exceeds  $n-3$ . We also simulate the algorithm for showing the reconfiguration scheme with high processor utilization.

In order to preserve low diameter and obtain better processor utilization, identifying the maximal fault-free substar-ring  $R_s(k-1, h)$  is the main objective of this study. Determining the maximal fault-free substar-ring  $R_s(k-1, h)$  is controlled by what values of  $k$  and  $h$  being are the best selection. It is possible to construct a  $R_s(k-1, h)$ ,  $k \leq n-2$ , to obtain higher diameter and processor utilization. Specially, if  $k-1 = 1$ , our scheme becomes simple problem of ring embedding on the faulty star graph. However, in this paper, we only focus on identifying the maximal fault-free substar-ring  $R_s(n-3, h)$  to keep lower diameter and obtain reasonable processor utilization.

## REFERENCES

- [1] S. G. Akl, *Parallel Sorting Algorithms*. Academic Press, Inc, 1985.
- [2] M. D. Davis and E. J. Weyuker, *Computability, Complexity, and Languages*, Academic Press, Inc, 1983.
- [3] R. P. Grimaldi, *Discrete and Combinatorial Mathematics: An Applied Introduction*, Addison-Wesley Publishing Company, 1985.
- [4] F. T. Leighton, *Introduction to Parallel Algorithms and Architecture: Array · Tree · Hypercube*, Morgan Kaufmann Publishers, 1992.
- [5] S. Akers, D. Harel, and B. Krishnamurthy, "The Star Graph: An Attractive Alternative to the n-Cube," *Proc. of Int. Conf. Parallel Processing 1987*, pp. 393-400.
- [6] S. B. Akers and B. Krishnamurthy, "A Group-Theoretic Model for Symmetric Interconnection Networks," *IEEE Transactions on Computers*, Vol. 38, No. 4, pp. 555-565, April 1989.
- [7] S. G. Akl, K. Qiu, and I. Stojmenovic, "Data communication and Computational Geometry on the Star and Pancake Interconnection Network," *Proc. of the Third IEEE Symposium on Parallel and Distributed Processing*, pp.415-422, Dec. 1991.
- [8] S. G. Akl, K. Qiu, and I. Stojmenovic, "Fundamental Algorithms for the Star and Pancake Interconnection Networks with Applications to Computational Geometry," *Networks*, Vol. 23, No. 4, pp. 215-225, July 1993.
- [9] N. Bagherzadeh, N. Nassif, and S. Latifi, "A Routing and Broadcasting scheme on Faulty Star Graphs," *IEEE Transactions on Computers*, Vol. 41, No. 11, pp. 1398-1403, Nov. 1993.
- [10] M. Y. Chan and S. J. Lee, "Distributed Fault-Tolerant Embeddings of Rings in Hypercubes," *Journal of Parallel and Distributing Computing*, Vol. 11, 1991, pp. 63-71.
- [11] H. L. Chen and N. F. Tzeng, "Quick Determination of Subcubes in a Faulty Hypercube," *Proc. of 1992 International Conference on Parallel Processing*, 1992, Vol. III, pp. 338-345.
- [12] Y. S. Chen and J. P. Sheu, "A Reliable Sorting Algorithm on Hypercube Multicomputers," *Journal of Parallel Algorithms and Applications*, Vol. 5, No. 2, pp. 165-186, 1995.
- [13] K. Day and A. Tripathi, "A Comparative Study of Topological Properties of Hypercubes and Star Graphs," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 5, No. 1, Jan. 1994, pp. 31-38.
- [14] J. S. Jwo, S. Lakshmirarahan, and S.K. Dhall, "Embedding of Cycles and Grids in Star Graphs," *Proc. of 2nd IEEE Symp. Parallel and Distributed Processing*, 1990, pp. 540-547.
- [15] S. Latifi, "Task Allocation in the Star Graph", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 5, No. 11, pp. 1220-1224, Nov. 1994.
- [16] J. Mišić, Z. Jovanović, "Communication Aspects of the Star Graph Interconnection Network," *IEEE Transactions on Parallel and Distributed systems*, Vol. 5, No. 7, 1994, pp. 678-687.
- [17] A. Menn and A. K. Somani, "An Efficient Sorting Algorithm for the Star Graph Interconnection Network," *Proc. of the 19th International Conference on Parallel Processing*, Pennsylvania, U. S. A., Vol. III, pp. 1-8, Aug. 1990.
- [18] D. Nassimi, "Parallel Algorithms for the Classes for  $\pm 2^b$  DESCEND and ASCEND Computations on a SIMD Hypercube," *IEEE Transactions on Parallel and Distributed systems*, Vol. 4, No. 12, 1993, pp. 1372-1381.
- [19] F. J. Provost and R. Melhem, "Distributed fault tolerant embedding of binary trees and rings in hypercubes," *Proc. of International Workshop in Defect and Fault tolerance in VLSI systems*, 1988.
- [20] K. Qiu, S. G. A. and H. Meijer, "On Some Properties and Algorithms for the Star and Pancake Interconnection Networks," *Journal of Parallel and Distributing Computing*, Vol. 22, 1994, pp. 16-25
- [21] K. Qiu, H. Meijer and S. Akl, "Decomposing a Star Graph into Disjoint Cycles," *Information Processing Letters*, 39, pp. 125-129, Aug. 1991.
- [22] J. P. Sheu, Y. S. Chen, and C. Y. Chang, "Fault-Tolerant Sorting Algorithm on Hypercube Multicomputers," *Journal of Parallel and Distributed Computing*, Vol. 16, No. 2, pp. 185-197, Oct. 1992.
- [23] J. P. Sheu and Y. S. Chen, "Tolerating Faults in Faulty Hypercubes Using Maximal Fault-Free Subcube-Ring," *Proc. of EURO-PAR'95, International Conference on Parallel Processing, Merging PARLE and CONPAR-VAPP*, Stockholm, Sweden, pp. 661-672, Aug. 1995.
- [24] S. Sur and P. K. Srimani, "A Fault Tolerant Routing Algorithm in Star Graph Interconnection Network," *Proc. of the 20th International Conference on Parallel Processing*, Pennsylvania, U. S. A., Vol III, pp. 267-270, 1991.
- [25] Y. C. Tseng, S. H. Chang, and J. P. Sheu, "Fault-Tolerant Ring Embedding in Star Graphs," will appear in *Proc. of the 10th International Parallel Processing Symposium*, 1996.