Strong Fault-Tolerance: Parallel Routing in Star Networks with Faults

EUNSEUK OH †       JIANER CHEN‡

Abstract

We study the strong fault-tolerance for the star networks. Let $G$ be a network in which all nodes have degree $d$. We say that $G$ is strongly fault-tolerant if it has the following property: let $G_f$ be a copy of $G$ with at most $d - 2$ faulty nodes removed, then for any pair of non-faulty nodes $u$ and $v$ in $G_f$, there is a container of width $t$ between $u$ and $v$, where $t$ is the minimum of degree of $u$ and degree of $v$ in $G_f$. We show that the star networks are strongly fault-tolerant and develop an algorithm that constructs a maximum width container of nearly optimal length in a star network with faults. Our algorithm requires no prior knowledge for the faulty nodes and runs in optimal time.

1 Introduction

Routing on large size networks with faults is an important issue in the study of computer interconnection networks. In this paper, we study a new measure for network fault tolerance: the strong fault tolerance. Following the conventions introduced in [14], for any two nodes $u$ and $v$ in a given network $G$, a container $C(u,v)$ is a set of node-disjoint paths between $u$ and $v$ in the network $G$. The width of the container $C(u,v)$ is the number of paths in $C(u,v)$, and the length of $C(u,v)$ is the length of the longest path in $C(u,v)$. Let $G_f$ be a copy of a network $G$ with a set $S_f$ of faulty nodes removed. Let $u$ and $v$ be two non-faulty nodes in $G_f$. Based on local information, we know the degrees $\text{deg}_f(u)$ and $\text{deg}_f(v)$ of $u$ and $v$ in $G_f$ and are interested in constructing a container of largest width between $u$ and $v$ in $G_f$. Obviously, the width of a container between $u$ and $v$ in $G_f$ cannot be larger than $\min\{\text{deg}_f(u), \text{deg}_f(v)\}$. We are interested in knowing the precise bound on the size of the faulty node set $S_f$ such that for any two non-faulty nodes $u$ and $v$ in $G_f$, there is a container $C(u,v)$ of width $\min\{\text{deg}_f(u), \text{deg}_f(v)\}$.

Clearly, if the network $G$ has all its nodes of degree $d$, then in general the number of faulty nodes in the set $S_f$ should not exceed $d - 2$ to ensure a container of width $\min\{\text{deg}_f(u), \text{deg}_f(v)\}$ between any two nodes $u$ and $v$ in $G_f$. This can be seen as follows. Let $u$ and $v$ be two nodes in $G$ whose distance is larger than 3. Pick any neighbor $u'$ of $u$ and remove the other $d - 1$ neighbors of $u'$. Note that no neighbor of $u'$ can be a neighbor of $v$ since the distance from $u$ to $v$ is at least 4. Let the resulting network be $G_f$. The degrees of the nodes $u$ and $v$ in $G_f$ are $d$. However, there are obviously no container of width $d$ in $G_f$ from $u$ to $v$ since one of the $d$ neighbors of $u$ in $G_f$, the node $u'$, leads to a “deadend”. This motivates the following definition:

**Definition** A network $G$ in which all nodes are of degree $d$ is strongly fault tolerant if for any
copy $G_f$ of $G$ with at most $d - 2$ faulty nodes removed, the following holds: for any pair $u$ and $v$ of nodes in $G_f$, there is a container $C(u, v)$ of width $t$ in $G_f$ connecting $u$ and $v$, where $t$ is the minimum of degree of $u$ and degree of $v$ in $G_f$.

Strong fault tolerance characterizes the property of parallel routing (i.e., constructing node-disjoint paths) in a network with faulty nodes. Since one of the motivations of network parallel routing is to provide alternative routing paths when failures occur, strong fault tolerance can also be regarded as the study of fault tolerance in networks with faults.

To authors’ knowledge, there has not been a systematic study on parallel routing on networks with faults. We will discuss this issue in the current paper in detail based on the star networks proposed in the literature [2]. The $n$-dimensional star network (or simply the $n$-star network) $S_n$ is an undirected graph consisting of $n!$ nodes labeled with the $n!$ permutations on symbols $\{1, 2, \ldots, n\}$. There is an edge between two nodes $u$ and $v$ in $S_n$ if and only if $v$ can be obtained from $u$ by exchanging the positions of the first symbol and another symbol in $u$. The $n$-star network has all its nodes of degree $n - 1$. The star networks have received considerable attention as an attractive alternative to the widely used hypercube network model, because of its rich structure, smaller diameter, lower degree, and symmetry properties [2, 23].

Parallel routing on star networks without faulty nodes has been studied in the literature. Sur and Srimani [23] demonstrated that a width $n - 1$ container can be constructed between any two nodes in $S_n$ in polynomial time. Dietzfelbinger, Madhavapeddy, and Sudborough [9] derived an improved algorithm that constructs a width $n - 1$ container of length bounded by 4 plus the diameter of $S_n$. The algorithm was further improved by Day and Tripathi [8] who developed an efficient algorithm that constructs a width $n - 1$ container between two given nodes $u$ and $v$ whose length is bounded by 4 plus the distance from $u$ to $v$ in $S_n$. More recently, Chen and Chen [6] developed an efficient algorithm that constructs a width $n - 1$ container of optimal length for any two given nodes in the $n$-star network $S_n$. The problem of constructing node-disjoint paths connecting a set of nodes in the star networks has also been studied [7, 13].

Fault tolerance on the star networks has been studied. The general fault tolerance properties of the star networks were first studied and analyzed in [1, 2, 3]. The problem of determining the diameter of a star network with faults was considered in [3, 11, 16, 22]. Algorithms for node-to-node routing in star networks with faults were developed in [4, 12]. Broadcasting algorithms in star networks with faults have been considered by a number of researchers [4, 11, 10, 17, 18]. A randomized algorithm, based on the Information Dispersal Algorithm(IDA) [20], for parallel routing in star networks with faults was proposed in [21].

In the current paper, we study the strong fault-tolerance of the star networks, and develop an efficient algorithm that constructs a container of the maximum width between any two non-faulty nodes in the $n$-star network $S_n$ with at most $n - 3$ faulty nodes. Let $T_n$ be a copy of $S_n$ with at most $n - 3$ faulty nodes removed. For any two non-faulty nodes $u$ and $v$ in $T_n$, our algorithm constructs, in time $O(n^2)$, a width $t$ container of length $d_f(u, v) + c$, where $t$ is the minimum of degree $u$ and degree of $v$ in $T_n$, $d_f(u, v)$ is the distance from the node $u$ to the node $v$ in $T_n$, and $c$ is a small constant. The time complexity of our algorithm is optimal, and our algorithm requires no prior knowledge of faulty nodes.
2 Preliminaries

A permutation \( u = (a_1 a_2 \cdots a_n) \) of the symbols 1, 2, \ldots, \( n \) can be given by a product of disjoint cycles \([5]\), which is called the cycle structure of the permutation. A cycle is nontrivial if it contains more than one symbol. Otherwise the cycle is trivial. The cycle containing the symbol 1 will be called the primary cycle. For example, the permutation (32541) has the cycle structure (351)(2)(4). The cycles can be interpreted as follows: the primary cycle (351) indicates that 3 is at 1’s position, 5 is at 3’s position, and 1 is at 5’s position. The trivial cycles (2) and (4) indicate that 2 and 4 are in their “correct” positions.

We define two groups of operations \( \rho_i \) and \( \sigma_i \) on permutations as follows. Given a permutation \( u \), for each position \( i \), \( 1 \leq i \leq n \), and for each symbol \( a \) in \{1, 2, \ldots, \( n \)\}, \( \rho_i(u) \) is the permutation obtained from \( u \) by exchanging the first symbol and \( i \)th symbol in \( u \), and \( \sigma_i(u) \) is the permutation obtained from \( u \) by exchanging the first symbol and the symbol \( a \) in \( u \).

Let us consider how these operations change the cycle structure of a permutation. Write \( u \) in its cycle structure

\[
  u = (a_{11} \cdots a_{1n_1} 1)(a_{21} \cdots a_{2n_2}) \cdots (a_{k1} \cdots a_{kn_k})
\]

If the \( i \)th symbol \( a \) of \( u \) is not in the primary cycle, then \( \rho_i(u) = \sigma_i(u) \) “merges” the cycle containing \( a \) into the primary cycle. More precisely, suppose that \( a = a_{1j} \) (note that each cycle can be cyclically permuted and the order of the cycles is irrelevant), then the permutation \( \rho_i(u) = \sigma_i(u) \) will have the cycle structure:

\[
  \rho_i(u) = \sigma_i(u) = (a_{12} \cdots a_{2n_2} a_{11} \cdots a_{1n_1} 1)(a_{31} \cdots a_{3n_3}) \cdots (a_{k1} \cdots a_{kn_k})
\]

If the \( i \)th symbol \( a \) is in the primary cycle, then \( \rho_i(u) = \sigma_i(u) \) “splits” the primary cycle into two cycles. More precisely, suppose that \( a = a_{1j} \), where \( 1 \leq j \leq n_1 + 1 \) (here we have let \( a_{1n_1+1} = 1 \)), then \( \rho_i(u) = \sigma_i(u) \) will have the following cycle structure:

\[
  \rho_i(u) = \sigma_i(u) = (a_{11} \cdots a_{1j-1})(a_{1j} \cdots a_{1n_1} 1)(a_{21} \cdots a_{2n_2}) \cdots (a_{k1} \cdots a_{kn_k})
\]

In particular, if \( a = a_{12} \), then we say that the operation \( \rho_i \) “deletes” the symbol \( a_{11} \) from the primary cycle.

Since the nodes in the \( n \)-star network \( S_n \) are labeled by the permutations on the symbols \{1, 2, \ldots, \( n \)\}, throughout this paper, we assume that each node in the \( n \)-star network \( S_n \) is given by its corresponding permutation. By the definition of the \( n \)-star network \( S_n \), each node \( u \) is adjacent to the \( n - 1 \) nodes \( \rho_i(u) \), \( 2 \leq i \leq n \). Equivalently, the \( n-1 \) neighbors of \( u \) are the \( n-1 \) permutations \( \sigma_i(u) \), where \( a \) is any symbol in \{1, 2, \ldots, \( n \)\} except the first symbol in \( u \). A path in \( S_n \) from a node \( u \) to a node \( v \) corresponds to a sequence of nodes obtained by applying the operations \( \rho_i \) or \( \sigma_i \), starting from the node \( u \) and ending at the node \( v \).

Denote by \( \varepsilon \) the node labeled by the identity permutation, \( \varepsilon = (12 \cdots n) \). Since the \( n \)-star network \( S_n \) is vertex-symmetric \([2]\), a set of containers from a node \( w \) to a node \( v \) can be mapped to a set of containers from a node \( u \) to \( \varepsilon \) in a straightforward way. Therefore, we will concentrate on the construction of a container from \( u \) to \( \varepsilon \) in \( S_n \).

Denote the distance from a node \( u \) to \( \varepsilon \) by \( \text{dist}(u) \). Let \( u \) have the cycle structure \( u = c_1 \cdots c_k e_1 \cdots e_m \), where \( c_i \) are nontrivial cycles and \( e_j \) are trivial cycles. If we further let \( l = \frac{n}{2} \),
Lemma 3.1 Let $T_n[i]$ be a copy of the substar $S_n[i]$ with $k_i \leq n - 3$ faulty nodes removed, $i \neq 1$. For a node $u$ in $T_n[i]$, a fault-free path $P$ from $u$ to $\rho_i(\varepsilon)$ in $T_n[i]$ can be constructed in time \[ \sum_{i=1}^k |c_i|, \] where $|c_i|$ denotes the number of symbols in the cycle $c_i$, then the distance $\text{dist}(u)$ from the node $u$ to the identity node $\varepsilon$ is given by the following formula [2].

$$\text{dist}(u) = \begin{cases} 
    l + k & \text{if the primary cycle is a trivial cycle} \\
    l + k - 2 & \text{if the primary cycle is a nontrivial cycle}
\end{cases}$$

Combining this formula with the above discussion on the effect of applying the operations $\rho_i$ and $\sigma_a$ on a permutation, we derive the following necessary and sufficient rules for tracing a shortest path from the node $u$ to the identity node $\varepsilon$ in the $n$-star network $S_n$.

**Shortest Path Rules**

**Rule 1.** If the primary cycle is a trivial cycle in $u$, then in the next node on any shortest path from $u$ to $\varepsilon$, a nontrivial cycle $c_i$ is merged into the primary cycle. This corresponds to applying the operation $\sigma_a$ on $u$ with $a \in c_i$.

**Rule 2.** If the primary cycle $c_1 = (a_{11}a_{12} \cdots a_{1n_1}a_{1n_1+1})$ is a nontrivial cycle in $u$, where $a_{1n_1+1} = 1$, then in the next node on any shortest path from $u$ to $\varepsilon$, either a nontrivial cycle $c_i \neq c_1$ is merged into the primary cycle (this corresponds to applying the operation $\sigma_a$ on $u$, where $a \in c_i$), or the symbol $a_{11}$ is deleted from the primary cycle $c_1$ (this corresponds to applying the operation $\sigma_{a_{12}}$ on $u$).

**Fact 2.1.** If an edge $[u, v]$ in $S_n$ does not lead to a shortest path from $u$ to $\varepsilon$, then $\text{dist}(v) = \text{dist}(u) + 1$. Consequently, let $P$ be a path from $u$ to $\varepsilon$ in which exactly $k$ edges do not follow the Shortest Path Rules, then the length of the path $P$ is equal to $\text{dist}(u) + 2k$.

Two simple procedures will be used in following a shortest path from a node $u$ to $\varepsilon$. The first is called the “Delete” procedure, written as $\rightarrow \cdots \rightarrow$, which repeatedly deletes the first symbol in the non-trivial primary cycle. The second one is called the “Merge-Delete” procedure [6], written as $\rightarrow M + D \rightarrow$, which works in two stages: first repeatedly merges in an arbitrary order each of the nontrivial cycles into the primary cycle, then repeatedly deletes the first symbol in the primary cycle. It is easy to verify that both the “Delete” procedure and the “Merge-Delete” procedure follow the Shortest Path Rules strictly.

For the $n$-star network $S_n$, let $S_n[i]$ be the set of nodes in which the symbol 1 is at the $i$th position. It is well-known [1] that the set $S_n[1]$ is an independent set (i.e., no two nodes in $S_n[1]$ are adjacent to each other), and the subgraph induced by the set $S_n[i]$ for $i \neq 1$ is an $(n-1)$-star network. Note that a node is in the substar $S_n[i]$, $i \neq 1$, if and only if the primary cycle of the node is of the form $(\cdots i1)$, and a node is in $S_n[1]$ if and only if the primary cycle of the node is a trivial cycle (1).

A nice property of the Delete procedure and Merge-Delete procedure is that if they start with a node $u$ in a substar $S_n[i]$, $i \neq 1$, then all nodes, possibly except the last one, on the constructed shortest path are also in the substar $S_n[i]$.

3 Bridging paths from a node to a substar

**Lemma 3.1** Let $T_n[i]$ be a copy of the substar $S_n[i]$ with $k_i \leq n - 3$ faulty nodes removed, $i \neq 1$. For a node $u$ in $T_n[i]$, a fault-free path $P$ from $u$ to $\rho_i(\varepsilon)$ in $T_n[i]$ can be constructed in time \[ \sum_{i=1}^k |c_i|, \] where $|c_i|$ denotes the number of symbols in the cycle $c_i$, then the distance $\text{dist}(u)$ from the node $u$ to the identity node $\varepsilon$ is given by the following formula [2].

$$\text{dist}(u) = \begin{cases} 
    l + k & \text{if the primary cycle is a trivial cycle} \\
    l + k - 2 & \text{if the primary cycle is a nontrivial cycle}
\end{cases}$$

Combining this formula with the above discussion on the effect of applying the operations $\rho_i$ and $\sigma_a$ on a permutation, we derive the following necessary and sufficient rules for tracing a shortest path from the node $u$ to the identity node $\varepsilon$ in the $n$-star network $S_n$.

**Shortest Path Rules**

**Rule 1.** If the primary cycle is a trivial cycle in $u$, then in the next node on any shortest path from $u$ to $\varepsilon$, a nontrivial cycle $c_i$ is merged into the primary cycle. This corresponds to applying the operation $\sigma_a$ on $u$ with $a \in c_i$.

**Rule 2.** If the primary cycle $c_1 = (a_{11}a_{12} \cdots a_{1n_1}a_{1n_1+1})$ is a nontrivial cycle in $u$, where $a_{1n_1+1} = 1$, then in the next node on any shortest path from $u$ to $\varepsilon$, either a nontrivial cycle $c_i \neq c_1$ is merged into the primary cycle (this corresponds to applying the operation $\sigma_a$ on $u$, where $a \in c_i$), or the symbol $a_{11}$ is deleted from the primary cycle $c_1$ (this corresponds to applying the operation $\sigma_{a_{12}}$ on $u$).

**Fact 2.1.** If an edge $[u, v]$ in $S_n$ does not lead to a shortest path from $u$ to $\varepsilon$, then $\text{dist}(v) = \text{dist}(u) + 1$. Consequently, let $P$ be a path from $u$ to $\varepsilon$ in which exactly $k$ edges do not follow the Shortest Path Rules, then the length of the path $P$ is equal to $\text{dist}(u) + 2k$.

Two simple procedures will be used in following a shortest path from a node $u$ to $\varepsilon$. The first is called the “Delete” procedure, written as $\rightarrow \cdots \rightarrow$, which repeatedly deletes the first symbol in the non-trivial primary cycle. The second one is called the “Merge-Delete” procedure [6], written as $\rightarrow M + D \rightarrow$, which works in two stages: first repeatedly merges in an arbitrary order each of the nontrivial cycles into the primary cycle, then repeatedly deletes the first symbol in the primary cycle. It is easy to verify that both the “Delete” procedure and the “Merge-Delete” procedure follow the Shortest Path Rules strictly.

For the $n$-star network $S_n$, let $S_n[i]$ be the set of nodes in which the symbol 1 is at the $i$th position. It is well-known [1] that the set $S_n[1]$ is an independent set (i.e., no two nodes in $S_n[1]$ are adjacent to each other), and the subgraph induced by the set $S_n[i]$ for $i \neq 1$ is an $(n-1)$-star network. Note that a node is in the substar $S_n[i]$, $i \neq 1$, if and only if the primary cycle of the node is of the form $(\cdots i1)$, and a node is in $S_n[1]$ if and only if the primary cycle of the node is a trivial cycle (1).

A nice property of the Delete procedure and Merge-Delete procedure is that if they start with a node $u$ in a substar $S_n[i]$, $i \neq 1$, then all nodes, possibly except the last one, on the constructed shortest path are also in the substar $S_n[i]$.
Proof. Day and Tripathi [8] have presented the following results: for any node $u$ in the $n$-star network $S_n$, there is a width $n - 1$ container $C(u, \varepsilon)$ between $u$ and $\varepsilon$, where each path in $C(u, \varepsilon)$ can be constructed in time $O(n)$ independently. Moreover, each path in $C(u, \varepsilon)$ has at most two edges (in case $u$ is in $S_n[1]$, at most one edge) not following the Shortest Path Rules. Applying Day and Tripathi’s algorithm on the $(n-1)$-substar $S_n[i]$, we can obtain such a width $n - 2$ container $C(u, \rho_i(\varepsilon))$ in $S_n[i]$ connecting the node $u$ and the node $\rho_i(\varepsilon)$. Since $k_i \leq n - 3$, by tracing at most $k_i + 1$ of these paths in $C(u, \rho_i(\varepsilon))$, in time $O(k_i n + n)$, we can find a path from $u$ to $\rho_i(\varepsilon)$ in $T_n[i]$ that satisfies the conditions in the lemma.

Our parallel routing algorithm is heavily based on the following concept of bridging paths that connect a given node to a specific substar network in the $n$-star network.

**Definition** Let $u$ be a node in the $n$-star network $S_n$ and $u'$ be a neighbor of $u$ in the substar $S_n[i], i \neq 1$. For each neighbor $v$ of $u'$, $v \neq u$, a $(u', j)$-bridging path (of length at most 4) from $u$ to the substar $S_n[j], j \neq 1, i$, is defined as follows: if $v$ is in $S_n[1]$ then the path is $[u, u', v, \rho_j(v)]$, while if $v$ is in $S_n[i]$ then the path is $[u, u', v, \rho_i(v), \rho_j(\rho_i(v))]$.

Thus, from each neighbor $u'$ in $S_n[i]$ of the node $u$, $i \neq 1$, there are $n - 2$ $(u', j)$-bridging paths of length bounded by 4 that connect the node $u$ to the substar $S_n[j]$. See Figure 1 for an intuitive illustration for bridging paths.

Since no two nodes in $S_n[i]$ share the same neighbor in $S_n[1]$ and no two nodes in $S_n[1]$ share the same neighbor in $S_n[j]$, for any neighbor $u'$ of $u$, two $(u', j)$-bridging paths from $u$ to $S_n[j]$ have only the nodes $u$ and $u'$ in common. Moreover, for any two neighbors $u'$ and $u''$ of $u$ in $S_n[i]$ (in this case, the node $u$ must itself be also in $S_n[i]$), since $u'$ and $u''$ have no other common neighbor except $u$ (see, for example, [7, 8]), a $(u', j)$-bridging path from $u$ to $S_n[j]$ and

![Figure 1: Bridging paths from node $u$ to substar $S_n[j]$: (A) $u$ is in $S_n[1]$; (B) $u$ is in $S_n[i]$](image-url)
Moreover, for two divergent
Lemma 3.2
A divergent
edges. In particular, if there are three edges not following the Shortest Path Rules.

Definition
Let \( u \) be a node in \( S_n \) and let \( u' \) be a neighbor of \( u \) in \( S_n[i], i \neq 1, j \). A \((u', j)\)-bridging path \( P \) from the node \( u \) to the substar \( S_n[j] \) is divergent if in the subpath of \( P \) from \( u \) to \( S_n[1] \), there are three edges not following the Shortest Path Rules.

Note that the subpath from \( u \) to \( S_n[1] \) of a \((u', j)\)-bridging path \( P \) contains at most three edges. In particular, if \( P \) contains only two edges, then \( P \) is automatically non-divergent.

**Lemma 3.2** A divergent \((u', j)\)-bridging path \( P \) from a node \( u \) to a substar \( S_n[j] \), where \( u' \in S_n[i], i \neq 1, j \), can be extended into a path \( Q \) from \( u \) to \( \rho_j(\varepsilon) \) in time \( O(n) \), such that at most 4 edges in \( Q \) do not follow the Shortest Path Rules and the extended part is entirely in \( S_n[j] \).

Moreover, for two divergent \((u', j)\)-bridging paths, the two corresponding extended paths only share the nodes \( u, u' \), and \( \rho_j(\varepsilon) \).

**Proof.** Let \( P \) be a divergent \((u', j)\)-bridging path from the node \( u \) to the substar \( S_n[j] \), where \( u' \) is a neighbor of \( u \). Since the path \( P \) is divergent, it has length 4. Thus, the path can be written as \( P = \{u, u', v, v', v''\} \), where \( u' \) is in \( S_n[i], v \) is a neighbor of \( u' \) in \( S_n[i], v' = \rho_i(v) \) is in \( S_n[1] \), and \( v'' = \rho_j(v') \) is in \( S_n[j] \).

Let \( u' = (a_1 \cdots a_q i1) **, where ** stands for “other cycles”. Since the edge \([u', v]\) does not follow the Shortest Path Rules and \( v \) is in \( S_n[i] \), the node \( v \) must have the form either \( v = (b_1 \cdots a_q i1) * * \), or \( v = (b_1 \cdots a_q i1)(1) * * \), where \( 2 \leq q \leq p \). Now since \([v, v']\) is an edge in \( S_n \), and \( v' \) is in \( S_n[1] \), the node \( v' \) must be of the form either \( v' = (b_1 \cdots a_q i1)(1) * * \), or \( v' = (b_1 \cdots a_q i1)(1) * * \). Moreover, since the edge \([v, v']\) does not follow the Shortest Path Rules, when \( v' = (b_1 \cdots a_q i1)(1) * * \), we must have \( q + 1 \leq p \). In summary, if \( P \) is a divergent path, then the fourth node \( v' \) on \( P \) must be of form \((b_1 b_2 \cdots i)(1)\), where the cycle \((b_1 b_2 \cdots i)\) is nontrivial. Moreover, the \((u', j)\)-bridging path \( P \) is distinguished from other \((u', j)\)-bridging paths by the symbol \( b_1 \) in the above format (i.e., two different divergent \((u', j)\)-bridging paths will have two different symbols \( b_1 \) in the above format).

Now consider the fourth edge \([v', v'']\) on the path \( P \), where \( v'' \) is in \( S_n[j], j \neq 1, i \).

If the symbol \( j \) is in a trivial cycle in the node \( v' \), then the extended path \( Q \) is

\[
Q: \quad u \rightarrow u' \rightarrow v \rightarrow v' = (b_1 b_2 \cdots i)(1) * * \rightarrow v'' = (b_1 b_2 \cdots i)(j1) * * \rightarrow (b_2 \cdots ib_1 j1) * * \rightarrow M+D \rightarrow (b_1 j1) \rightarrow (j1) = \rho_j(\varepsilon) \quad (1)
\]

The extended path \( Q \) has no common nodes in \( S_n[j] \), except \( \rho_j(\varepsilon) \), with the paths extended from the other \((u', j)\)-bridging paths since the symbol \( b_1 \) distinguishes the path \( Q \) from other extended paths: the first part of \( Q \) has a unique cycle \((b_1 b_2 \cdots i)\) while the second part of \( Q \) has a cycle of the unique format \((b_1 b_1 j1)\).

If the symbol \( j \) is in a nontrivial cycle in \( v' \), then there are two possible cases:

**Case 1.** \( j \) is not in the cycle \((b_1 b_2 \cdots i)\). Then the extended path \( Q \) is

\[
Q: \quad u \rightarrow u' \rightarrow v \rightarrow v' = (b_1 b_2 \cdots i)(1) * * \rightarrow v'' = (b_1 b_2 \cdots i)(\cdots j1) * * \rightarrow D \rightarrow (b_1 b_2 \cdots i)(j1) * * \rightarrow (b_2 \cdots ib_1 j1) * * \rightarrow M+D \rightarrow (b_1 j1) \rightarrow (j1) = \rho_j(\varepsilon) \quad (2)
\]
Again, because of \( b_1 \), the extended path \( Q \) has no common nodes in \( S_n[j] \), except \( \rho_j(\varepsilon) \), with the paths extended from the other \((u',j)\)-bridging paths.

**Case 2.** \( j \) is in the cycle \((b_1b_2 \cdots i)\).

If \( j = b_1 \), then \((b_1b_2 \cdots i) = (b_2 \cdots ij)\), and the path \( Q \) is

\[
Q : \ u \to u' \to v \to v'' = (b_2 \cdots ij)(1) \to (ij1) \to (j1) = \rho_j(\varepsilon) \quad (3)
\]

This path is node-disjoint from the paths extended from the other \((u',j)\)-bridging paths because all nodes of it in \( S_n[j] \) contain a cycle of form \((\cdots ij1)\). If \( j \neq b_1 \), then \((b_1b_2 \cdots i) = (b_1 \cdots j \cdots i)\), and the path \( Q \) is

\[
Q : \ u \to u' \to v \to v'' = (b_1 \cdots j \cdots i)(1) \to (\cdots ib_1 \cdots j1) \to (\cdots ib_1j1) \to (b_1j1) \to (j1) = \rho_j(\varepsilon) \quad (4)
\]

Again this path is node-disjoint from the paths extended from the other \((u',j)\)-bridging paths because of the symbol \( b_1 \).

For all cases, we can easily verify that the path \( Q \) contains at most 4 edges not following the Shortest Path Rules, and that the part of \( Q \) extended from the \((u',j)\)-bridging path \( P \) is entirely in the substar \( S_n[j] \). Finally, from (1)-(4), it can be easily see that the construction of the path \( Q \) takes time \( O(n) \) and is independent of the construction of other paths.

\[ \square \]

### 4 Parallel routing algorithm on faulty star networks

We are ready to present our main algorithm. Suppose that the \( n \)-star network \( S_n \) has at most \( n - 3 \) faulty nodes. Let \( T_n \) be the copy of \( S_n \) with the faulty nodes removed and the node \( \varepsilon \) is in \( T_n \). Denote by \( T_n[i] \) the subgraph of the substar \( S_n[i] \) induced by the non-faulty nodes in \( S_n[i] \). Let \( deg_f(u) \) be the degree of node \( u \) in the graph \( T_n \). The algorithm is presented in Figure 2.

We give more detailed explanations for each step of the algorithm.

Step 1 of the algorithm constructs paths between some neighbors of the node \( u \) and the node \( \varepsilon \). Step 2 of the algorithm maximally pairs the rest neighbors of \( u \) and \( \varepsilon \) in \( T_n \), and Step 3 constructs a path in \( T_n \) from \( u \) to \( \varepsilon \) for each of these pairs. Thus, the number of paths constructed by the algorithm **Parallel-Routing** is exactly the minimum of degree of \( u \) and degree of \( \varepsilon \) in \( T_n \). What remains to explain is how these paths are constructed, in time \( O(n^2) \), so that these paths are fault-free, node-disjoint, and of length bounded by \( dist(u) + 8 \).

**Step 1 of the algorithm**

In case the node \( u \) is in \( T_n[1] \), for each index \( j \neq 1 \) such that both \( \rho_j(u) \) and \( \rho_j(\varepsilon) \) are in \( T_n[j] \), by Lemma 3.1, we can construct in time \( O(k_jn + n) \) a path \( Q_j \) from \( \rho_j(u) \) to \( \rho_j(\varepsilon) \) that is entirely in \( T_n[j] \) such that at most two edges in \( Q_j \) do not follow the Shortest Path Rules, where \( k_j \leq n - 3 \) is the number of faulty nodes in the substar \( S_n[j] \). Now the path \( P_j \) from \( u \) through \( Q_j \) to \( \varepsilon \) has at most three edges not following the Shortest Path Rules (the edge \([\rho_j(\varepsilon), \varepsilon] \) always follows the Shortest Path Rules). By Fact 2.1, the length of the path \( P_j \) is bounded by \( dist(u) + 6 \). This path \( P_j \) is disjoint with other paths constructed in Step 1 because all internal nodes of \( P_j \) are in \( T_n[j] \).
Parallel-Routing

Input: a node $u$ in $T_n$, which is a copy of $S_n$ with at most $n - 3$ faulty nodes removed.

Output: A container of the maximum width from $u$ to $\varepsilon$ in $T_n$.

1. if the node $u$ is in $T_n[1]$ 
   1.1. then 
      for each $j \neq 1$ such that both $\rho_j(u)$ and $\rho_j(\varepsilon)$ are in $T_n[j]$ do 
      construct a path $P_j$ of length $\leq \text{dist}(u) + 6$ from $u$ to $\varepsilon$ such that all internal nodes of the path are in $T_n[j]$; 
   1.2. else (* the node $u$ is in $T_n[i]$, for some $i \neq 1$ *) 
      1.2.1. if the node $\rho_i(\varepsilon)$ is in $T_n[i]$ then pick a neighbor $v$ of $u$ in $T_n[i]$ and construct a path $P_v$ of length $\leq \text{dist}(u) + 4$ from $u$ to $\varepsilon$ via $v$ such that all internal nodes of $P_v$ are in $T_n[i]$ and $P_v$ does not intersect a $(u', j)$-bridging path for any neighbor $u' \neq v$ of $u$ and for any $j \neq 1, i$; 
      1.2.2. if the node $u_1 = \rho_i(u)$ is in $T_n[1]$ then find a $j$, $j \neq 1, i$, such that both $\rho_j(u_1)$ and $\rho_j(\varepsilon)$ are in $T_n[j]$, and extend the path $[u, u_1, \rho_j(u_1)]$ to a path $P_1$ of length $\leq \text{dist}(u) + 8$ from $u$ to $\varepsilon$ such that all nodes between $\rho_j(u_1)$ and $\rho_j(\varepsilon)$ are in $T_n[j]$; 
   1.2. else (* the node $u$ is in $T_n[i]$, for some $i \neq 1$ *) 
   2. maximally pair the remaining neighbors of $u$ and $\varepsilon$ in $T_n$; 
  3. for each pair $(\rho_h(u), \rho_j(\varepsilon))$ constructed in step 2 do 
    3.1. if there is a non-divergent $(\rho_h(u), j)$-bridging path with no used nodes in $T_n$ then pick this non-divergent $(\rho_h(u), j)$-bridging path $P$ else pick a divergent $(\rho_h(u), j)$-bridging path $P$ with no used nodes in $T_n$; 
    3.2. extend $P$ into a path $P_j$ of length $\leq \text{dist}(u) + 8$ from $u$ to $\varepsilon$ such that the extended part of $P_j$ is entirely in $T_n[j]$;

Figure 2: Parallel routing in $S_n$ with $\leq n - 3$ faulty nodes
In the case the node \( u \) is in \( T_n[i] \), for some \( i \neq 1 \), Step 1 constructs at most two paths from \( u \) to \( \varepsilon \). If the node \( \rho_l(\varepsilon) \) is in \( T_n[i] \), by Lemma 3.1, we can construct in time \( O(k_i n + n) \) a path \( Q_v \) in \( T_n[i] \) from \( u \) to \( \rho_l(\varepsilon) \) (where \( v \) is the second node on \( Q_v \)) such that at most two edges in \( Q_v \) do not follow the Shortest Path Rules, where \( k_i \leq n - 3 \) is the number of faulty nodes in \( S_n[i] \). This path \( Q_v \) plus the edge \([\rho_l(\varepsilon), \varepsilon]\) gives a path \( P_v \) of length bounded by \( \text{dist}(u) + 4 \) in which all internal nodes are in \( T_n[i] \). We show that the path \( P_v \) can be constructed without intersecting a \((u', j')\)-bridging path from \( u \) for any neighbor \( u' \neq v \) of \( u \) and any \( j' \neq 1, i \). Suppose the contrast that the path \( P_v \) intersects such \((u', j')\)-bridging paths. Let \( w \) be the last node on \( P_v \) that belongs to such a \((u', j')\)-bridging path \( Q_w \) from \( u \). Note that the neighbor \( u' \) of \( u \) is uniquely determined by the node \( w \) since for two different neighbors \( u' \) and \( u'' \) of \( u \) in \( S_n[i] \), a \((u', j')\)-bridging path and a \((u'', j'')\)-bridging path have no common nodes except \( u \). Now, if we let \( P_w \) be the subpath of \( Q_w \) from \( u \) to \( w \) plus the subpath of \( P_v \) from \( w \) to \( \varepsilon \), then it is not hard to see that the length of the path \( P_w \) is not larger than the length of the path \( P_v \), and that the path \( P_w \) does not intersect any \((u'', j'')\)-bridging path from \( u \) for any neighbor \( u'' \neq u' \) of \( u \) and for any \( j'' \neq 1, i \).

If the neighbor \( u_1 = \rho_1(u) \) of \( u \) is in \( T_n[1] \), consider the \( n - 2 \) pairs \((\rho_j(u_1), \rho_j(\varepsilon))\) of neighbors of \( u_1 \) and \( \varepsilon \), where \( j \neq 1, i \). Since the \( n\)-star network \( S_n \) has at most \( n - 3 \) faulty nodes, one of these pairs \((\rho_j(u_1), \rho_j(\varepsilon))\) has both nodes in \( T_n[j] \). By Lemma 3.1, a path \( Q_1 \) from \( \rho_j(u_1) \) to \( \rho_j(\varepsilon) \) can be constructed in \( T_n[j] \) in time \( O(k_i n + n) \) such that at most two edges of \( Q_1 \) do not follow the Shortest Path Rules, where \( k_j \leq n - 3 \) is the number of faulty nodes in the substar \( S_n[j] \). Now the concatenation \( P_1 \) of the path \([u, u_1, \rho_j(u_1)]\), the path \( Q_1 \), and the edge \([\rho_j(\varepsilon), \varepsilon]\) gives a path in \( T_n \) from \( u \) to \( \varepsilon \) of length bounded by \( \text{dist}(u) + 8 \). Note that this path is obviously node-disjoint with the path constructed in Step 1.2.1.

**Step 2 of the algorithm**

Step 2 of the algorithm obviously takes time \( O(n) \).

**Step 3 of the algorithm**

We consider Step 3 of the algorithm for two different cases.

**Case 3.1.** The node \( u \) is in \( T_n[1] \).

For each pair \((\rho_h(u), \rho_j(\varepsilon))\) formed in Step 2, the nodes \( \rho_h(\varepsilon) \) and \( \rho_j(u) \) must be faulty, otherwise the index \( h \) or \( j \) would have been picked in Step 1.1.

We construct a path \( Q_{hj} \) from \( u \) to \( \varepsilon \) by concatenating a \((\rho_h(u), j)\)-bridging path from \( u \) to \( T_n[j] \) with a path \( Q_j \) entirely in \( T_n[j] \). Note that such a path \( Q_{hj} \) contains one node in \( T_n[1] \), and all other nodes in \( T_n[h] \) and \( T_n[j] \). We say that a node \( u_1 \neq u \) in \( T_n[1] \) is used if \( u_1 \) has been used by a path constructed by the algorithm. Inductively, assume that for \( r \) pairs of neighbors of \( u \) and \( \varepsilon \) in Step 2, \( r \) node-disjoint paths have been constructed. We consider the \((r + 1)\)st pair \((\rho_h(u), \rho_j(\varepsilon))\).

Each \((\rho_{h'}(u), \rho_{j'}(\varepsilon))\) of the previous \( r \) pairs implies at least two faulty nodes, the node \( \rho_{h'}(\varepsilon) \) and the node \( \rho_{j'}(u) \), and one used node in \( T_n[1] \). Also notice that the paths constructed in Step 1.1 do not use any nodes in \( T_n[1] \). Thus, the number of faulty nodes in the sets \( S_n[1] \), \( S_n[h] \), and \( S_n[j] \) is at most \((n - 3) - 2r = n - 2r - 3 \). Let \( k_j \) be the number of faulty nodes in \( S_n[j] \), \( k_j \leq n - 2r - 3 \).

**Case 3.1.A.** There is a non-divergent \((\rho_h(u), j)\)-bridging path \( P_{hj} \) in \( T_n \) for \( u \) that contains no used nodes. By definition, at most two edges before the last edge on \( P_{hj} \) do not follow the Shortest Path Rules. Consider the last edge \([v', v'']\) on \( P_{hj} \). If the edge \([v', v'']\) follows the Shortest Path Rules, then the path \( P_{hj} \) has at most two edges.
not following the Shortest Path Rules. By Lemma 3.1, we can construct a path $Q'_j$ in $T_n[j]$ from $v''$ to $\rho_j(\varepsilon)$ in time $O(k_jn + n)$ such that at most two edges in $Q'_j$ do not follow the Shortest Path Rules. Now the concatenation of the $(\rho_h(u), j)$-bridging path $P_{hj}$, the path $Q'_j$, and the edge $[\rho_j(\varepsilon), \varepsilon]$ gives a path $P_j$ from $u$ to $\varepsilon$ with at most 4 edges not following the Shortest Path Rules, whose length is bounded by $\text{dist}(u) + 8$.

If the edge $[v', v'']$ does not follow the Shortest Path Rules, then the path $P_{hj}$ may have three edges not following the Shortest Path Rules. Since $v'$ is in $T_n[1]$, its primary cycle is trivial. Now since $v''$ is in $T_n[j]$ and the edge $[v', v'']$ does not follow the Shortest Path Rules, the primary cycle of $v''$ must be $(j1)$. By Lemma 3.1, a path $Q'_j$ from $v''$ to $\rho_j(\varepsilon)$ in $T_n[j]$ can be constructed in time $O(k_jn + n)$ in which at most one edge does not follow the Shortest Path Rules. Now the concatenation of the $(\rho_h(u), j)$-bridging path $P_{hj}$, the path $Q'_j$, and the edge $[\rho_j(\varepsilon), \varepsilon]$ gives a path $P_j$ from $u$ to $\varepsilon$ in which at most 4 edges do not follow the Shortest Path Rules, whose length is bounded by $\text{dist}(u) + 8$.

Therefore, in case 3.1.A, for the pair $(\rho_h(u), \rho_j(\varepsilon))$, in time $O(k_jn + n)$, we can always construct a path $P_j$ with no used nodes and of length bounded by $\text{dist}(u) + 8$ from node $u$ to node $\varepsilon$ in the graph $T_n$. This path is node-disjoint with all previously constructed paths since the part extended from the $(\rho_h(u), j)$-bridging path $P_{hj}$ is entirely in $T_n[j]$ that is not used by any other paths.

**Case 3.1.B.** all non-divergent $(\rho_h(u), j)$-bridging paths contain used nodes.

There are totally $n - 2$ $(\rho_h(u), j)$-bridging paths from $u$ to $S_n[j]$. Suppose that $q'$ of them contain either faulty nodes or used nodes, and that $q = n - 2 - q'$ of them contain neither faulty nodes nor used nodes. We first show $q > 0$.

Assume the contrary $q = 0$. Then $q' = n - 2$. Since any two $(\rho_h(u), j)$-bridging paths from $u$ have only the nodes $u$ and $\rho_h(u)$ in common and there are at most $n - 3$ faulty nodes in $S_n$, there are $q'_1$ of these $n - 2$ $(\rho_h(u), j)$-bridging paths, $q'_1 > 0$, containing only used nodes. Each of the rest $q'_2 = q' - q'_1 = n - q'_1 - 2$ $(\rho_h(u), j)$-bridging paths contains at least one faulty node. Thus, at least $q'_1$ paths have been constructed by the algorithm for $q'_1$ pairs (note that each constructed path uses exactly one node in $T_n[1]$). Each $(\rho_h(u), \rho_j(\varepsilon))$ of these $q'_1$ pairs implies two faulty nodes $\rho_j'(u)$ and $\rho_h(u)$. Thus, the total number of faulty nodes in the $n$-star network $S_n$ would have been at least $q'_2 + 2q'_1 = n + q'_1 - 2 > n - 3$, contradicting the assumption that the $n$-star network $S_n$ has at most $n - 3$ faulty nodes. This shows $q > 0$, i.e., there is at least one $(\rho_h(u), j)$-bridging path in $T_n$ that contains no used nodes.

By the assumption, the $q$ $(\rho_h(u), j)$-bridging paths with no used nodes in $T_n$ are all divergent. By Lemma 3.2, these $q$ $(\rho_h(u), j)$-bridging paths can be extended into $q$ paths from $u$ to $\rho_j(\varepsilon)$ with at most 4 edges not following the Shortest Path Rules. The constructed paths contain no used nodes since the extended part of each path is entirely in $S_n[j]$. Moreover, no two of these $q$ paths share a node that is not $u$, $\rho_h(u)$, and $\rho_j(\varepsilon)$.

We claim that at least one of these $q$ extended paths contains no faulty nodes. To the contrary, if each of these $q$ extended paths contains at least one faulty node, then the total number of faulty nodes in the sets $S_n[1]$, $S_n[2]$, and $S_n[j]$ is at least $q + (q' - r) = n - r - 2 > n - 2r - 3$ (recall that $r$ is the number of paths that have been constructed by the algorithm so far. Thus, among the $q'$ $(\rho_h(u), j)$-bridging paths that contain either faulty nodes or used nodes, at least $q' - r$ of them must each contain at least one faulty node). This contradicts the fact that there are at most $n - 2r - 3$ faulty nodes in the sets $S_n[1]$, $S_n[2]$, and $S_n[j]$.

Thus, an extended path $Q'_{hj}$ from $u$ to $\rho_j(\varepsilon)$ in $T_n$ without used nodes can be constructed.
This path $Q'_{hj}$ plus the edge $[\rho_j(\varepsilon), \varepsilon]$ gives a path $P_j$ from $u$ to $\varepsilon$ with no used nodes and with at most 4 edges not following the Shortest Path Rules. Thus, the length of the path $P_j$ is bounded by $\text{dist}(u) + 8$. Moreover, the path $P_j$ can be constructed in time $O(k_j n + n)$ by tracing at most $k_j + 1$ of the extended paths from $u$ to $\rho_j(\varepsilon)$. Finally, this path is node-disjoint with all previously constructed paths since its extended part is entirely in $T_n[j]$, which is not used by any other paths.

**Case 3.2.** The node $u$ is in $T_n[i]$, for some $i \neq 1$.

In this case, the node $u$ has one neighbor in $S_n[1]$, and $n - 2$ neighbors in $S_n[i]$ (see Figure 1). Note that if the neighbor $\rho_k(u)$ of $u$ is in $T_n[1]$, then a path from $u$ to $\varepsilon$ via $\rho_k(u)$ has been constructed in Step 1.2.2. Thus, we only need to consider the neighbors of $u$ that are in $S_n[i]$.

Again we assume that the algorithm has constructed $r$ paths from $u$ to $\varepsilon$ by extending $r$ bridging paths from $u$. Consider the $(r + 1)$st pair $(\rho_h(u), \rho_j(\varepsilon))$.

Since the $n$-star network contains no cycle of length less than 6 [7, 8], two neighbors of $u$ share no common neighbors except $u$. Let $u_1$ and $u_2$ be two neighbors of $u$ in $S_n[i]$. Since no two nodes in $S_n[i]$ have the same neighbor in $S_n[1]$ and no two nodes in $S_n[1]$ have the same neighbor in $S_n[j]$, a $(u_1, j_1)$-bridging path and a $(u_2, j_2)$-bridging path share no common nodes except $u$ for any $j_1$ and $j_2$. Therefore, for the previous $r$ paths from $u$ to $\varepsilon$ constructed by the algorithm by extending bridging paths from $u$, none of them would intersect a $(\rho_h(u), j)$-bridging path. Thus, no $(\rho_h(u), j)$-bridging path contains used nodes.

Thus, if there is a non-divergent $(\rho_h(u), j)$-bridging path $P_{hj}$ with no faulty nodes, we can extend the path $P_{hj}$, in the way of Case 3.1.A, into a path $P_j$ from $u$ to $\varepsilon$ such that the length of the path $P_j$ is bounded by $\text{dist}(u) + 8$ and the extended part of $P_j$ is entirely in $T_n[j]$. On the other hand, if all non-divergent $(\rho_h(u), j)$-bridging paths contain faulty nodes, then, as in Case 3.1.B, we can extend at least one divergent $(\rho_h(u), j)$-bridging path from $u$ into a path $P_j$ from $u$ to $\varepsilon$ such that the length of the path $P_j$ is bounded by $\text{dist}(u) + 8$ and the extended part of $P_j$ is entirely in $T_n[j]$.

We summarize the above discussion in the following main theorem.

**Theorem 4.1** Let $T_n$ the $n$-star network $S_n$ with at most $n - 3$ faulty nodes removed, and suppose the nodes $u$ and $\varepsilon$ are in $T_n$. Then in time $O(n^2)$ the algorithm Parallel-Routing constructs a container of width $t$ and length bounded by $\text{dist}_f(u) + 8$ from the node $u$ to the node $\varepsilon$, where $t$ is the minimum of degree of $u$ and degree of $v$ in $T_n$, and $\text{dist}_f(u)$ is the distance between nodes $u$ and $\varepsilon$ in $T_n$.

**Proof.** As we have discussed in detail above, the algorithm Parallel-Routing constructs a width $t$ container of length bounded by $\text{dist}(u) + 8$ from the node $u$ to the node $\varepsilon$ in $T_n$, where $\text{dist}(u)$ is the distance between $u$ and $\varepsilon$ in the nonfaulty $n$-star network $S_n$. Since the distance $\text{dist}_f(u)$ between $u$ and $\varepsilon$ in the $n$-star network $S_n$ with faults (i.e., the graph $T_n$) cannot be smaller than $\text{dist}(u)$, we conclude that the length of the container constructed by our algorithm is bounded by $\text{dist}_f(u) + 8$. The only thing remaining is to show that the running time of the algorithm is bounded by $O(n^2)$.

Each path is constructed by the algorithm by searching a proper path in a specific substar $S_n[j]$, which takes time $O(k_j n + n)$, where $k_j$ is the number of faulty nodes in the substar $S_n[j]$. No substar is used in extending more than one such a path. Therefore, the time complexity for constructing all these paths is bounded by $O(k_2 n + k_3 n + \cdots + k_n n + n(n - 1)) = O(n^2)$ since
by our assumption, the number $k_2 + k_3 + \cdots + k_n$ is bounded by $n - 3$. Thus, the time complexity of the algorithm Parallel-Routing is bounded by $O(n^2)$. 

**Corollary 4.2** The star networks are strongly fault tolerant.

The following example shows that the bound on the container length in Theorem 4.1 is actually almost optimal. Consider the $n$-star network $S_n$. Let the source node be $u = (21)$, here we have omitted the trivial cycles in the cycle structure. Then $dist(u) = 1$. Suppose that all neighbors of $u$ and $\varepsilon$ are non-faulty. By Theorem 4.1, there is a container of width $n - 1$ in $T_n$ from $u$ fro $\varepsilon$. Thus, for each $i$, $3 \leq i \leq n$, the edge $[u, u_i]$, where $u_i = (i21)$, leads to one path $P_i$ in the container. Note that the edge $[u, u_i]$ does not follow the Shortest Path Rules. Now suppose that the $n - 3$ faulty nodes in $S_n$ are $(i2)(1)$, for $i = 3, 4, \ldots, n - 1$ (thus, $dist_f(u) = dist(u)$). Then the third node on the path $P_i$ must be $v_i = (j21)$ for some $j \neq 1, 2, i$, and the edge $[u_i, v_i]$ does not follow the Shortest Path Rules. Since the only edge from $v_i$ that follows the Shortest Path Rules is the edge $[v_i, u_i]$, the next edge $[v_i, w_i]$ on $P_i$ again does not follow the Shortest Path Rules. Now since all the first three edges on $P_i$ do not follow the Shortest Path Rules, by Fact 2.1, $dist(w_i) = dist(u) + 3 = 4$, and the path $P_i$ needs at least four more edges to reach $\varepsilon$. That is, the length of the path $P_i$ is at least $7 = dist(u) + 6 = dist_f(u) + 6$. Thus, with the $n - 3$ faulty nodes, among any width $n - 1$ containers from $u$ to $\varepsilon$, at least $n - 3$ of them must have length larger than or equal to $dist_f(u) + 6$, which is very close to the bound $dist_f(u) + 8$ given in Theorem 4.1.

The situation given above seems a little special since the distance $dist(u)$ from $u$ to $\varepsilon$ is very small. In fact, even for large distance nodes $u$, we can still construct many examples in which any container of width $n - 1$ connecting $u$ and $\varepsilon$ must contain at least one path of length at least $dist(u) + 6$. We leave this to the interested readers.

**5 Conclusion**

Network strong fault tolerance is a natural extension of the study of network fault tolerance and network parallel routing. In particular, it studies the fault tolerance of large size networks with faulty nodes. In this paper, we have demonstrated that the star networks are strongly fault tolerant. We presented an algorithm of running time $O(n^2)$ that for two given non-faulty nodes $u$ and $v$, constructs the maximum width container of length bounded by $dist(u, v) + 8$ from $u$ to $v$, under the condition that the $n$-star network has at most $n - 3$ faulty nodes. The time complexity of our algorithm is optimal since each path from $u$ to $v$ may have length as large as $\Theta(n)$, and there can be a container of width as large as $n - 1$ from $u$ to $v$. Thus, even printing these paths should take time $\Omega(n^2)$. We have shown that the length of the container constructed by our algorithm is almost optimal. Finally, our algorithm does not require prior knowledge of the failures: in a single round communication, the algorithm can find out the faulty neighbors of the nodes $u$ and $v$, then the algorithm constructs a container and avoids faulty nodes whenever they are encountered during the routing.

We should mention that Rescigno [21] recently developed a randomized parallel routing algorithm on star networks with faults, based on the Information Dispersal Algorithm (IDA) [20]. The algorithm in [21] is randomized thus it does not always guarantee a container of the
maximum width. Moreover, in terms of the length of the constructed container and running time of the algorithm, our algorithm seems also to have provided significant improvements.

The study of strong fault tolerance shows another advantage of the star networks over the popular hypercube networks. In particular, the orthogonal partition of the star networks [7], i.e., decomposing the n-star network into \( n - 1 \) \((n-1)\)-substars \( S_n[2], \ldots, S_n[n] \), plus an independent set \( S_n[1] \), seems very convenient for construction of a container: we basically can construct a path in each separated substar, ensuring that the constructed path is node-disjoint with other paths. On the other hand, other popular network topologies, such as the hypercube networks, do not seem to have this nice decomposition structure. For example, in the construction of a container in a hypercube network with faults, we seemed not able to construct each path using a separated subcube, which has made the construction more involved [19].

**Acknowledgements.** The authors would like to thank the anonymous referees and Professor Frank Hsu for their constructive comments and suggestions, which have improved the presentation of the paper.

**References**


