Fault Tolerant Communication Algorithms on the Star Network Using Disjoint Paths

Paraskevi Fragopoulou and Selim G. Akl
(fragopou, akl)@qucis.queensu.ca
Department of Computing & Information Science
Queen’s University
Kingston, Ontario, Canada, K7L 3N6

Abstract

One way to achieve fault tolerant communication on interconnection networks is by exploiting and effectively utilizing the disjoint paths that exist between pairs of source and destination nodes. In this paper, we construct a graph that consists of \( n - 1 \) directed edge-disjoint spanning trees on the star network. This graph is used to derive fault tolerant algorithms for the single-node and multinode broadcasting, and for the single-node and multinode scattering problems under the all-port communication assumption. Fault tolerance is achieved by transmitting the same messages through a number of edge-disjoint spanning trees. These algorithms operate successfully in the presence of up to \( n - 2 \) faulty nodes or edges in the network.

1 Introduction

The star network was proposed in [1] as "an attractive alternative to the \( n \)-cube" topology for interconnecting processors in parallel computers. Each node is labeled with a different permutation of \( n \) distinct symbols and is connected to those nodes that can be obtained by interchanging its first with its \( i^{th} \) symbols, \( 2 \leq i \leq n \), Fig. 1. We call these \( n - 1 \) connections dimensions. As shown in [1, 2], \( S_n \) enjoys a number of properties desirable in interconnection networks. These include node and edge symmetry, maximal fault tolerance, and strong resilience. Because of its symmetry, the network is easily extensible, can be decomposed in various ways and allows for simple routing algorithms. In addition, \( S_n \) is superior to \( Q_n \) (the \( n \)-hypercube) with respect to two key properties: degree (number of edges at each node), and diameter (maximum distance between any two nodes) [1]. The degree of \( S_n \) is \( n - 1 \), sublogarithmic to the number of its nodes, while a hypercube has degree logarithmic to the number of its nodes. Several algorithms and properties on the star network have been derived in [6, 12].

In order for a network of processors to be candidate for parallel processing, it must lend itself to the derivation of efficient and fault tolerant communication algorithms. Working towards this direction in this paper, we construct a graph that consists of \( n - 1 \) directed edge-disjoint spanning trees on the star network. A node \( s \) of \( S_n \) is the root of an \( n - 1 \) directed edge-disjoint spanning trees graph, denoted by \( EDT_s \), if each of the \( n - 1 \) nodes adjacent to \( s \) is the root of a tree that spans all nodes of \( S_n \) except \( s \). The edges of the \( n - 1 \) spanning trees are directed from parent to children nodes, and the trees are edge-disjoint. This is the maximum number of edge-disjoint spanning trees that can be constructed on \( S_n \), since the latest is a regular network with connectivity \( n - 1 \). The depth of the derived \( EDT_s \) graph differs from the minimum possible depth by a small additive constant. The root node \( s \) of \( EDT_s \) has \( n - 1 \) node disjoint paths to each one of the other nodes, one path through each of the \( n - 1 \) edge-disjoint spanning trees. Similar graphs have been previously constructed on other interconnection networks, such as the binary hypercube [10] and the cube-connected-cycles (CCC) [8]. Previous attempts to exploit the disjoint paths on \( S_n \) focused on the disjoint paths between two nodes, or between one node and a set of \( n - 1 \) distinct nodes [3, 4, 11]. The construction of the \( EDT_s \) graph is equivalent to the problem of exploiting the disjoint paths between node \( s \) and all the other \( n! - 1 \) nodes of \( S_n \). It is the first time the problem is treated to this extend.

Using the \( EDT_s \) graph we derive fault tolerant algorithms for the single-node and multinode broadcasting, and for the single-node and multinode scattering problems on \( S_n \). In broadcasting, one node distributes the same group of messages to all other nodes, while in scattering one node distributes distinct groups of messages to all other nodes.
of messages to each other node. We consider broadcasting and scattering from a single node (single-node broadcasting and scattering) and simultaneously from all nodes of the network (multinode broadcasting and scattering). The fault tolerant algorithms are developed by transmitting multiple copies of the same messages through a number of edge-disjoint spanning trees. A source processor $s$, sends the $M$ messages it wishes to transmit through each of the $n - 1$ edge-disjoint spanning trees of $EDT_s$. As a consequence, each destination node receives $n - 1$ copies of the messages from $s$ through $n - 1$ disjoint paths, under fault free conditions. However, if up to $n - 2$ faulty nodes or edges are present, it is still guaranteed that one of the $n - 1$ disjoint paths that lead to a destination node will be fault free, and a copy of the messages will successfully reach each destination node. As pointed out by many authors [9, 13], the advantage of this method is that no prior knowledge of the faulty elements of the network is required. The overhead of keeping all the processors up-to-date with the network faulty nodes or edges could be severe. The degree of fault tolerance of the algorithm can be adjusted, based on the network reliability. Furthermore, the special properties of the $EDT_s$ graph, which are discussed in the following sections, allow messages originating at individual nodes during a multinode broadcasting or scattering algorithm to be interleaved in such a manner that no two messages contend for the same edge at any time during the execution of the algorithm.

The algorithms are derived for the store-and-forward communication model. A processor must receive the entire message before it can process it and retransmit it. The communication is bidirectional, meaning that an edge can be used for message transmission in both directions at each time step and can be viewed as two directional edges. Each processor can exchange messages of fixed length with all of its neighbors at each time step, i.e. the all-port communication assumption (as opposed to the one-port assumption). Each message requires unit time to be transmitted on an edge. Finally, each source processor transmits the same number of $M$ messages to each one of its destination nodes. The number of messages $M$ is usually considered to be large.

This paper is organized as follows: Following the introduction to the subject in section 1, notations and definitions that are used throughout the paper are introduced in section 2. Section 3 presents the $n - 1$ directed edge-disjoint spanning trees on the star network. In section 4, we demonstrate several applications of this structure in the area of fault tolerant data communication. More specifically, lower bounds for all the problems are derived in subsection 4.1. The fault tolerant algorithms for the single-node and multinode broadcasting, and for single-node and multinode scattering problems, are presented in subsections 4.2 to 4.5, respectively. We conclude in section 5, along with a summary of the results.

2 Notations and definitions

The star network belongs to the family of Cayley graphs [2]. For networks in this class, nodes correspond to the elements of a finite group and edges correspond to a set of generators that act on the elements of the group. In this context, the $i$th, $2 \leq i \leq n$, generator of $S_n$ connects node $v = v_1 v_2 \ldots v_{i-1} v_i v_{i+1} \ldots v_n$ to node $v' = v_1 v_2 \ldots v_{i-1} v_i v_{i+1} \ldots v_n$, which results by transposing the first and the $i$th symbols of node $v$. We say that edge $(v, v')$ is of dimension $i$, or $dim(v, v') = i$.

Thus, each node of $S_n$ is connected to $n - 1$ other nodes through dimensions $2, 3, \ldots, n$. $S_{n-1}^k$, $2 \leq k \leq n$, is the subnetwork induced by all nodes of $S_n$ with symbol $1$ in the $k$th position of their permutation. It is well known that $S_{n-1}^k$, $2 \leq k \leq n$, is an $S_{n-1}$ defined on symbols $\{2, 3, \ldots, n\}$ [1]. For notation purposes, we use the symbol $S_{n-1}^k$ to denote the collection of $(n - 1)!$ isolated nodes of $S_n$ that start with symbol 1. The identity node of $S_n$ is the node labeled with the sorted permutation $1 2 \ldots n$.

We now define an operation on $S_n$ that will be of primary importance for the construction of the $n - 1$ directed edge-disjoint spanning trees. Having a spanning graph $EDT_{12 \ldots n}$, rooted at the identity node of $S_n$, we will derive an isomorphic spanning graph with the same properties, $EDT_s$, rooted at any other node $s$ of $S_n$, using a translation of $EDT_{12 \ldots n}$ with respect...
to \( s \). As a consequence, it is sufficient to construct a spanning graph rooted at the identity node of \( S_n \).

The translation operation on \( S_n \) is analogous to the exclusive-OR operation on nodes of the binary hypercube \([10, 3]\).

**Definition 1:** The translation of a node \( v \) with respect to node \( s \), is defined to be node \( Tr_s(v) = s \circ v = s_1s_2...s_{n-1}s_n \) (this is the operation of permutation composition). By translation of a network with respect to \( s \) we mean that each node of the network is translated with respect to \( s \).

The translation operation is an automorphism on \( S_n \) that preserves the dimension of each edge. If \((v, u)\) is an edge of dimension \( i \), then edge \((Tr_s(v), Tr_s(u))\) is also of dimension \( i \). This becomes clear, if we analytically express edges \((v, u)\) and \((Tr_s(v), Tr_s(u))\) as follows:

\[
(v_1v_2...v_{i-1}v_i+1...v_n, v_1v_2...v_{i-1}v_i+1...v_n)
\]

\[
(s_1s_2...s_{i-1}s_i,s_{i+1}...s_n, s_1s_2...s_{i-1}s_i,s_{i+1}...s_n)
\]

For example, edge \((3124, 4123)\) of \( S_4 \) and its translation with respect to node 3421, edge \((2341, 1342)\), are both of dimension 4.

We now define another operation on \( S_n \), namely the rotation operation, that will also be of primary importance for the construction of the \( n - 1 \) directed edge-disjoint spanning trees, and for the development of the multinode broadcasting and scattering algorithms. As emphasized in the introduction, these algorithms are designed so that messages originating at different nodes are interleaved in such a manner that no two messages contend for the same edge at any time during the execution of the algorithm. The properties of the rotation operation, as explained below, will help achieve this attribute.

**Definition 2:** Let us define a bijection \( r \) from the set \( \{1, 2, ..., n\} \) to itself as follows:

\[
r(i) = \begin{cases} 
  i, & \text{if } i = 1 \\
  (i - 1) \mod (n - 1) + 2, & \text{otherwise},
\end{cases}
\]

(Notice that \( r \) maps symbols 1 to itself and the remaining symbols are follows: \( 2 \to 3 \to ... \to n - 1 \to n \to 2 \)). The rotation of a node \( v \) of \( S_n \), is defined to be node:

\[
Ro(v) = r(v_1)r(v_n)r(v_2)...r(v_{n-1})
\]

or equivalently \( Ro(v) = v' \) so that \( v'_1 = r(v_1) \), \( 1 \leq i \leq n \). This means that the position of each symbol of \( v \) and the symbol itself are mapped through \( r \). For example, \( Ro(4123) = 2413 \) and \( Ro(1324) = 1243 \). By rotation of a network we mean that the rotation operation is applied to each node of the network.

The rotation operation is an automorphism on \( S_n \) that possesses the following properties:

1. It can be easily verified that it maps the identity node to itself, \( Ro(12...n) = 12...n \). As an extension to this, nodes \( v \) and \( Ro(v) \) are always at the same distance from the identity node.

2. Let \((v, u)\) be an edge of dimension \( i \). Then \((Ro(v), Ro(u))\) is an edge of dimension \( r(i) \). This becomes clear, if we analytically express edges \((v, u)\) and \((Ro(v), Ro(u))\) as follows:

\[
(ro(v_1), ro(v_n), ro(v_2), ..., ro(v_{n-1})) =
\]

\[
(r(v_1), r(v_n), r(v_2), ..., r(v_{n-1}), r(v_1), ..., r(v_{n-1}))
\]

Notice that edge \((Ro(v), Ro(u))\) is of dimension \( r(i) \), because from the definition of rotation the position of symbol \( r(v_i) \) in node \( Ro(v) \) is \( r(i) \). As an extension to this property, the \( n - 1 \) edges obtained after \( 0, 1, 2, ..., n - 2 \) applications of rotation on an edge of dimension \( i \), have dimensions \( i, i+1, i+2, ..., n, 2, ..., i-1 \), respectively. With this observation, we conclude that \( n - 1 \) edges, each obtained as a rotation of its preceding one, are all of different dimensions. For example, the edges of \( S_4 \), \((4123, 2413)\), \((2413, 3412)\), and \((3421, 4321)\), are obtained by consecutive applications of rotation have dimensions \( 3, 4, \) and \( 2 \), respectively.

The rotation operation on nodes of the star network has similar properties to the right cyclic shift operation on nodes of the binary hypercube \([3, 10]\).

The property of the rotation operation that the \( n - 1 \) edges each one obtained as a rotation of this preceding one are all of different dimensions, along with the property of edge dimension preservation of the translation operation will be used extensively in the development of the multinode broadcasting and scattering algorithms. These properties will help guarantee that messages originating at individual nodes will be interleaved in such a manner that no two messages will contend for the same edge at any time during the execution of the algorithm. We explain below how this attribute can be achieved.

In a multinode broadcasting or scattering algorithm, all nodes of the network are sources of messages. Under the all-port communication assumption \( n!(n - 1) \) directed edges are available on \( S_n \) for message transmission at each time step. Since the algorithm proceeds symmetrically from all nodes of the network, messages originating at each one of the \( n! \) nodes of \( S_n \) are transmitted through at most \( n - 1 \) directed edges at
each time step. Let us denote by \( E_i(12...n) \) the set of \( n - 1 \) directed edges on which messages originating at the identity node are transmitted at time step \( i \) of the algorithm. Since the algorithm proceeds symmetrically from each node of the network, the \( n - 1 \) edges on which messages originating at node \( s \) are transmitted at time step \( i \) of the algorithm. Let us denote by \( E_i(s) \) the set of \( n - 1 \) directed edges on which messages originating at node \( s \) are transmitted at time step \( i \) of the algorithm. Since the algorithm proceeds symmetrically from each node of the network, the \( n - 1 \) edges on which messages originating at node \( s \) are transmitted at time step \( i \), denoted by \( E_i(s) \), is obtained from \( E_i(12...n) \) using the operation of translation with respect to \( s \) (if \( (v, u) \in E_i(12...n) \) then \( (Tr_s(v), Tr_s(u)) \in E_i(s) \)). The following lemma is enough to guarantee that no conflicts arise during the execution of the algorithms.

**Lemma 1:** At each time step \( i \), if the \( n - 1 \) directed edges in \( E_i(12...n) \) are all of different dimensions, then the sets of directed edges \( E_i(s) \), where \( s \) ranges over all nodes of \( S_n \), are disjoint.

**Proof:** Assume two different edges \( (v, u) \neq (v', u') \in E_i(12...n) \) for some \( i \), and take the edges \( (Tr_s(v), Tr_s(u)) \in E_i(s) \) and \( (Tr_s(v'), Tr_s(u')) \in E_i(s') \) which are obtained by \( (v, u) \) and \( (v', u') \), respectively, under translation with respect to two different nodes of \( S_n \), \( s \) and \( s' \). Also assume that \( (Tr_s(v), Tr_s(u)) = (Tr_s(v'), Tr_s(u')) \). Since the dimension of an edge is preserved under translation, we conclude that \( \dim(v, u) = \dim(Tr_s(v), Tr_s(u)) = \dim(v', u') \) which contradicts our assumption that \( (v, u) \) and \( (v', u') \) are two different edges of \( E_i(12...n) \). \( \square \)

The multinode broadcasting and scattering algorithms will be developed so that at each time step \( i \), the set \( E_i(12...n) \) contains edges that are rotations of each other and as a consequence of different dimensions. According to lemma 1, this will guarantee that at each time step \( i \), the sets of edges \( E_i(s) \), where \( s \) ranges over all nodes of \( S_n \), are disjoint and as a consequence no two messages will compete for the same edge at any time step \( i \) during the execution of the algorithm.

### 3 The edge-disjoint spanning trees

In this section we construct the \( n - 1 \) directed edge-disjoint spanning trees graph, rooted at the identity node of \( S_n \), \( EDT_{12...n} \). The \( i \)th spanning tree, denoted by \( T_i \), is defined to be the tree rooted at the neighbor of \( 12...n \) over dimension \( i \), node \( i2...(i-1)1(i+1)...n \). Each spanning tree includes all nodes of \( S_n \) except the identity node.

Let us remind that a node \( v \) belongs to substar \( S^k_{n-1} \), \( 2 \leq k \leq n \), if the \( k \)th symbol of \( v \) is 1, \( v_k = 1 \). For notation purposes, we use symbol \( S^1_{n-1} \) to denote the collection of \( (n - 1)! \) isolated nodes of \( S_n \) that start with symbol 1. Furthermore, the correct position of a symbol is the position of this symbol in the identity node. Before we proceed to a more formal description of the spanning trees we need the following definition.

**Definition 3:** For each node \( v \) (excluding node \( 12...n \) and its neighbors), we define \( p \) as follows:

1. For \( v \in S^k_{n-1} \), with \( v_1 \neq k \), we define \( p = v_1 \). For example, for node \( v = 4123 \) of \( S_4 \), \( p = 4 \).

2. For \( v \in S^k_{n-1} \), with \( v_1 = k \), we define \( p \) to be the first position in \( v \), among positions \( k + 1, ..., n, 2, ..., k - 1 \), that does not include its correct symbol. For example, for node \( v = 4321 \) of \( S_4 \), \( v_p = v_2 = 3 \).

We now describe an algorithm, \( Parent(v, T_i) \), that for given node \( v \), computes the parent nodes of \( v \) in each one of the spanning trees \( T_i \), \( 2 \leq i \leq n \). In what follows, by \( p_i(v) \) denote the parent of node \( v \) in spanning tree \( T_i \).

**Parent( \( v, T_i \) )**

if \( (v_1 = 1) \) then

(1) \( p_i(v) = v_1 v_2 ... v_{i-1} 1 v_{i+1} ... v_n \)

else if \( (i \neq k \) and \( i \neq v_1 \) and \( i \neq v_p \) ) then

(2) \( p_i(v) = i v_2 ... v_n \)

else if \( (i = k) \) then

(3) \( p_i(v) = v_p v_2 ... v_{i-1} 1 v_{i+1} ... v_n \)

else if \( (i = v_1) \) then

(4) \( p_i(v) = 1 v_2 ... v_{k-1} 1 v_{k+1} ... v_n \)

else if \( (i = v_p) \) then begin

if \( (v_1 = k) \) then

(5) \( p_i(v) = 1 v_2 ... v_{k-1} 1 v_{k+1} ... v_n \)

else

(6) \( p_i(v) = k v_2 ... v_n \)

end Parent

The collection of \( n - 1 \) spanning trees constructed using the \( Parent(v, T_i) \) algorithm, constitutes the \( EDT_{12...n} \) graph. We simply define the parent of the root node of each spanning tree to be the identity node. We now need to prove that the spanning trees constructed using the \( Parent(v, T_i) \) algorithm indeed possess all the desirable properties:

**Lemma 2:** The \( Parent(v, T_i) \) algorithm defines a spanning tree, rooted at the neighbor of \( 12...n \) over dimension \( i \), node \( i2...(i-1)1(i+1)...n \), that possesses the following properties:

1. If all the edges of the spanning trees are directed from parent to children nodes, these spanning trees are edge-disjoint. Consequently, \( EDT_{12...n} \) contains all edges of \( S_n \) except those edges that are directed towards node \( 12...n \).
Therefore, all spanning trees are isomorphic, since the rotation operation is an automorphism on $S_n$. From the properties of the rotation operation, we conclude that each set of $n-1$ corresponding directed edges of the spanning trees, contains edges that are all of different dimensions.

**Proof:** A proof of this lemma can be found in [7].

The importance of the Parent($v, T_i$) algorithm is that each node $v$ can locally compute its parent in each of the spanning trees $T_i$, $2 \leq i \leq n$, with information given only by its permutation $v = v_1 v_2 ... v_n$.

Using the $EDT_{12...n}$ graph and the Parent($v, T_i$) algorithm we can easily derive an $EDT_s$, rooted at any other node $s$ of $S_n$. This graph is isomorphic to $EDT_{12...n}$ and has the same properties as this, namely, $n-1$ edge-disjoint spanning trees, node $s$ has $n-1$ node disjoint paths to each one of the other nodes, and corresponding directed edges of the spanning trees are of different dimensions. To derive $EDT_s$, we simply apply the operation of translation with respect to $s$, on $EDT_{12...n}$. If edge $(p_i(v), v)$ belongs to the $i^{th}$ spanning tree of $EDT_{12...n}$ then edge $(Tr_i(p_i(v)), Tr_i(v))$ belongs to the $i^{th}$ spanning tree of $EDT_s$ (the spanning tree rooted at the neighbor of $s$ over dimension $i$). Since the dimension of each edge is preserved under translation, these edges are of the same dimension. Each node $v$ (excluding node $s$ and its neighbors) can derive its parent in the $i^{th}$ spanning tree of $EDT_s$ by executing algorithm Parent$(Tr_i^{-1}(v), T_i)$ and obtaining the translation of the resulting node with respect to $s$.

The edges of each $EDT_s$ graph are directed from parent to children nodes. As mentioned in lemma 2, $EDT_{12...n}$ contains all edges of $S_n$ except those edges that are directed towards node $12...n$. In general, an $EDT_s$ graph contains all edges of $S_s$ except those that are directed towards node $s$. The $EDT_s$ graph will be used to achieve fault tolerance by transmitting the same messages through multiple disjoint paths. The case of the mininode broadcasting and scattering algorithms is of special interest. As pointed out in the previous section, the messages originating at individual nodes will be interleaved in such a manner that no two messages contend for the same edge at any time during the execution of the algorithm. A necessary condition in order to achieve this attribute was presented in lemma 1. Let us remind that by $E_i(s)$ we denote the set of $n-1$ directed edges on which messages originating at node $s$ are transmitted at time step $i$ of a mininode broadcasting or scattering algorithm. Since a mininode algorithm proceeds symmetrically from all nodes of the network, each $E_i(s)$ is obtained from $E_i(12...n)$ by a translation with respect to $s$. According to lemma 1, if the $n-1$ directed edges in $E_i(12...n)$ are all of dif-
ferent dimensions than the sets $E_i(s)$, for fixed $i$ (time step), and $s$ ranging over all nodes of $S_n$, are disjoint. In other words, at each time step $i$, messages originating at individual nodes are transmitted through different edges or $S_n$. By lemma 2, the $n - 1$ spanning trees of $EDT_{12}...n$ are rotations of each other, and as a consequence corresponding directed edges of the spanning trees are all of different dimensions. This property is true for any other graph $EDT_s$, since the dimension of each edge is preserved under translation. We conclude that in order to avoid conflict of messages originating at individual nodes during a multinode broadcasting or scattering algorithm, it is enough to use $n - 1$ corresponding edges of the $n - 1$ spanning trees of $EDT_{12}...n$.

4 Applications

The $EDT_s$ graph, is used to derive fault tolerant algorithms for four fundamental communication problems on interconnection networks, namely, the single-node and multinode broadcasting, and the single-node and multinode scattering. Before we proceed to a detailed description of the algorithms, we derive for each problem lower bounds for the number of time steps required for all messages to reach their destinations and for the number of message transmissions. The lower bounds are derived with respect to the communication model used (store-and-forward, all-port, unit time to transmit a message on an edge). For the fault tolerant algorithms the lower bounds are derived with respect to the approach we use to achieve fault tolerance, and for degree of fault tolerance $n - 2$. Let us remind that using this approach multiple copies of the messages are transmitted to each destination node through node disjoint paths. This offers the advantage that no prior knowledge of the faulty elements of the network is required and is not offered by other approaches.

4.1 Lower bounds

In a fault tolerant single-node broadcasting, each of the $n! - 1$ destination nodes receives $n - 1$ copies of the same group of $M$ messages from the source node, one copy through each of the $n - 1$ node disjoint paths that exist between the source node and each destination node. As a consequence, a lower bound for the number of message transmissions is $M(n - 1)(n! - 1)$. The number of time steps required for the algorithm to complete is $M + \left\lceil \frac{3(n-1)}{2} \right\rceil$, since the $M$ messages are transmitted through each of the $n - 1$ node disjoint paths that exist between the source node and each destination node.

<table>
<thead>
<tr>
<th>problem</th>
<th>time steps</th>
<th>messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNB</td>
<td>$M + \left\lceil \frac{3(n-1)}{2} \right\rceil$</td>
<td>$M(n - 1)(n! - 1)$</td>
</tr>
<tr>
<td>MNB</td>
<td>$M(n! - 1)$</td>
<td>$(n - 1)(n! - 1)n!$</td>
</tr>
<tr>
<td>SNS</td>
<td>$M(n! - 1)$</td>
<td>$(n - 1)n!t_n$</td>
</tr>
<tr>
<td>MNS</td>
<td>$Mn!t_n$</td>
<td>$(n - 1)(n! - 1)n!$</td>
</tr>
</tbody>
</table>

Table 1: Lower bounds on the star network for degree of fault tolerance $n - 2$ ($t_n = n!(n + \frac{2}{n} + H_n - 4)$).

In a fault tolerant multinode broadcasting, each node receives $M(n - 1)(n! - 1)$ messages, $n - 1$ copies of the $M$ messages originating at each one of the other $n! - 1$ nodes. Since each node of $S_n$ has $n - 1$ incident edges, a lower bound for the time steps required for this problem is $M(n! - 1)$. Each of the $n!$ nodes of $S_n$ receives $M(n - 1)(n! - 1)$ messages, and a lower bound for the number of message transmissions is $M(n - 1)(n! - 1)n!$.

In a fault tolerant single-node scattering algorithm, the source node transmits $M(n - 1)(n! - 1)$ messages, $n - 1$ copies of the $M$ messages destined to each one of the other $n! - 1$ nodes. Since each node of $S_n$ has $n - 1$ incident edges, a lower bound for the time steps required for this problem is $M(n! - 1)$. A message destined to a specific node must travel a number of edges equal to the shortest distance between that node and the source node. Therefore, a lower bound for the number of message transmissions required is the sum of the shortest distances of all nodes to the source node, multiplied by $M(n - 1)$ since each node receives $n - 1$ copies of $M$ messages from the source node. Let us denote by $N_d$ the number of nodes at distance $d$ from the source node. A lower bound for the number of message transmissions is:

$$M(n-1) \sum_{d=1}^{\left\lceil \frac{3(n-1)}{2} \right\rceil} dN_d = M(n-1)n! \left[ \sum_{d=1}^{\left\lceil \frac{3(n-1)}{2} \right\rceil} \frac{n!}{d!} dN_d \right] = M(n-1)n!(n + \frac{2}{n} + H_n - 4)$$

The quantity $\left( \sum_{d=1}^{\left\lceil \frac{3(n-1)}{2} \right\rceil} dN_d/n! \right)$ was shown in [2] to be equal to $n + \frac{2}{n} + H_n - 4$, where $H_n$ is the $n$th harmonic number $H_n = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n}$. In what follows by $t_n$ we denote the quantity $n + \frac{2}{n} + H_n - 4$.

A fault tolerant multinode scattering problem can be viewed as $n!$ separate single-node scattering problems, one from each node of $S_n$. A lower bound for the number of message transmission is derived from the lower bound for the number of message transmissions required for the single-node scattering prob-
4.2 Fault tolerant single-node broadcast

The \( EDT_s \) graph can be used to derive a fault tolerant single-node broadcasting algorithm with degree of fault tolerance \( n - 2 \). The source node \( s \) transmits the group of \( M \) messages it wishes to broadcast through all of its incident edges simultaneously. The \( M \) messages are pipelined down each one of the \( n - 1 \) directed edge-disjoint spanning trees that are rooted at the neighbors of node \( s \). As soon as a node receives a message from its parent in the \( i^{th} \) spanning tree of \( EDT_s \), it saves a copy, and forwards the message to all of its children nodes in the same spanning tree simultaneously. The result is that each destination node receives \( n - 1 \) copies of the \( M \) messages as the source node, under fault free conditions. However, if up to \( n - 2 \) faulty nodes or edges exist in the network that block the messages, it is guaranteed that each destination node will receive \( n - 1 \) messages from its parent in the \( i^{th} \) spanning tree of \( EDT_s \) and is pipelined down \( x \) different directed edge-disjoint spanning trees. As a consequence all of the \( n - 1 \) \( (x = \frac{n - 1}{2}) \) edge-disjoint spanning trees are used for message transmission. The result is that each destination node receives \( x \) copies of the \( M \) messages through \( x \) node disjoint paths from the source node, and the degree of fault tolerance is \( x - 1 \).

The number of time steps required for this algorithm is \( M(n - 1)(n! - 1) \) since each node receives \( n - 1 \) copies of the \( M \) messages, which is also optimal.

Using a similar technique, we can control the degree of fault tolerance of the single-node broadcasting algorithm, by transmitting the same group of \( M \) messages through a certain number of the directed edge-disjoint spanning trees. Assume that the required degree of fault tolerance is \( x - 1 \leq n - 2 \), meaning that each node must receive the same group of \( M \) messages through \( x \) node disjoint paths from the source node. In order to achieve maximal utilization of the network resources, the group of \( M \) messages is split into \( \frac{n - 1}{x} \) subgroups, each of size \( \frac{M}{n - 1} \). Consequently, a lower bound for the number of time steps required for this problem is \( M(n - 1)! \).

Table 1 summarizes the lower bounds on the star network for the single-node and multinode broadcasting and for the single-node and the multinode scattering problems, denoted by SNB, MNB, SNS and MNS, respectively.
spanning tree of $\text{EDT}_i$, it transmits an acknowledgement to its parent node in this spanning tree.

3. When a node $v$ receives an acknowledgement from one of its children nodes in the $i$th spanning tree of $\text{EDT}_i$, it forwards the group of $M$ messages it received in the past from its parent in this spanning tree, to its next child node in the same spanning tree. When a node receives an acknowledgement from its last child in a spanning tree, it transmits an acknowledgement to its parent node in the same spanning tree.

The algorithm terminates when each source node receives acknowledgements from all its neighbors. In the above algorithm, the transmission of messages in each $\text{EDT}_i$ correspond to a simultaneous depth first traversal of its $n-1$ spanning trees. In order to prove that using this algorithm, no two messages contend for the same edge at any time during its execution, we have to show that the requirement of lemma 1 is satisfied. Let us remind that by $E_i(s)$ we denote the group of $n-1$ directed edges of which messages originating at node $s$ are transmitted at time step $i$ of a multinode broadcasting algorithm. When an algorithm proceeds symmetrically from all nodes of $S_n$, the $n-1$ directed edges is each $E_i(s)$, are obtained as a translation with respect to $s$ of the $n-1$ directed edges that belong to $E_i(12...n)$. According to lemma 1, if at each time step $i$, the edges in $E_i(12...n)$ are all of different dimensions, then the sets of edges $E_i(s)$, for $s$ ranging over all nodes of $S_n$ are disjoint, and as a consequence messages originating at individual nodes are transmitted over disjoint sets of edges at time step $i$. The multinode broadcasting algorithm proceeds symmetrically from all nodes of $S_n$, since each $\text{EDT}_i$, is a translation with respect to $s$ of $\text{EDT}_{12...n}$. This means that, if an edge $(v, u)$ is used for transmission of a message originating at node $12...n$ during time step $i$ of the execution of the algorithm, then edge $(T_s(v), T_s(u))$ is used for the transmission of a message originating at node $s$ of $S_n$ at time step $i$. At each time step messages originating at node $12...n$ are transmitted over $n-1$ corresponding edges of the $n-1$ spanning trees rooted at the neighbors of $s$. From the properties of $\text{EDT}_{12...n}$ (lemma 2), these edges are rotations of each other and as a consequence of different dimensions, and the requirement of lemma 1 is satisfied.

The number of time steps required for this algorithm to complete is $M(n!-1)$, which is optimal. The number of message transmissions performed is $M(n-1)(n!-1)n!$, because each of the $n!$ nodes receives $n-1$ copies of the $M$ messages from each one of the other $n!-1$ nodes. This is the minimum number of message transmissions required for this problem. Finally, controlling the degree of fault tolerance is possible using a technique similar to the one described in subsection 4.2 for the fault tolerant single-node broadcasting algorithm.

4.4 Fault tolerant single-node scattering

The $\text{EDT}_i$, graph can be used to derive a fault tolerant single-node scattering algorithm with degree of fault tolerance $n-2$. The distinct groups of $M$ messages that the source node $s$ wishes to transmit to each destination node, are transmitted through all of the incident edges of $s$, and are pipelined down each one of the $n-1$ directed edge-disjoint spanning trees rooted at the neighbors of $s$. The result is that each destination node receives $n-1$ copies of the $M$ messages, one copy through each of the $n-1$ node disjoint paths from $s$, under fault free conditions.

If the transmission of messages is scheduled so that messages destined to nodes that are the furthest from the source node are transmitted first, the number of time steps required for this algorithm is $M(n!-1)$, which is optimal. This is true because the $n-1$ edges incident to the source node constitute a bottleneck for the transmission of $M(n-1)(n!-1)$ messages. The number of message transmissions performed is $O(M(n-1)!/n!)$, which is only asymptotically optimal, because the lengths of the $n-1$ node disjoint paths between each destination node and the source node, through the $\text{EDT}_i$, graph, differ from the shortest possible length by a small additive constant. Finally, controlling the degree of fault tolerance is possible using a technique similar to the one described in subsection 4.2 for the fault tolerant single-node broadcasting algorithm.

4.5 Fault tolerant multinode scattering

In a multinode scattering algorithm, each node transmits distinct groups of $M$ messages to each other node. To achieve fault tolerance, each source node transmits the $M$ messages destined to each other node, through all of the $n-1$ directed edge-disjoint spanning trees rooted at its neighbors. As in the multinode broadcasting algorithm, messages originating at individual nodes will be interleaved in such a manner that no two messages will contend for the same edge at any time during the execution of the algorithm (lemma 1). The fault tolerant multinode scattering algorithm on $S_n$ proceeds as follows:

For each node $v$ of $S_n$, except node $12...n$ do the following:
Figure 3: An instance of the fault tolerant multinode scattering algorithm on $S_3$.

1. Node 12...n transmits the $n-1$ groups of messages destined to nodes $v$, $Ro(v)$, $Ro^2(v)$, ..., $Ro^{n-2}(v)$, through spanning trees $T_2$, $T_3$, $T_4$, ..., $T_n$ of $EDT_{12...n}$, respectively.

2. The algorithm proceeds symmetrically from each node of $S_n$, and as a consequence, any other node $s$ transmits the $n-1$ groups of $M$ messages destined to nodes $Tr_s(v)$, $Tr_s(Ro(v))$, $Tr_s(Ro^2(v))$, ..., $Tr_s(Ro^{n-2}(v))$, through spanning trees $T_2$, $T_3$, $T_4$, ..., $T_n$, of $EDT_s$, respectively.

3. When the messages transmitted from node $s$ have reached their destinations, $s$ can transmit messages to another group of nodes $Tr_s(u)$, $Tr_s(Ro(u))$, $Tr_s(Ro^2(u))$, ..., $Tr_s(Ro^{n-2}(u))$, for $u \neq v$.

From the properties of $EDT_{12...n}$ graph, the $n-1$ paths that lead from node 12...n to nodes $v$, $Ro(v)$, $Ro^2(v)$, ..., $Ro^{n-2}(v)$, through spanning trees $T_2$, $T_3$, $T_4$, ..., $T_n$, respectively, are rotations of each other (lemma 2), and as a consequence, the $n-1$ directed edges at each level of these paths are of different dimensions. Each node in the path receives the $M$ messages from its parent before it starts transmitting them to the next node down the path. As a consequence, at each time step $n-1$ directed edges that are all at the same level of the paths are used. Since these edges are all of different dimensions the requirement of lemma 1 is satisfied, and no two messages contend for the same edge during the execution of the algorithm. The destination nodes of each source node for $v = 213$ of $S_3$ are marked black in Fig. 3. Notice that the edges that are used simultaneously for messages transmissions (i.e. bold face) are all different.

The number of time steps required for this algorithm to complete is $O(Mn!t_n)$ and the number of message transmissions performed is $O(M(n-1)(n!)^2t_n)$. These are both asymptotically optimal because the lengths of the $n-1$ disjoint paths between each destination node and a source node $s$, through the $EDT_s$ graph, differ from the shortest possible lengths by a small additive constant. The degree of fault tolerance can be controlled using a technique similar to the one described in subsection 4.2.

5 Conclusions

We presented several algorithms on the star interconnection network, in the areas of data communication and fault tolerance. New definitions like that of the rotation operation for nodes of $S_n$ were introduced to facilitate the construction of the $n-1$ directed edge-disjoint spanning trees. Using these trees fault tolerant algorithms, that can tolerate up to $n-2$ faulty nodes or edges, were presented for the single-node and multinode broadcasting, and for the single-node and multinode scattering problems. All of the algorithms operate under the all-port assumption and are near optimal in terms of time and number of message transmissions.

We now provide a comparison of the algorithms presented in this paper on the star network, with algorithms for the same problems, under the same communication assumptions, on the popular hypercube network. In tables 2 and 3 below the performances of the two networks are compared. Since the star network is defined for numbers of nodes which are factorials, while the hypercube is defined for powers of two, the comparison cannot be exact. In the comparison below a hypercube network with $O(n!)$ nodes and degree $O(n \log n)$ is assumed. From table 2 we notice that whenever the performance of an algorithm depends on the degree of the network, as for example the number of time steps required for the fault tolerant multinode broadcasting and single-node scattering problems, the hypercube network performs better than the star network by a factor of $\log n$. On the other hand, whenever the performance of an algorithm depends on the diameter of the network, or the lengths of the short-
est paths between nodes, as for example the number of message transmissions of the fault tolerant single-node scattering and multinode scattering problem, the star network performs better by a factor of log n. The number of time steps for the single-node broadcasting and the fault tolerant single-node broadcasting problems depend on both the degree and the diameter of the networks and this is reflected in the comparison of their performances.

<table>
<thead>
<tr>
<th>problem</th>
<th>time</th>
<th>messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNB</td>
<td>$O\left(\frac{M^x}{n} + n\right)$</td>
<td>$O(Mx!n!)$</td>
</tr>
<tr>
<td>MNB</td>
<td>$O\left(\frac{M^x}{n\log n}\right)$</td>
<td>$O(Mx!n!\log n)$</td>
</tr>
<tr>
<td>SNS</td>
<td>$O\left(\frac{M^x}{n\log n}\right)$</td>
<td>$O(Mx!n!\log n)$</td>
</tr>
<tr>
<td>MNS</td>
<td>$O(Mx!n!)$</td>
<td>$O(Mx!n!\log n)$</td>
</tr>
</tbody>
</table>

Table 2: Performance of a hypercube with $O(n!)$ nodes and degree of fault tolerance $x - 1$.

<table>
<thead>
<tr>
<th>problem</th>
<th>time</th>
<th>messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNB</td>
<td>$O\left(\frac{M^x}{n\log n} + n\log n\right)$</td>
<td>$O(Mx!n!)$</td>
</tr>
<tr>
<td>MNB</td>
<td>$O\left(\frac{M^x}{n\log n}\right)$</td>
<td>$O(Mx!(n\log n)^2)$</td>
</tr>
<tr>
<td>SNS</td>
<td>$O\left(\frac{M^x}{n\log n}\right)$</td>
<td>$O(Mx!n!\log n)$</td>
</tr>
<tr>
<td>MNS</td>
<td>$O(Mx!n!)$</td>
<td>$O(Mx!(n\log n)^2)$</td>
</tr>
</tbody>
</table>

Table 3: Performance of a star network with $n!$ nodes and degree of fault tolerance $x - 1$.

Acknowledgements

We wish to thank Prof. Henk Meijer for several suggestions that improved the presentation of this manuscript. This research was supported by the Telecommunications Research Institute of Ontario and the Natural Sciences and Engineering Research Council of Canada.

References


