Technical Report No. 93-346

OPTIMAL COMMUNICATION ALGORITHMS ON STAR GRAPHS USING SPANNING TREE CONSTRUCTIONS

by
Paraskevi Fragopoulou and Selim G. Akl

Department of Computing & Information Science
Queen's University
Kingston, Ontario, Canada

February 1993

This research was supported by the Telecommunications Research Institute of Ontario and by the Natural Sciences and Engineering Research Council of Canada.

Abstract

In this paper we consider three fundamental communication problems on the star interconnection network: the problem of simultaneous broadcasting of the same message from every node to all other nodes, or *multinode broadcasting*, the problem of a single node sending distinct messages to each one of the other nodes, or *single node scattering*, and finally the problem of each node sending distinct messages to every other node, or *total exchange*. All of these problems are studied under two different assumptions: the assumption that each node can transmit a message of fixed length to one of its neighbors and simultaneously it can receive a message of fixed length from one of its neighbors (not necessarily the same one) at each time step, or *single link availability* (SLA), and the assumption that each node can exchange messages of fixed length with all of its neighbors at each time step, or *multiple link availability* (MLA). In both cases communication is assumed to be bidirectional. The cases where the originating processor wishes to send only one or more than one message to each one of the other processors are distinguished when necessary. Lower bounds are derived for these problems under the stated assumptions, and optimal algorithms are designed in terms both of time and number of message transmissions. Although the algorithms derived for the first two problems require the minimum amount of the above resources, the algorithm designed for the total exchange problem is optimal only to within a multiplicative factor. All the communication algorithms presented in this paper are based on the construction of spanning trees with special properties on the star graph to fit different communication needs. A special framework is developed to facilitate the construction of these trees. The scheduling disciplines that lead to optimal results in each case are described.

Key words and phrases: communication algorithm, interconnection network, parallel algorithm, spanning tree, star graph.

1 Introduction

When algorithms execute on a parallel computer, processors are often required to exchange information. It is well known that the overhead associated with this interprocessor communication is the major drawback of parallel computers in which processors are linked by an interconnection network. Surprisingly, the communication problems arising during the execution of many algorithms are not arbitrary but fall into certain categories that define communication patterns. It is important to find ways to efficiently execute those communication patterns. In this paper we are concerned with three major communication problems for the star interconnection network, namely the multinode broadcasting, the single node scattering and the total exchange problems. *Multinode broadcasting* is the problem of simultaneous broadcasting of the same message from every node to all other nodes, *single node scattering* is the problem of a single node sending distinct messages to each one of the other nodes and *total exchange* is the problem of each node sending distinct messages to every other node. All of these problems are studied under the following two major assumptions: 1) the *Single Link Availability*, or SLA assumption, meaning that in one time step a node can send a message of fixed length over one of its incident links and at the same time it can receive a message of fixed length from one of its incident links (not necessarily the same link), and 2) the *Multiple Link Availability*, or MLA assumption, meaning that in one time step a node can send (receive) messages of fixed length to (from) all of its incident links simultaneously. Information is transferred along the links of the network in packets. The size of all the packets is equal to one and it takes unit time to transmit a packet over a link. Communication is assumed to be bidirectional meaning that in one time step one packet can be transmitted along a link in each direction. Unless otherwise specified, in all of the algorithms presented, it is assumed that each node has one packet to send to every other node. The multinode broadcasting, single node scattering and total exchange problems are respectively the same as the *all-to-all broadcasting*, *one-to-all* and *all-to-all personalized communication* problems, as addressed in [16] for the hypercube network. Also the single link and multiple link availability assumptions are respectively the same as the *one-port* and *all-port* communication models [16].

A common approach to implement communication algorithms on interconnection networks is to embed spanning trees with special properties on those networks. The root of the tree is the origin of the messages. The links of the embedded tree are used for message transmission. All of the algorithms presented in this paper are based on the construction of spanning trees with special properties and the use of appropriate scheduling disciplines to achieve optimal results.

The three communication problems described above are not new, and have been previously studied for a number of interconnection networks such as the hypercube [11, 16]. However it is the first time they are considered on the star graph (defined below). The multinode broadcasting problem on the star graph under the assumption that messages of arbitrary length can be exchanged between two adjacent processors at each time step has been studied in [9, 20]. Another class of spanning trees, called *edge-disjoint* spanning trees, that reduce the communication time of the single node broadcasting problem on the star network and offer many applications in the area of fault tolerant communication algorithms have been constructed in [14]. The problem of generalized routing on star graphs was first addressed in [23]. Other communication algorithms on the star graph can be found in [1, 7, 9, 20, 25, 26, 30].

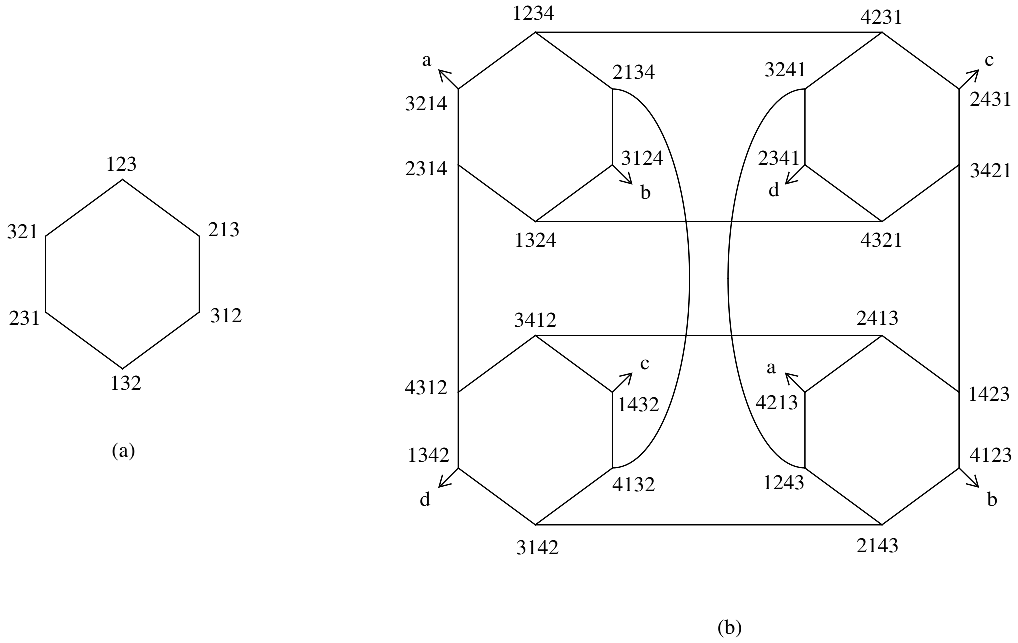

Figure 1: The 4-star: 4 interconnected 3-stars

The n-star graph, denoted by S_n, has $n!$ nodes. Each of the nodes is labeled with a different permutation of n distinct symbols (without loss of generality, we henceforth use the set of symbols $\{1, 2, ..., n\}$ to label the nodes of S_n). Each node of S_n is connected to those nodes that can be obtained by interchanging its first with its i^{th} symbols, $2 \leq i \leq n$. In this way, every node is an endpoint of $n-1$ links as shown in Fig.1 [1]. As shown in [1, 2], S_n enjoys a number of properties desirable in interconnection networks. These include node and link symmetry, maximal fault tolerance, and strong resilience. Because of its symmetry, the graph is easily extensible, can be decomposed in various ways and allows for simple routing algorithms [24, 28]. The graph was shown to be Hamiltonian in [4, 17, 22], and efficient algorithms for sorting [21] and Fourier transform computation [12, 13], were developed on it. Various other algorithms have been developed on the star graph in [5, 6, 8, 27, 29]. In addition S_n is superior to C_n (the n-cube) [3] with respect to two key properties: degree (number of links at each node), and diameter (maximum distance between any two nodes) [2]. The degree of S_n is $n-1$, i.e. *sublogarithmic* to the number of its nodes while a hypercube with $\Theta(n!)$ nodes has degree $\Theta(\log n!) = \Theta(n \log n)$, i.e. *logarithmic* to the number of its nodes. The same can be said for the diameter of S_n which is $\lfloor \frac{3(n-1)}{2} \rfloor$. The star as well as the hypercube networks belong to the family of Cayley graphs. It is also known that S_n can be decomposed into n subgraphs S_{n-1} by fixing each different symbol in one particular position 2 to n [1]. If we fix a specific symbol in the last position we observe that there are $(n-1)!$ nodes that constitute an S_{n-1}. For example if the symbol in the last position is held fixed with any symbol, then we get $(n-1)!$ nodes (i.e. an S_{n-1}) for every one of the n symbols. Thus the nodes of the S_n can be partitioned into n groups each containing $(n-1)!$ nodes and each being isomorphic to S_{n-1} as shown in Fig.1. If this decomposition is recursively applied to the resulting substars, S_n can be decomposed into $n!/k!$ substars S_k, $1 \leq k \leq n-1$.

This paper is organized as follows. In section 2 lower bound are derived for all of the above problems under the stated assumptions. Section 3 describes two separate algorithms for the multinode broadcasting problem under the MLA assumption. The first of these is based on the construction of Hamiltonian paths on the star graph. The second algorithm uses a technique previously developed for the hypercube interconnection network in [11]. Due to the completely different structure of the star graph, however, the implementation is nontrivial. It also leads to the interesting observation that although the technique developed in [11] was motivated by specific properties of the hypercube topology, it can actually be used in other networks, having little or no resemblance to the hypercube. In section 4 the multinode broadcasting problem under the SLA assumption is solved using a Hamiltonian cycle. In section 5, we develop two algorithms for the single node scattering problem under the MLA assumption. The technique used in the first algorithm resembles partially the one developed in [11] for the hypercube and uses all the definitions and proofs presented in section 2 to make the technique applicable to the star graph. In the second algorithm a simpler technique is presented that leads to asymptotically optimal results. In section 6 an algorithm is given for the single node scattering problem under the SLA assumption. In section 7 and 8 we give algorithms for the total exchange problem under the MLA and the SLA assumptions respectively. Finally, a summary of the results obtained in this paper is presented in section 9 along with some suggestions for further research.

2 Lower bounds

Multinode broadcasting on an interconnection network is the problem where each node wishes to send the same message to all other nodes. In a multinode broadcasting algorithm on S_n, each node must receive a total of $n!-1$ messages, one from each one of the other nodes. As a consequence the minimum number of message transmissions required for the algorithm to complete is $n!(n!-1)$. Under the SLA assumption $n!$ links are available at each time step. This means that the minimum time required for the algorithm to complete is $n!-1$. Under the MLA assumption all $n!(n-1)$ links can be used for message transmissions at each time step. This means that the minimum time required for the algorithm to complete is $\lceil \frac{n!-1}{n-1} \rceil$.

Single node scattering on an interconnection network is the problem where a specific node wishes to send a different message to each one of the other nodes. In a single node scattering algorithm on S_n, $n!-1$ different messages must be transmitted by the origin of messages. Under the SLA assumption the origin can transmit only one message at each time step and the minimum time required for the algorithm to complete is $n!-1$. However under the MLA assumption all the $n-1$ links incident to the origin of the messages can be used simultaneously at each time step and as a consequence the minimum time required for the algorithm to complete is $\lceil \frac{n!-1}{n-1} \rceil$. The number of message transmissions required can be found as follows: A message destined for a specific node must travel as many links as the shortest distance from the origin to this node. If we sum the shortest distances from the origin to each node, this will be the minimum number of message transmissions required for this problem:

$$\sum_{k=1}^{\lfloor \frac{3(n-1)}{2} \rfloor} k|D_k| = n! \frac{\sum_{k=1}^{\lfloor \frac{3(n-1)}{2} \rfloor} k|D_k|}{n!} = n!d$$

$|D_k|$ is the number of nodes at a distance k from the origin and d has been shown to be [2]:

$$d = n + \frac{2}{n} + H_n - 4$$

H_n is the n^{th} harmonic number: $H_n = 1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n}$. As a consequence the minimum number of message transmissions required for a single node scattering algorithm on S_n is:

$$n!(n + \frac{2}{n} + H_n - 4)$$

Total exchange on an interconnection network is the problem where each node wishes to send a distinct message to every other node, in other words, every possible pair of nodes exchange distinct messages. The total exchange algorithm is equivalent to $n!$ different single node scattering algorithms, one from each one of the nodes in S_n. As a consequence the minimum number of message transmissions required is $(n!)^2(n + \frac{2}{n} + H_n - 4)$. Under the SLA assumption $n!$ links are available at each time step. This means that the minimum time required for the algorithm to complete is $n!(n + \frac{2}{n} + H_n - 4)$. Under the MLA assumption $n!(n-1)$ links can be used for message transmission at each time step simultaneously. This means that the minimum time required for the algorithm to complete is $\lceil \frac{n!(n+\frac{2}{n}+H_n-4)}{n-1} \rceil$.

If we denote by t_n the quantity $n!(n + \frac{2}{n} + H_n - 4)$, then the above lower bounds are summarized in the table below:

problem	number of transmissions	time(SLA)	time(MLA)
multinode broadcasting	$n!(n!-1)$	$n!-1$	$\lceil \frac{n!-1}{n-1} \rceil$
single node scattering	t_n	$n!-1$	$\lceil \frac{n!-1}{n-1} \rceil$
total exchange	$n!t_n$	t_n	$\lceil \frac{t_n}{n-1} \rceil$

The algorithms derived here for all of the above problems are optimal in terms of time and number of message transmissions. The methods used in this section to derive lower bounds for the communication problems under consideration are straightforward and similar to the methods used in [11] to derive lower bounds for the same problems on the hypercube network.

3 Multinode broadcasting under the MLA assumption

We will present two different ways to embed spanning trees on the star graph that both lead to optimal multinode broadcasting algorithms under the MLA assumption, in terms both of time and of number of message transmissions. The first such spanning tree is based on the construction of Hamiltonian paths on the star graph. The second is based on a specific technique for tree construction that has been previously used for other interconnection networks (i.e. the hypercube), and that can be modified for the star graph.

Before we proceed to the construction of the multinode broadcasting algorithm we list the characteristics that a broadcasting algorithm from a single node of the star graph should have. We choose this node to be node $12...n$. We then show how this broadcasting algorithm can be replicated to each other node of the star graph and the conditions that are necessary to ensure that the concurrent execution of broadcasting algorithms from all nodes of the star graph constitutes an optimal multinode broadcasting algorithm. The broadcasting algorithm from node $12...n$ must have the following characteristics:

1. The message from node $12...n$ must reach all nodes of S_n.

2. To minimize the number of message transmissions a node must not receive the same message more than once.

3. To guarantee that the multinode broadcasting algorithm obtained terminates in $\lceil \frac{n!-1}{n-1} \rceil$ time steps, at each time step the message (from node $12...n$) must reach $n-1$ new nodes.

4. The $n-1$ nodes that receive the message (from node $12...n$) at the same time step must be directly connected to nodes that have received this message at a previous time step.

Definition 1: The *type* of a link (x, y) of S_n, denoted by $t(x, y)$, $2 \leq t(x, y) \leq n$, is defined to be the position of the symbol that we need to exchange with the first symbol of node x in order to get node y.

Definition 2: Consider a node $q = q_1 q_2 ... q_n$ of the star graph. We define F_q, the *translation with respect to q*, as the bijection from the set of the $n!$ permutations of the symbols $\{1, 2, ..., n\}$ to itself:

$$F_q(x) = q \cdot x$$

(this operation is often referenced as *permutation composition*). For example, if $q = 312$ then $F_q(213) = 132$. By *translation of a graph with respect to q* we mean that each node of the graph is translated with respect to q. The *inverse translation with respect to q*, denoted by F_q^{-1}, is defined as:

$$F_q^{-1}(x) = q^{-1} \cdot x$$

Let us denote by $L_i(q)$ the set of links on which the message from node q is transmitted at time step i, $1 \leq i \leq \lceil \frac{n!-1}{n-1} \rceil$, of the algorithm. For each i, $L_i(q)$ is obtained from $L_i(12...n)$ by translation under q (this means that if $(x, y) \in L_i(12...n)$ then $(F_q(x), F_q(y)) \in L_i(q)$).

Theorem 1: If the sets of links $L_i(12...n)$, $1 \leq i \leq \lceil \frac{n!-1}{n-1} \rceil$, form a single node broadcasting algorithm from node $12...n$, then the sets of links $L_i(q)$, $1 \leq i \leq \lceil \frac{n!-1}{n-1} \rceil$, define a single node broadcasting algorithm from node q of the star graph.

Proof: We prove that if $(x, y) \in L_i(12...n)$ is a link then $(F_q(x), F_q(y)) \in L_i(q)$ is also a link. This is an immediate consequence of the fact that the translation operation induces an automorphism of a Cayley graph. More analytically this becomes obvious if we write (x, y) and $(F_q(x), F_q(y))$ as:

$$(x_1 x_2 ... x_{i-1} x_i x_{i+1} ... x_n,\ x_i x_2 ... x_{i-1} x_1 x_{i+1} ... x_n)$$

$$(q_{x_1} q_{x_2} ... q_{x_{i-1}} q_{x_i} q_{x_{i+1}} ... q_{x_n},\ q_{x_i} q_{x_2} ... q_{x_{i-1}} q_{x_1} q_{x_{i+1}} ... q_{x_n})$$

Clearly if (x, y) is a link then $(F_q(x), F_q(y))$ is also a link of the same type. Since the spanning tree rooted at node $12...n$ covers all the nodes of the star graph and F_q is a bijection, the spanning tree rooted at node q also covers all the nodes of the star graph. $\square$

This theorem guarantees that if a single node broadcasting algorithm from node $12...n$ exists (in other words link sets $L_i(12...n)$, $1 \leq i \leq \lceil \frac{n!-1}{n-1} \rceil$, are specified) then the link sets $L_i(q)$ for all i, $1 \leq i \leq \lceil \frac{n!-1}{n-1} \rceil$, that form a single node broadcasting algorithm from node q, can be easily derived, for all nodes q of S_n.

However our objective is to create a multinode broadcasting algorithm. If we can guarantee that the broadcasting algorithms from all nodes of S_n can be executed simultaneously conflict free, then our objective is achieved. In other words we want for each i, $1 \leq i \leq \lceil \frac{n!-1}{n-1} \rceil$, the sets $L_i(q)$ where q ranges over all the nodes of the star graph to be disjoint. The following theorem gives a necessary condition so that the above is true (this theorem is proved in [11] for the hypercube; here we show that it also holds for the star graph, the proof we use is similar).

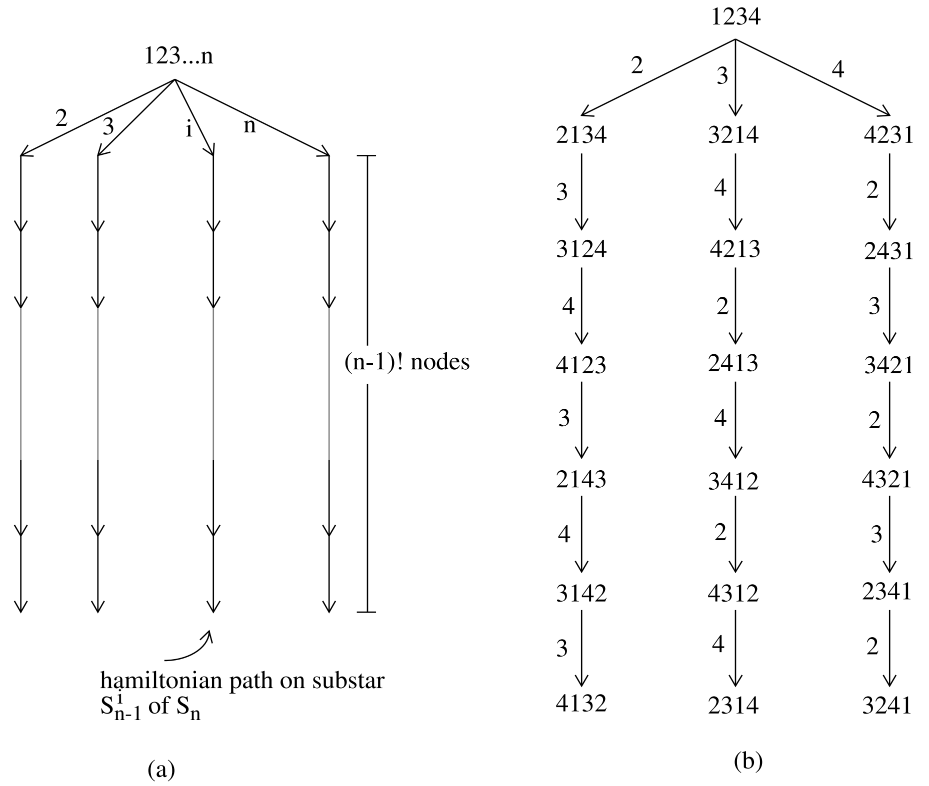

Figure 2: Multinode broadcasting under the MLA assumption (algorithm 1): (a) Sp anning tree construction (b) Spanning tree construction on S_4

Theorem 2: If the links in each $L_i(12...n)$ are all of different types, then for each i, the sets $L_i(q)$, where q ranges over all possible permutations of symbols $\{1, 2, ..., n\}$, are disjoint.

Proof: Assume two different links $(x, y) \neq (x', y') \in L_i(12...n)$ for some i, and take the links $(F_q(x), F_q(y)) \in L_i(q)$ and $(F_{q'}(x'), F_{q'}(y')) \in L_i(q')$ which are obtained by (x, y) and (x', y') respectively using translation under two different nodes of S_n, q and q'. Also assume that $(F_q(x), F_q(y)) = (F_{q'}(x'), F_{q'}(y'))$. Since the link type is preserved under translation (theorem 1), this means that $t(x, y) = t(F_q(x), F_q(y)) = t(F_{q'}(x'), F_{q'}(y')) = t(x', y')$ which contradicts our assumption that (x, y) and (x', y') are two different links of $L_i(12...n)$. $\Box$

3.1 Algorithm 1

We are now ready to describe a multinode broadcasting algorithm under the MLA assumption. The algorithm is composed of two phases. The first phase of the algorithm is based on the construction of a spanning tree of depth $(n-1)!$ rooted at each node of the star graph. The tree rooted at node $x_1x_2...x_n$ is responsible for transferring the message broadcast from this node to all other nodes except those that start with symbol x_1. The second phase of the algorithm completes the broadcasting of information. The two phases of the algorithm are described separately in what follows.

3.1.1 Phase 1

We first need some definitions.

Definition 3: It is well known that each permutation can be defined by its *cycle notation*. In a cycle notation each symbol's position is that occupied by the next symbol in the cycle (cyclically) [18]. Cycles have two or more symbols, singletons are not considered to be cycles. For example the cycle notation of 341526 is (13)(245).

Definition 4: Two cycle notations are said to be *equivalent* if they correspond to the same permutations.

Definition 5: Let us define the function:

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

with domain and range $\{1, 2, ..., n\}$. We define R the cycle rotation of a node x so that for every cycle $(x_{i_1}x_{i_2}...x_{i_j})$ that belongs to the cycle notation of x, cycle $(r(x_{i_1})r(x_{i_2})...r(x_{i_j}))$ belongs to the cycle notation of $R(x)$. By $R^j = R \circ R^{j-1}$ we denote j applications of cycle rotation. By *cycle rotation of a graph* we mean that cycle rotation is applied to each node of the graph.

Theorem 3: If x and y represent nodes of S_n, and x' and y' are produced by x and y respectively under cycle rotation then:

1. If x and y are connected then x' and y' are also connected (in other words connectivity is preserved under cycle rotation in S_n).

2. If (x, y) (the link connecting nodes x and y) is a link of type $t(x, y)$ then (x', y') is a link of type $t(x', y') = (t(x, y) - 1)\text{mod}(n-1) + 2$.

Proof: See Appendix A. □

We are now ready to describe the construction of the depth $(n-1)!$ spanning tree rooted at node $12...n$ of the star graph and spanning all other nodes except those starting with symbol 1. This tree will follow all the appropriate requirements so that the tree rooted at any other node q can be obtained under translation with respect to q.

There are $n-1$ links of types $2, 3, ..., n$ leaving node $12...n$. Let T_i, $2 \le i \le n$, denote the subtree rooted at the node connected to $12...n$ over the link of type i. Subtree T_i is defined to be a Hamiltonian path of length $(n-1)!$ spanning the nodes of that substar S_{n-1} of S_n, which results if we fix symbol 1 at the i^{th} position; this substar will be denoted by S^i_{n-1} in what follows (see Fig.2(a)). Since there are many algorithms to construct Hamiltonian paths on the star graph [4, 17, 22] this seems to be a very easy approach to follow.

Let us recall that $L_i(q)$ denotes the set of links on which a message from node q is transmitted at time step i of the algorithm. In order to guarantee that the sets $L_i(q)$, where q ranges over all the nodes of the star graph, are disjoint and as a consequence the single node broadcasting algorithms can proceed conflict free from all nodes of S_n simultaneously, we must make sure that for each i all links in $L_i(12...n)$ are of different types (theorem 2). The maximum number of links each set $L_i(q)$ can have is $n-1$, since there are only $n-1$ different link types.

Assume that T_2, rooted at node $213...n$, has been constructed using a Hamiltonian path algorithm. Each T_j, $3 \le j \le n$, is obtained from T_2 under $j-2$ cycle rotations. This guarantees that for each i, the links in

$L_i(12...n)$ are all of different types (theorem 3). The following theorem is necessary to complete the spanning tree construction.

Theorem 4: If T_2 is a Hamiltonian path starting at node $213...n$ and spanning all nodes of S^2_{n-1}, then T_j, $3 \le j \le n$, obtained by T_2 under $j-2$ cycle rotations, is a Hamiltonian path starting at node $j23...(j-1)1(j+1)...n$ and spanning all nodes of S^j_{n-1}.

Proof: Since T_2 contains only links of types $\{3, 4, ..., n\}$, T_j will contain only links of types $\{2, 3, ..., (j-1), (j+1), ..., n\}$ (theorem 3). Starting at node $j23...(j-1)(j+1)...n$, and following links with types $\{2, 3, ..., (j-1), (j+1), ..., n\}$ only, symbol 1 stays fixed at position j. In other words all the nodes visited belong to S^j_{n-1}.

We now have to show that T_j visits all nodes of S^j_{n-1}, or in other words no node in S^j_{n-1} is visited more than once. There is an one-to-one and onto function between the link types of T_2 and those of T_j, $3 \le j \le n$, which is the following:

$$t_j(i) = (i + j - 4)\mathrm{mod}(n-1) + 2, \qquad 3 \le j \le n, \ \ 2 \le i \le n$$

Since the star graph is link symmetric and the nodes traversed by path T_2 are distinct this implies that the nodes traversed by each other path T_j are also distinct. This completes the proof that T_j is a Hamiltonian path for S^j_{n-1}. □

In what follows we refer to this tree by *spanning tree based on rotated Hamiltonian paths.* Subtree T_i of this spanning tree is a Hamiltonian path on nodes of S^i_{n-1} and each subtree is a cycle rotation of its previous one cyclically.

The same tree can be reproduced under translation (theorem 1) so that it is rooted at every node of the star graph and all trees can be simultaneously used at each time step without collision. We now describe how these trees can be used by allowing each node to store the minimum amount of information for the structure of the trees. All nodes must use all of the $n-1$ available links they have at each time step. Since no collisions arise during the execution of the algorithm, at each time step i, node x belongs to $n-1$ different spanning trees rooted at $n-1$ different nodes of the star graph. Let y and z be two of those nodes. If $T_j(y)$ and $T_k(z)$ are the specific subtrees of the spanning trees rooted at nodes y and z, respectively, to which x belongs, then $j \ne k$. In other words, if T_2 is specified and in time step i of the algorithm a node forwards a message received by links i_{inp} to link i_{out} then for each j, $3 \le j \le n$, it also forwards a message received by $t_j(i_{inp})$ to $t_j(i_{out})$.

Since there are $(n-1)(n-1)!$ nodes that the broadcast message must reach at this first phase of the algorithm, the time required is $(n-1)!$. An example of a spanning tree on S_4 can be seen in Fig.2(b). At this point each node $x_1x_2...x_n$ has received all broadcast messages except those from nodes that start with symbol x_1. This is taken care of by the second phase of the algorithm described below.

3.1.2 Phase 2

Each node $x_1x_2...x_n$ must now receive $(n-1)!-1$ additional messages coming from nodes that start with symbol x_1. Each node $x_1x_2...x_n$ is connected to $n-1$ other nodes $x_ix_2...x_{i-1}x_1x_{i+1}...x_n$ that have received all messages coming from nodes that start with symbol x_1. So the messages that $x_1x_2...x_n$ must receive

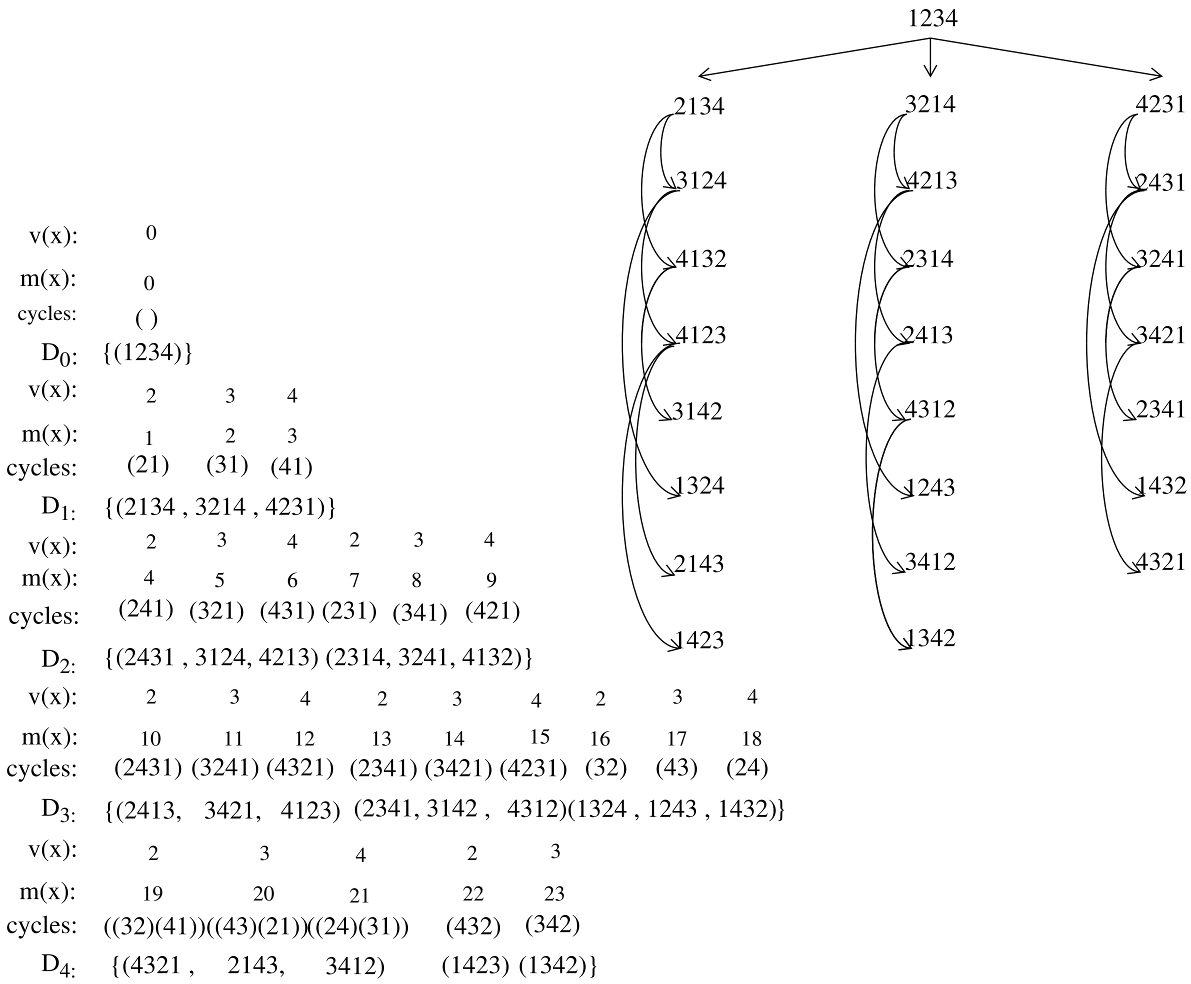

Figure 3: Multinode broadcasting under the MLA assumption (algorithm 2): Spanni ng tree construction on S_4

should be received uniformly in $\lceil\frac{(n-1)!-1}{n-1}\rceil$ time steps from all of its $n-1$ incident links. A rule that uniformly distributes the messages over the $n-1$ links is the following:

Rule 1: Node $x_1x_2...x_n$ receives from its neighbor with first symbol x_i, a message broadcast from node $x_1 * x_i$, where $x_i \in \{1,2,...,n\} - \{x_1\}$, and $*$ represents any permutation of the $(n-2)!$ symbols $\{1,2,...,n\} - \{x_1, x_i\}$.

Using this rule node x receives $(n-2)!$ messages over each one of its incident links, except link with type n over which it receives $(n-2)!-1$ messages. Since there are $(n-1)!-1$ messages that are uniformly received by each node over $n-1$ links this step requires $\lceil\frac{(n-1)!-1}{n-1}\rceil$ time. So the total time required by the algorithm is $(n-1)! + \lceil\frac{(n-1)!-1}{n-1}\rceil = \lceil\frac{n!-1}{n-1}\rceil$. The number of message transmissions required by the algorithm is $n!(n!-1)$. This means that this algorithm is optimal in terms of time and number of message transmissions.

3.2 Algorithm 2

Before describing the construction of the spanning tree we need to establish some facts.

Definition 6: A group of nodes for which one is derived from the other under a cycle rotation in called a *necklace* (the term necklace was initially used in [19] for similar groups of nodes in the shuffle-exchange graph).

Theorem 5: A necklace has at most $n-1$ distinct nodes.

Proof: A node is mapped to itself after $n-1$ applications of cycle rotation. This is simple to show because $r^{n-1}(i) = i$, for all $i \in \{1, 2, ..., n\}$ and the cycle notations of nodes x and $R^{n-1}(x)$ are the same. However we say *at most* $n-1$ because an equivalent cycle notation can result after only $j < n-1$ cycle rotations. For example node 14325 of S_4, with cycle notation (24), is mapped to its self only after 2 (and not $n-1=4$) cycle rotations, because the node obtained after 2 cycle rotations has cycle notation (42) which is equivalent to (24). The size of a necklace is a divisor of $n-1$. □

Theorem 6: Each node that does not start with symbol 1 belongs to a necklace that has exactly $n-1$ nodes.

Proof: Each node that does not start with symbol 1 has a cycle that includes symbol 1. This cycle is mapped to itself only after $n-1$ applications of R since 1 does not change under cycle rotation. □

Theorem 7: Let x' be obtained by applying R to node x of S_n ($x' = R(x)$) then, the distance of x' from node $12...n$ equals the distance of x from node $12...n$.

Proof: In what follows we use c_x and s_x to denote the number of cycles in the cycle notation of node x, and the number of symbols that belong to these cycles, respectively (see definition 3).

If node x starts with symbol 1, then its distance from node $12...n$ is: $c_x + s_x$ [1]. If x starts with a symbol other than one, then its distance from node $12...n$ is $c_x + s_x - 2$. If x' is obtained from x by application of a cycle rotation then x' has the same cycle structure as x and we conclude that $c_x = c_{x'}$ and $s_x = s_{x'}$. Also if x starts with symbol 1 then x', also starts with symbol 1. To see this, note that since 1 does not belong to any of the cycles in x, it does not belong to any of the cycles in x' either, because any symbol in $\{2, ..., n\}$ is mapped to symbols in $\{2, ..., n\}$ through r (see definition 5), and x' starts with a symbol other than 1 as well. If x starts with a symbol other than 1, which means that symbol 1 belongs to the cycle notation of x, then 1 also belongs to the cycle notation of x', since 1 is mapped to itself through r (see definition 5). The reason r maps 1 to itself is to guarantee that distances between nodes of S_n are preserved under cycle rotation. That was necessary since nodes that start with symbol 1 and nodes that do not, have different formulas to express the distance. □

Given a star graph node, there are many different ways to proceed to another node that is closer to node $12...n$. In each case one of the following rules can be applied:

Rule 2: For nodes that start with symbol 1, there is more than one way to proceed. If the number of symbols contained in the cycle notation of the node is k, then there are k different ways: move 1 to any of the k positions not occupied by their correct symbols.

Rule 3: For nodes that start with a symbol other than 1 and have only one cycle in their cycle notation, there is only one way to proceed: move the symbol in the first position to its proper position.

Rule 4: For nodes that start with a symbol other than 1 and have more than one cycle, there are two different ways to proceed:

(a) Move the symbol in the first position to its proper position. In this case either the number of cycles remains the same and the number of symbols that belong to the cycles is reduced by one, or the number of cycles is reduced by one and the number of symbols that belong to the cycles is reduced by two but symbol 1 comes to the first position of the node. The last case arises when there is a cycle of the form $(x, 1)$ with two symbols only and we move x to its proper position.

(b) If there are k symbols that belong to cycles that do not include symbol 1 then we can move along any link defined by these symbols. This reduces the number of cycles by one (two cycles are merged into one) without reducing the number of symbols in the cycles.

At this point we have described the necessary framework for the construction of the spanning tree that satisfies all the requirements to form an optimal multinode broadcasting algorithm. The algorithm presented here is motivated by the one presented in [11] for the hypercube interconnection network. An important contribution of this work is to show that the same technique can be adapted to construct spanning trees on the star graph which has a structure that is fundamentally different from that of the hypercube.

The nodes of S_n are grouped into $\lfloor\frac{3(n-1)}{2}\rfloor$ different sets D_k, $0 \le k \le \lfloor\frac{3(n-1)}{2}\rfloor$. Set D_k contains all nodes of S_n that are at distance k, $0 \le k \le \lfloor\frac{3(n-1)}{2}\rfloor$, from node $12...n$. The nodes in each set D_k are further grouped into subsets $N_{k1}, N_{k2}, ..., N_{kl_k}$. Each N_{kj} is a different necklace. For $1 \le k \le n-1$, N_{k1} is the necklace produced by the node that has cycle notation $(k+1, k, ..., 2, 1)$; for $n \le k \le \frac{3n-4}{2}$, and n even, and for $n \le k \le \frac{3n-5}{2}$, and n odd, we pose the restriction that N_{k1} is the necklace produced by the node that has cycle notation $(n, n-1)(n-2, n-3)...(i, i-1)...(..., 2, 1)$. This cycle notation can be seen as follows: arrange all the n symbols in reverse order and group in cycles $k+1-n$ pairs of symbols from left to right. The symbols remaining to the right are grouped into a single cycle. The distance from this node to node $12...n$ is: $c + s - 2 = k + 2 - n + n - 2 = k$. We name the nodes corresponding to these cycle notations *generator nodes* (this corresponds to the nodes used in [19] to distinguish one node of a necklace from nodes in the shuffle-exchange graph). It is obvious from the way the first necklace in each D_k was created that it includes exactly $n-1$ nodes (theorem 6). The necessity for these restrictions will become apparent later.

Each node is now associated with a distinct number $m(x)$ in the order: $(12...n)N_{11}N_{21}N_{22}N_{31}...$ $N_{3l_3}...N_{k1}...N_{kl_k}...N_{\lfloor\frac{3(n-1)}{2}\rfloor 1}...N_{\lfloor\frac{3(n-1)}{2}\rfloor l_{\lfloor\frac{3(n-1)}{2}\rfloor}}$. We further associate each node with number $v(x)$ so that: $v(x) = [(m(x)-1)\mathrm{mod}(n-1)] + 2$. It is clear that $0 \le m(x) \le n! - 1$ and $2 \le v(x) \le n$ (except of $v(12...n)$ which is 0). The nodes are now cyclically shifted within each necklace so that the following are satisfied:

1. For necklaces that have nodes that do not start with symbol 1 and do not form N_{k1}, for any k, we distinguish between the following cases:

   (a) If the nodes have only one cycle then the first node x is chosen so that the first symbol in the node is $v(x)$ (rule 3).

   (b) If the nodes have more than one cycles then the first node x is chosen so that either the first symbol in the node is $v(x)$ (rule $4(a)$), or $v(x)$ belongs to one of the cycles of the node that does not include symbol 1 (rule $4(b)$).

2. For necklaces that include nodes that start with symbol 1 the first node x is chosen so that $v(x)$ belongs to one of the cycles of the node (rule 2).

3. For necklaces that form N_{k1} we distinguish between two cases:

   (a) For $k \leq n-1$ nodes in N_{k1} have only one cycle. The generator node of this necklace is moved to position with $v(x)$ equal to the first symbol of the node (rule 3).

   (b) For $k \geq n$ nodes in N_{k1} have more than one cycles. The rightmost cycle always includes symbol 1. The generator node of this necklace is moved to position with $v(x)$ the smallest symbol in the second cycle from the right (rule 4(b)). This has the effect of merging the two rightmost cycles into a single cycle with its symbols sorted from left to right.

The rest of the nodes within each necklace are arranged so that each node is obtained by its preceding one (cyclically) under a single cycle rotation.

There are now $\lceil \frac{n!-1}{n-1} \rceil$ clusters of $n-1$ nodes each, which have $v(x)$'s equal to $2, 3, ..., n$ in sequence. We name them C_i, $1 \leq i \leq \lceil \frac{n!-1}{n-1} \rceil$. C_i is the set of nodes that receive a message broadcast from node $12...n$ at time step i of the algorithm. The set of links connected to nodes of cluster C_i that have types $2, 3, ...n$ respectively are the ones used for message transmission at time step i of the algorithm. Since the set of $n-1$ links used at time step i of the algorithm are all of different types, the tree can be replicated using translation at any other node of S_n, and all trees can be used simultaneously conflict free to form a multinode broadcasting algorithm (theorem 2). Let the spanning tree obtained by this construction be denoted by $MBST$ (to stand for Multinode Broadcast Spanning Tree).

Theorem 8: Spanning tree $MBST$ satisfies the requirements of a multinode broadcasting algorithm because:

1. All $n-1$ links that are used for message transmission at each step of the algorithm are of different types.

2. Node x' produced by x following a link of type $v(x)$ is one link closer to node $12...n$; this guarantees that the path from the root of the tree to x has minimum length.

3. If x' is produced by x following a link of type $v(x)$ then $m(x) - m(x') \geq n-1$; this guarantees that x' receives the message before x.

Proof: See Appendix B. □

An example of a spanning tree on S_4 is shown in Fig.3.

4 Multinode broadcasting under the SLA assumption

A multinode broadcasting algorithm under the SLA assumption is easy to construct on any interconnection network if this has been proven to be Hamiltonian, which means that a Hamiltonian cycle can be constructed on this network. The star graph has been proven to be Hamiltonian and several Hamiltonian cycles have been constructed on it [4, 17, 22]. This method has been used before to construct multinode broadcasting algorithms under the SLA assumption on other interconnection networks such as the hypercube [10].

Suppose that a Hamiltonian cycle of length $n!$ has been constructed on S_n and a specific direction has been defined on it so that each node i knows its next and previous nodes on the Hamiltonian cycle, i_{next}

and i_{prev}, respectively, with respect to the predefined direction. In the first time step of the algorithm each node i transmits the message it wants to broadcast to node i_{next}. In each subsequent time step each node i transmits the message it received in the previous time step from node i_{prev} to node i_{next}.

The algorithm completes in $n! - 1$ steps. This is true because the message initiated at step 1 by node i, has traversed $n! - 1$ links down the Hamiltonian cycle after $n! - 1$ steps and has reached node i_{prev}. The number of message transmissions required is $n!(n! - 1)$ since at each time step one message is send by each of the $n!$ nodes. Therefore the algorithm is optimal in terms of time and number of message transmissions.

5 Single node scattering under the MLA assumption

We will work towards the construction of a spanning tree with some special properties that lead to an optimal single node scattering algorithm under the MLA assumption. Let r be the node wishing to transmit messages. Assume that T is the desirable tree rooted at node r, and T_i is the subtree of T rooted at the neighbor of r over the link of type i. The rule that r uses to transmit information to the nodes of the subtrees is the following:

Rule 5: Send messages to all subtrees simultaneously. Send messages to nodes that are the furthest from the root first. Break ties arbitrarily.

The time required for the algorithm to complete if rule 5 is used equals the number of nodes in the largest of the subtrees T_i, $2 \leq i \leq n$. In order for the algorithm to be optimal the largest subtree T_i must have at most $\lceil \frac{n!-1}{n-1} \rceil$ nodes. In other words T must be as balanced as possible, meaning that the difference in the number of nodes of any two subtrees of the root should not be more than one. In addition in order for the algorithm to need the minimum number of message transmissions, each path from r to any node must be as short as possible, i.e. T must be a shortest path tree (often referenced as breadth first tree). If we construct a spanning tree rooted at node $12...n$ with these properties, each other tree rooted at node q could be obtained using translation with respect to q.

We will present two different techniques to construct trees with the above characteristics on the star graph. Below, we describe each of these techniques separately.

5.1 Algorithm 1

As mentioned earlier the key to creating an optimal single node scattering algorithm on the star graph under the MLA assumption is to built a balanced, shortest path spanning tree rooted at the node that wishes to scatter the messages. We show how a tree rooted at node $12...n$ and having these characteristics can be obtained. The following two rules describe how the parent of each node is defined:

Rule 6: Nodes that do not start with symbol 1, and have only one cycle in their cycle notation, are connected to their parents following the link defined by their first symbol (rule 3).

Rule 7: Nodes that do not start with symbol 1, and have more than one cycle in their cycle notation, are connected to their parents following the link defined by one of the symbols belonging to a cycle that does not contain symbol 1 (rule 4(b)).

By following these rules the parent of each node is one link closer to node $12...n$ than the node itself (theorem 8), to guarantee that the path from the root of the tree to each node has minimum length. In

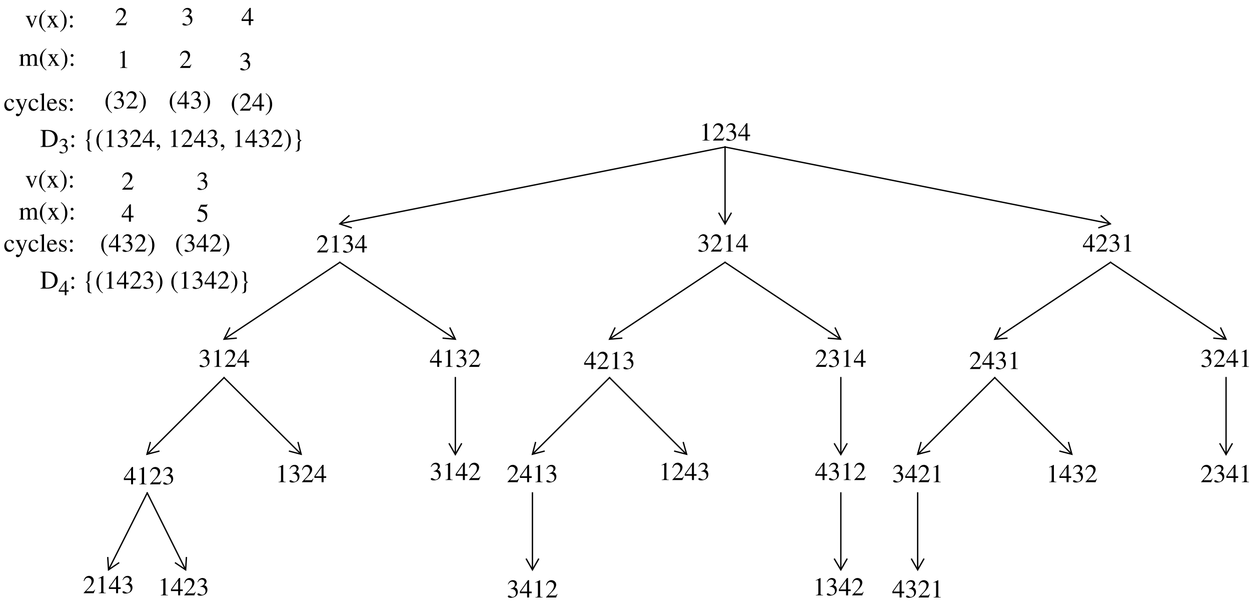

Figure 4: Single node scattering under the MLA assumption (algorithm 1): Spanning tree construction on S_4

addition if a node belongs to some substar S^i_{n-1} of S_n, its parent belongs to the same substar and all nodes of S^i_{n-1}, $2 \leq i \leq n$ belong to subtree T_i of the root.

At this point T is a balanced shortest path tree that spans all nodes of S_n except those starting with symbol 1. The only problem now is to find a way to equally distribute all the nodes that start with symbol 1 among the subtrees, so that T becomes a balanced, shortest path tree that spans all nodes of S_n.

The technique described in subsection 3.2 for all nodes of the star graph will now be applied only to nodes that start with symbol 1. The nodes of S_n starting with symbol 1 are grouped into different sets D_k, $3 \leq k \leq \lfloor \frac{3(n-1)}{2} \rfloor$. Set D_k contains all nodes of S_n, staring with symbol 1, at distance k from node $12...n$. The nodes in each set D_k are further grouped into subsets $N_{k1}, N_{k2}, ..., N_{kl_k}$. Each N_{kj} is a different necklace. The nodes within each necklace are arranged so that each node is obtained by its preceding one (cyclically) under a single cycle rotation. Each node is associated with a distinct number $m(x)$ in the order: $N_{31}, N_{41}, ..., N_{4l_4}, ..., N_{k1}, ..., N_{kl_k}, ..., N_{\lfloor \frac{3(n-1)}{2} \rfloor 1}, ..., N_{\lfloor \frac{3(n-1)}{2} \rfloor l_{\lfloor \frac{3(n-1)}{2} \rfloor}}$. We define for each node the number $v(x) = [(m(x)-1)\text{mod}(n-1)] + 2$. Clearly, $1 \leq m(x) \leq (n-1)! - 1$ and $2 \leq v(x) \leq n$. We rotate the nodes within each N_{kj} so that a node x has $v(x)$ that belongs to the cycle notation of the node (rule 2). Node x' obtained by x after following a link of type $v(x)$ is one link closer to node $12...n$ (theorem 8). Since all nodes start with symbol 1, starting from x and following a link of type $v(x)$ the resulting node x' belongs to that subgraph S_{n-1} of S_n having symbol 1 fixed at position $v(x)$, or $S^{v(x)}_{n-1}$, and also to subtree $T_{v(x)}$ of T. From the way the $v(x)$'s were created there are at most $\lceil \frac{(n-1)!-1}{n-1} \rceil$ nodes for each possible value of $v(x) \in \{2, 3, ..., n\}$. This means that the $(n-1)!-1$ nodes that start with symbol 1 are distributed as equally as possible among subtrees T_i, $2 \leq i \leq n$, and as a consequence T is as balanced as possible. An example of a tree for a single node scattering algorithm under the MLA assumption on S_4 is shown in Fig.4.

5.2 Algorithm 2

The shortest path, balanced spanning tree obtained by Algorithm 1 has the disadvantage that it is difficult to define the parent and children functions for nodes that start with symbol 1. We now describe a shortest path tree which is balanced to within a constant factor (meaning that the ratio of the number of nodes between any pair of subtrees of the root is not more than a constant) and for which we can define the parent and children functions in a straightforward way. We start with some definitions.

Definition 7: The *generator node* of a necklace is defined as follows:

1. For necklaces that contain nodes that start with a symbol other than 1, node $x = x_1 x_2 ... x_n$ with $x_2 = 1$ is the generator node.
2. For necklaces that contain nodes that start with symbol 1, the generator node can be found as follows: take the cycle in each node that contains symbol 2 (if this exists) and bring it to the form in which 2 is the leftmost symbol. Pick as generator node the one that has the smallest such cycle in lexicographic order.

Definition 8: The *displacement* of a node x in a necklace, denoted by $d(x)$, is the number of cycle rotations required to obtain node x from the generator node of this necklace. (This notion was initially introduced in [19] for nodes of the shuffle-exchange graph.)

Definition 9: The *period* of a node x, denoted by $p(x)$, is the number of nodes contained in the necklace x belongs to.

Definition 10: The i^{th} *ordering*, $2 \leq i \leq n$, of symbols $1, 2, 3, ..., n$ is the ordering $\prec_i$ so that: $1 \prec_i i \prec_i i+1 \prec_i ... \prec_i n \prec_i 2 \prec_i ... \prec_i i-1$.

Definition 11: Given a sequence of numbers, a *left minimum* is a number that is the smallest among all the numbers to its left in the sequence.

Definition 12: The *canonical cycle notation* of a node that starts with a symbol other than 1, and has more than one cycle in its cycle notation, is defined as follows: for a node that has symbol 1 in the i^{th} position, rotate the symbols within each cycle so that the smallest symbol according to the i^{th} ordering is the rightmost symbol of the cycle. Then arrange the cycles in decreasing ordering (again according to the i^{th} ordering) of the rightmost symbols in the cycles.

We are now ready to define a shortest path spanning tree, balanced to within a constant factor, and rooted at node $12...n$ of S_n.

Definition 13: A shortest path spanning tree, balanced to within a constant factor, and rooted at node $12...n$ of S_n, is defined by its parent and children functions:

$$\text{parent}(i, 12...n) = \begin{cases} \emptyset, & \text{if } i = 12...n, \\ i' = (i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n), & \text{if } i_1 = 1 \text{ and } d(i) = d(i') = j-2, \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n, & \text{if } i_1 \neq 1, i \text{ has one cycle and } i_1 = j, \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n, & \text{if } i_1 \neq 1, i \text{ has more than one cycles and } j \text{ is the rightmost symbol of the second cycle from the left in the canonical cycle notation of } i \text{ (this move has the effect of merging the two rightmost cycles).} \end{cases}$$

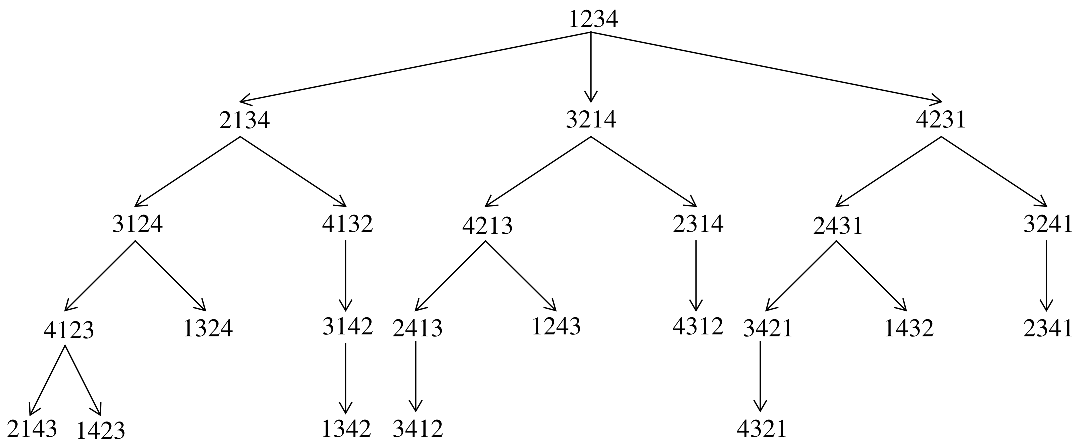

Figure 5: Single node scattering under the MLA assumption (algorithm 2): Spanning tree construction on S_4

$$\text{children}(i, 12...n) = \begin{cases} \emptyset, & \text{if } i_1 = 1, \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n,\ 2 \leq j \leq n, & \text{if } i = 12...n, \\ i' = (i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n), & \text{if } i_j = 1 \text{ and } d(i) = d(i') = j-2, \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n, & \forall j \geq 2 : i_j = j\ (i_1 \neq 1), \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n, & \text{$\forall\ j \geq 2$ which belong to the cycle of i that contains symbol 1 ($i_1 \neq 1$), and is left minimum according to the k^{th} ordering (when $i_k = 1$), if this cycle is written in the form that has symbol 1 in the rightmost position (symbols that are cyclically adjacent to symbol 1 are excluded).} \end{cases}$$

It is easy to see that these two functions are consistent.

Theorem 9: The tree of definition 13 has the following properties:

1. All nodes of S_n^i, $2 \leq i \leq n$, belong to the $(i-1)^{th}$ subtree of the root.

2. All nodes with displacement d belong to the $(d+1)^{th}$ subtree of the root.

3. All nodes that start with symbol 1 are leaves.

4. It is a shortest path tree.

5. The tree is balanced to within a constant factor.

Proof: Parts $1, 2, 3$, and 4 of the theorem can be trivially proved from the way the parent and children functions are defined. Part 5 can be proved as follows: If we had only nodes that belong to full necklaces in the tree then this would be perfectly balanced. This is because each of the $n-1$ nodes of a full necklace belongs to a different subtree. The nodes that create the imbalance are the ones that belong to nonfull necklaces. As we have already proven only nodes that start with symbol 1 can belong to nonfull necklaces (theorems 5, 6). If we take the extreme case in which all nodes that start with symbol 1 belong to the first subtree, then the first subtree contains all the nodes of S_{n-1}^1 and S_{n-1}^2, i.e. $2(n-1)!-1$ nodes, and the j^{th}, $2 \leq j \leq n-1$ subtree contains all the nodes of S_{n-1}^{j+1}, i.e. $(n-1)!$ nodes, which means that even in this extreme case the tree is balanced to within a constant factor. In reality the imbalance is much smaller among the subtrees. □

In order to achieve optimal time for a single node scattering algorithm under the MLA assumption, we must have a tree which is as balanced as possible. If we use the shortest path, balanced tree, described in subsection 5.1, the size of the largest subtree is $\lceil \frac{n!-1}{n-1} \rceil$ and optimal time is achieved. However, if we use the shortest path tree, balanced to within a constant factor, which is easier to describe through the parent and children functions, optimal time can be achieved to within a constant factor, i.e. $O(\lceil \frac{n!-1}{n-1} \rceil)$, since there is a constant imbalance among the subtrees. In both cases the minimum number of message transmissions is achieved since we have shortest path trees. The scheduling discipline used in this algorithm for the transmission of messages is defined by rule 5.

This tree can be defined at any other node r of the star graph as follows:

Definition 14: A shortest path tree, balanced to within a constant factor, and rooted at an arbitrary node r of S_n, is defined by its parent and children functions. If we define by $c = F_r^{-1}(i)$ and $c' = F_r^{-1}(i')$, the inverse translation with respect to r of i and i' respectively, then:

$$\text{parent}(i,r) = \begin{cases} \emptyset, & \text{if } i = r, \\ i' = (i_j i_2 \ldots i_{j-1} i_1 i_{j-1} \ldots i_n), & \text{if } c_1 = 1 \text{ and } d(c) = d(c') = j-2, \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n, & \text{if } c_1 \neq 1,\ c \text{ has one cycle and } c_1 = j, \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n, & \text{if } c_1 \neq 1,\ c \text{ has more than one cycles and } j \text{ is the rightmost} \\ & \text{symbol of the second cycle from the left in the canonical} \\ & \text{cycle notation of } c. \end{cases}$$

$$\text{children}(i,r) = \begin{cases} \emptyset, & \text{if } c_1 = 1, \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n,\ 2 \leq j \leq n, & \text{if } i = r, \\ i' = (i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n), & \text{if } c_j = 1 \text{ and } d(c) = d(c') = j-2, \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n, & \forall j \geq 2 : c_j = j\ (c_1 \neq 1), \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n, & \forall\ j \geq 2 \text{ which belong to the cycle of } c \text{ that contains} \\ & \text{symbol 1 } (c_1 \neq 1), \text{ and is left minimum according to} \\ & \text{the } k^{th} \text{ ordering (when } c_k = 1), \text{ if this cycle is written} \\ & \text{in the form that has symbol 1 in the rightmost posi-} \\ & \text{tion (symbols that are cyclically adjacent to symbol} \\ & \text{1 are excluded).} \end{cases}$$

It is easy to see that these two functions are consistent. A shortest path, balanced to within a constant factor tree on S_4 is shown in Fig.5

In a single node scattering algorithm under the MLA assumption, things change slightly when the origin of the messages r wants to transmit M distinct messages to each one of the other nodes. In this case although a minimum number of message transmissions can be achieved using either of the above trees, since these are shortest path trees, none of these can lead to optimal results in terms of time. If we use the shortest path tree that is as balanced as possible, then certain subtrees have one node more than others and as a consequence the root will have to transmit M more messages to these subtrees. So the time required is $M\lceil \frac{n!-1}{n-1} \rceil$. If we use the shortest path tree, balanced to within a constant factor, the time required for the algorithm to complete is even larger, namely $O(M\lceil \frac{n!-1}{n-1} \rceil)$. This means that optimality is achieved only to within a constant factor.

The minimum time we can achieve in this case is $\lceil\frac{M(n!-1)}{n-1}\rceil$. The only tree that can lead to optimal results is the shortest path, perfectly balanced tree with repeated nodes defined as follows:

Definition 15: A shortest path, perfectly balanced tree with repeated nodes, rooted at node $12...n$ of S_n, is defined by its parent and children functions:

$$\text{parent}(i, 12...n) = \begin{cases} \emptyset, & \text{if } i = 12...n, \\ i' = (i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n), & \text{if } i_1 = 1 \text{ and } d(i') = d(i) + k * p(i) = j - 2, \text{ for some} \\ & k : 0 \leq k \leq \frac{n-1}{p(i)} - 1, \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n, & \text{if } i_1 \neq 1,\ i \text{ has one cycle and } i_1 = j, \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n, & \text{if } i_1 \neq 1,\ i \text{ has more than one cycles and } j \text{ is the} \\ & \text{rightmost symbol of the second cycle from the left in} \\ & \text{the canonical cycle notation of } i. \end{cases}$$

$$\text{children}(i, 12...n) = \begin{cases} \emptyset, & \text{if } i_1 = 1, \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n,\ 2 \leq j \leq n, & \text{if } i = 12...n, \\ i' = (i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n), & \text{if } i_j = 1 \text{ and } d(i) = d(i') + k * p(i') = j - 2, \\ & \text{for some } k : 0 \leq k \leq \frac{n-1}{p(i')} - 1, \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n, & \forall j \geq 2 : i_j = j\ (i_1 \neq 1), \\ i_j i_2 ... i_{j-1} i_1 i_{j+1} ... i_n, & \forall\, j \geq 2 \text{ which belong to the cycle of } i \text{ that con-} \\ & \text{tains symbol 1 } (i_1 \neq 1), \text{ and is left minimum} \\ & \text{according to the } k^{th} \text{ ordering (when } i_k = 1), \\ & \text{if this cycle is written in the form that has} \\ & \text{symbol 1 in the rightmost position (symbols} \\ & \text{that are cyclically adjacent to symbol 1 are} \\ & \text{excluded).} \end{cases}$$

It is easy to see that these two functions are consistent.

Theorem 10: The tree in definition 15 has the following properties:

1. All nodes of S^i_{n-1}, $2 \leq i \leq n$, belong to the $(i-1)^{th}$ subtree.
2. Nodes that belong to necklaces with period p appear $\frac{n-1}{p}$ times in the tree. Since a node can appear in the tree more than once (it has more than one parent nodes) we can say that in reality we have a directed graph.
3. Nodes with displacement d that belong to necklaces with period p belong to the subtrees $ip + (d+1)$ for $0 \leq i \leq \frac{n-1}{p} - 1$.
4. All nodes that start with symbol 1 are leaves.
5. It is a shortest path tree.
6. It is a perfectly balanced tree.
7. Subtree i can be obtained from subtree $(i-1)$ under a cycle rotation.

Proof: This theorem can be trivially proved from the way the parent and children functions of the tree are defined. □

In this case all subtrees have exactly the same number of nodes, and one node can belong to more than one subtree. If one node belongs to k different subtrees then $\frac{M}{k}$ of the messages destined to this node are

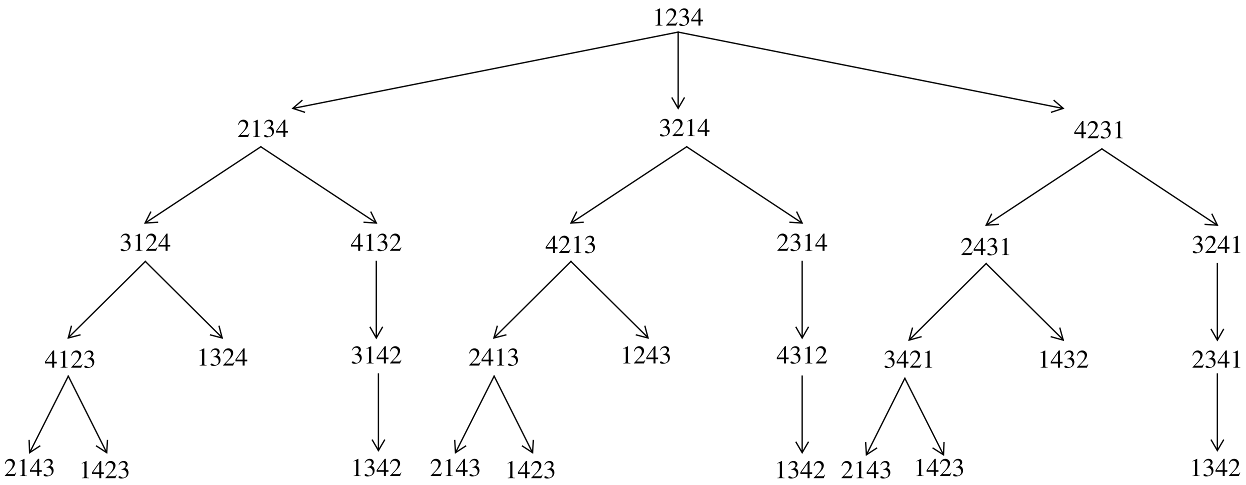

Figure 6: Single node scattering under the MLA assumption when M messages must be communicated to each node: Spanning tree construction on S_4

transferred over the links of each subtree. In order for $\frac{M}{k}$ to be integer for each $k = \frac{n-1}{p}$, where p is the period of any necklace, M must be a multiple of $n-1$. If this tree is used, the messages are evenly distributed to all subtrees and the time required for the algorithm to complete is $\lceil \frac{M(n!-1)}{n-1} \rceil$ which is optimal.

The tree can be defined at any other node of the star graph as follows:

Definition 16: A shortest path, perfectly balanced tree with repeated nodes, and rooted at an arbitrary node r of S_n, is defined by its parent and children functions. If we define by $c = F_r^{-1}(i)$ and $c' = F_r^{-1}(i')$, the inverse translation with respect to r of i and i' respectively, then:

$$\text{parent}(i,r) = \begin{cases} \emptyset, & \text{if } i = r, \\ i' = (i_j i_2 \ldots i_{j-1} i_1 i_{j-1} \ldots i_n), & \text{if } c_1 = 1 \text{ and } d(c') = d(c) + k * p(c) = j-2, \text{ for some} \\ & k: 0 \le k \le \frac{n-1}{p(c)} - 1, \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n, & \text{if } c_1 \neq 1, c \text{ has one cycle and } c_1 = j, \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n, & \text{if } c_1 \neq 1, c \text{ has more than one cycles and } j \text{ is the rightmost} \\ & \text{symbol of the second cycle from the left in the canonical,} \\ & \text{cycle notation of } c. \end{cases}$$

$$\text{children}(i,r) = \begin{cases} \emptyset, & \text{if } c_1 = 1, \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n, \ 2 \le j \le n, & \text{if } i = r, \\ i' = (i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n), & \text{if } c_j = 1 \text{ and } d(c) = d(c') + k * p(c') = j-2, \text{ for} \\ & \text{some } k: 0 \le k \le \frac{n-1}{p(c')} - 1, \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n & \forall j \ge 2 : c_j = j \ (c_1 \neq 1), \\ i_j i_2 \ldots i_{j-1} i_1 i_{j+1} \ldots i_n, & \forall\, j \ge 2 \text{ which belong to the cycle of } c \text{ that contains} \\ & \text{symbol 1 } (c_1 \neq 1), \text{ and is left minimum according} \\ & \text{to the } k^{th} \text{ ordering (when } c_k = 1), \text{ if the cycle is} \\ & \text{written in the form that has symbol 1 in the right-} \\ & \text{most position (symbols that are cyclically adjacent} \\ & \text{to symbol 1 are excluded).} \end{cases}$$

It is easy to see that these two functions are consistent. A shortest path, perfectly balanced tree with repeated nodes on S_4 is shown in Fig.6

6 Single node scattering under the SLA assumption

Any of the shortest path trees constructed in section 5 can be used for a single node scattering algorithm, if the SLA assumption is followed. In this case each message follows the shortest path to its destination since we have a shortest path tree and as a consequence the algorithm is again optimal in terms of message transmissions. The time optimality is achieved if the messages are transmitted according to the following rule:

Rule 8: All messages to subtree T_{i-1} are transmitted before any message to subtree T_i. Within a subtree, messages to nodes that are the furthest from the root are transmitted first.

The last message to subtree T_j is transmitted at time $NT_j - 1$, where $NT_j = \sum_{i=1}^{j} |T_i|$, is the sum of the nodes in subtrees T_1 to T_j. As a consequence the time required for the algorithm to complete is $\sum_{i=1}^{n-1} |T_i| = n! - 1$ and the algorithm is time optimal.

The spanning trees constructed for the single node scattering algorithm on the star graph can also be used for the reverse problem, namely the *single node gathering*. This is the problem where a specific node must receive a distinct message from each one of the other nodes. The time and communication resources required for the single node scattering and gathering problems are the same. The only difference is that the flow of information is reversed from the children to the parents.

7 Total exchange under the MLA assumption

We construct a total exchange algorithm under the MLA assumption using the shortest path tree, balanced to within a constant factor, and described in subsection 5.2. Although the algorithm will achieve the minimum possible number of message transmissions the time will be optimal only to within a constant factor.

We assume that the shortest path tree, balanced to within a constant factor, is replicated at each node of S_n. In each time step at most $n-1$ different links of each tree rooted at each different node can be used for message transmission. Let us denote by $L_i(q)$ the set of $n-1$ links of the tree rooted at node q that are used for message transmission at time step i of the algorithm. In order to guarantee that for each i, and for all nodes q of S_n, the sets $L_i(q)$ are disjoint, the links in each $L_i(q)$ must be of different types (theorem 2). The rule used for the transmission of messages is:

Rule 9: Each node transmits messages destined to nodes of the same necklace simultaneously. Messages destined to necklaces that are the closest to the root are transmitted first. When one group of messages reaches its destination, another group is sent from the root.

Using this rule, the links in each $L_i(q)$ are all of different types, since the paths that lead from a specific node to the nodes of a necklace are cycle rotations of each other. If we denote by h the total number of necklaces, and by d the quantity $\frac{\sum_{k=1}^{\lfloor \frac{3(n-1)}{2} \rfloor} k|D_k|}{n!} = n + \frac{2}{n} + H_n - 4$ ($|D_k|$ is the number of nodes at a distance k from the origin, then the time required for this algorithm is: $h \cdot d \leq [2(n-1)! - 1] \cdot d = O(\lceil \frac{t_n}{n-1} \rceil)$ (by t_n we denote the quantity $n!(n + \frac{2}{n} + H_n - 4)$). In other words the algorithm is optimal only to within a constant factor.

In the case where each processor wants to transmit M distinct messages to each one of the other processors, the only tree that can give optimal results is the shortest path, perfectly balanced tree with repeated nodes. A node that belongs to a necklace with period p, belongs to $k = \frac{n-1}{p}$ different subtrees and receives $\frac{M}{k}$ of the messages from a specific node through each of the subtrees. The scheduling discipline used is the following:

Rule 10: The root sends messages destined to the nodes of each necklace simultaneously. Messages destined to necklaces close to the root are transmitted first. If each node must receive M different messages from a specific node, then the root sends $\frac{Mp}{n-1}$ groups of messages to a necklace with period p.

Under this scheduling discipline the sets of links $L_i(q)$ for each i and each node q of S_n are all of different types. This can become clear, if we notice that the paths connecting the root of a tree with the nodes of a necklace are all cycle rotations of each other. The time and the number of message transmissions achieved using this algorithm are $\lceil \frac{Mt_n}{n-1} \rceil$ and $Mn!t_n$ respectively, which are optimal.

8 Total exchange under the SLA assumption

The only requirement to achieve an optimal total exchange algorithm under the SLA assumption is a shortest path spanning tree. Any of the shortest path spanning trees defined in the previous section can be used. At each time step messages are transmitted along all links of the same type in S_n. If we use the shortest path tree, balanced to within a constant factor, and traverse the link types of the shortest path to each destination node, then this sequence of link types constitutes an optimal total exchange algorithm under the SLA assumption. This algorithm is optimal since each message travels along a shortest path from its source node to its destination node and in each time step the maximum number of messages is transmitted.

9 Conclusions

Tables 1, 2 and 3 summarize the problems solved in this paper, the time and number of message transmissions achieved for each of them, and the method that leads to optimal results in each case.

<table>
<tr><th></th><th>1-message</th><th>M-messages</th></tr>
<tr><td>SLA</td><td>Method: Hamiltonian cycle
Time: $n! - 1$
Messages: $n!(n! - 1)$</td><td>Method: Hamiltonian cycle, M times
Time: $M(n! - 1)$
Messages: $Mn!(n! - 1)$</td></tr>
<tr><td rowspan="2">MLA</td><td>Method: Spanning tree based on rotated Hamiltonian paths
Time: $\lceil \frac{n!-1}{n-1} \rceil$
Messages: $n!(n! - 1)$</td><td rowspan="2">Method: Spanning tree based on rotated Hamiltonian paths, M times
Time: $\lceil \frac{M(n!-1)}{n-1} \rceil$
Messages: $Mn!(n! - 1)$</td></tr>
<tr><td>Method: Spanning tree based on technique from [11]
Time: $\lceil \frac{n!-1}{n-1} \rceil$
Messages: $n!(n! - 1)$</td></tr>
</table>

Table 1: Multinode broadcasting algorithms

	1-message	M-messages
SLA	**Method:** Shortest path spanning tree **Time:** $n! - 1$ **Messages:** t_n	**Method:** Shortest path spanning tree, M times **Time:** $M(n! - 1)$ **Messages:** Mt_n
MLA	**Method:** Shortest path, as balanced as possible spanning tree **Time:** $\lceil \frac{n!-1}{n-1} \rceil$ **Messages:** t_n	
	Method: Shortest path, almost balanced spanning tree **Time:** $O(\lceil \frac{n!-1}{n-1} \rceil)$ **Messages:** t_n	**Method:** Shortest path, perfectly balanced spanning tree **Time:** $\lceil \frac{M(n!-1)}{n-1} \rceil$ **Messages:** Mt_n

Table 2: Single node scattering algorithms

	1-message	M-messages
SLA	**Method:** Shortest path spanning tree **Time:** t_n **Messages:** $n!t_n$	**Method:** Shortest path spanning tree, M times **Time:** Mt_n **Messages:** $Mn!t_n$
MLA	**Method:** Shortest path, almost balanced spanning tree **Time:** $O(\lceil \frac{t_n}{n-1} \rceil)$ **Messages:** $n!t_n$	**Method:** Shortest path, perfectly balanced spanning tree **Time:** $\lceil \frac{Mt_n}{n-1} \rceil$ **Messages:** $Mn!t_n$

Table 3: Total exchange algorithms

As can be seen, the total exchange algorithm under the MLA assumption (and when only one message needs to be transmitted by each node to each other node) is optimal only to within a constant factor. Since communication algorithms should be as efficient as possible, it is an interesting open problem to find an algorithm that terminates in exactly $\lceil \frac{t_n}{n-1} \rceil$ communication steps. Further, all of the above algorithms could be investigated under different communication models [15]. The assumption that one message needs one time step to be transmitted is the simplest possible. More realistic communication models assume a linear cost model where one message needs t_c time to be transmitted and there is an overhead of τ to transmit M messages between two adjacent nodes. In other words, one communication cycle lasts $t = \tau + Mt_c$ time. Finally, all of the algorithms presented in this paper could be extended to become fault tolerant.

We now provide a comparison of the algorithms presented in this paper for the three communication problems under consideration on the star network, with algorithms for the same problems, under exactly the same assumptions, on the popular hypercube network [10]. Tables 4 and 5 below give the number of message transmissions and the communication time required for each of the problems for the S_n and the hypercube network of dimension k, denoted by C_k, respectively. Table 4 is from [10].

problem	number of transmissions	time(SLA)	time(MLA)
multinode broadcasting	$n!(n!-1)$	$n!-1$	$\lceil \frac{n!-1}{n-1} \rceil$
single node scattering	t_n	$n!-1$	$\lceil \frac{n!-1}{n-1} \rceil$
total exchange	$n!t_n$	t_n	$\lceil \frac{t_n}{n-1} \rceil$

Table 4: Star network of dimension n

problem	number of transmissions	time(SLA)	time(MLA)
multinode broadcasting	$2^k(2^k-1)$	2^k-1	$\lceil \frac{2^k-1}{k} \rceil$
single node scattering	$k2^{k-1}$	2^k-1	$\lceil \frac{2^k-1}{k} \rceil$
total exchange	$k2^{2k-1}$	$k2^{k-1}$	2^{k-1}

Table 5: Hypercube network of dimension k

In table 6 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 $O(\log n!) = O(n \log n)$ is assumed.

problem	number of transmissions		time(SLA)		time(MLA)	
	S_n	C_n	S_n	C_n	S_n	C_n
multinode broadcasting	$O((n!)^2)$	$O((n!)^2)$	$O(n!)$	$O(n!)$	$O(\frac{n!}{n})$	$O(\frac{n!}{n \log n})$
single node scattering	$O(n!n)$	$O(n!n \log n)$	$O(n!)$	$O(n!)$	$O(\frac{n!}{n})$	$O(\frac{n!}{n \log n})$
total exchange	$O((n!)^2 n)$	$O((n!)^2 n \log n)$	$O(n!n)$	$O(n!n)$	$O(n!)$	$O(n!)$

Table 6: Comparison of star and hypercube performances

From table 6 we notice that whenever the performance of an algorithm depends on the degree of the network, such as the communication times of the multinode broadcasting and the single node scattering algorithms under the MLA assumption, 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 shortest paths between nodes, as for example the number of message transmissions for the single node scattering and the total exchange algorithms, the star network performs better by a factor of $\log n$. The communication time of the total exchange algorithm under the MLA assumption depends on both the degree and the diameter, and the performance is asymptotically the same for both networks. In any other case the performance of both networks is the same for all the problems under consideration. However we should not forget that the star network has smaller degree resulting in processors with a smaller number of ports and as a consequence smaller cost.

Acknowledgements

The authors wish to thank Prof. Henk Meijer, Dr. Ke Qiu and the anonymous referees for several suggestions that improved the quality of this manuscript.

References

[1] S.B. Akers, D. Harel, and B. Krishnamurthy, "The Star Graph: An Attractive Alternative to the Hypercube", in *Proceedings of the International Conference on Parallel Processing*, St. Charles, IL, pp. 393-400, 1987.

[2] S.B. Akers, and B. Krishnamurthy, "A Group Theoretic Model for Symmetric Interconnection Networks", *IEEE Transactions on Computers*, vol. 38, no. 4, pp. 555-566, 1989.

[3] S. G. Akl, *The Design and Analysis of Parallel Algorithms,* Prentice Hall, Englewood Cliffs, NJ, 1989.

[4] S.G. Akl, J. Duprat, and A. Ferreira, "Building Hamiltonian Circuits and Paths in Star Graphs", Technical Report no. 90-22, Laboratoire de l' Informatique du Parallélism, Ecole Normale Superiéure de Lyon, Lyon, France, October 1990.

[5] S.G. Akl, and K. Qiu, "Parallel Minimum Spanning Forest Algorithms on the Star and Pancake Interconnection Networks", in *Proceedings of the Joint Conference on Vector and Parallel Processing*, Lyon, France, pp. 565-570, 1992.

[6] S.G. Akl, K. Qiu, and I. Stojmenović, "Computing the Voronoi Diagram on the Star and Pancake Interconnection Networks", in *Proceedings of the 4^{th} Canadian Conference on Computational Geometry*, St. John's, Newfoundland, pp. 353-358, 1992.

[7] S.G. Akl, and K. Qiu, "A Novel Routing Scheme on the Star and Pancake Networks and its Applications", *Parallel Computing*, vol. 19, no. 1, pp. 95-101, 1993.

[8] S.G. Akl, K. Qiu, and I. Stojmenović, "Fundamental Algorithms for the Star and Pancake Interconnection Networks with Applications to Computational Geometry", to appear in *Networks*.

[9] P. Berthomé, A. Ferreira, and S. Perennes, "Decomposing Hierarchical Cayley Graphs, with Applications to Information Dissemination and Algorithm Design", Technical Report no. 92-32, Laboratoire de l' Informatique du Parallelism, Ecole Normale Supérieure de Lyon, Lyon, France, July 1992.

[10] D.P. Bertsekas, and J.N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Englewood Cliffs, NJ, Prentice Hall, 1989.

[11] D.P. Bertsekas, C. Ozveren, G.D. Stamoulis, P. Tseng, and J.N. Tsitsiklis, "Optimal Communication Algorithms for Hypercubes", *Journal of Parallel and Distributed Computing*, vol. 11, pp. 263-275, 1991.

[12] P. Fragopoulou, "Parallel Algorithms for the Fourier and Other Mathematical Transforms," Master's Thesis, Department of Computing and Information Science, Queen's University, Kingston, ON, Canada, August 1990.

[13] P. Fragopoulou, and S.G. Akl, "A Parallel Algorithm for Computing Fourier Transforms on the Star Graph," in *Proceedings of the International Conference on Parallel Processing*, St. Charles, IL, vol. III, pp. 100-106, 1991, to appear in *IEEE Transactions on Parallel and Distributed Systems.*

[14] P. Fragopoulou, and S.G. Akl, "Edge-Disjoint Spanning Trees on the Star Network with Applications to Fault Tolerance", Technical Report No. 93-354, Department of Computing and Information Science, Queen's University, Kingston, ON, Canada, October 1993.

[15] P. Fraigniaud, E. Lazard, "Methods and Models of Communication in Usual Networks", Technical Report, LIP, ENS-Lyon, France, 1991.

[16] S.L. Johnson, and C.T. Ho, "Optimum Broadcasting and Personalized Communication in Hypercubes", *IEEE Transactions on Computers*, vol. 38, no. 9 , pp. 1249-1268, 1989.

[17] J.S. Jwo, S. Lakshimivarahan, and S.K. Dhall, "Embedding of Cycles and Grids in Star Graphs", in *Proceedings of the IEEE Symposium on Parallel and Distributed Processing*, Dallas, TX, pp. 540-547, 1990.

[18] D.E. Knuth, *The Art of Computer Programming*, Addison-Wesley, Reading, MA, 1973.

[19] F.T. Leighton, *Complexity Issues in VLSI: Optimal Layouts for the Shuffle Exchange and Other Networks*, MIT Press, 1983.

[20] V.E. Media, and D. Sarkar, "Optimal Broadcasting on the Star Graph", *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, no. 4, pp. 389-396, 1992.

[21] A. Menn, and A.K. Somani. "An Efficient Sorting Algorithm for the Star Graph Interconnection Network", in *Proceedings of the International Conference on Parallel Processing*, St. Charles, IL, vol. III, pp. 1-8, 1990.

[22] M. Nigam, S. Sahni, and B. Krishnamurthy, "Embedding Hamiltonians and Hypercubes in Star Interconnection Graphs", in *Proceedings of the International Conference on Parallel Processing*, St. Charles, IL, pp. 340-343, 1990.

[23] M. Palis, S. Rajasekaran, and D.S.L. Wei, "General Routing Algorithms for Star Graphs", in *Proceedings of the Fourth International Parallel Processing Symposium*, pp. 597-611, 1990.

[24] K. Qiu, H. Meijer, and S. G. Akl, "Decomposing a Star Graph into Disjoint Cycles", *Information Processing Letters*, vol. 39, no. 3, pp. 125-129, 1991.

[25] K. Qiu, S.G. Akl, and I. Stojmenović, "Data Communication and Computational Geometry on the Star and Pancake Networks", in *Proceedings of the IEEE Symposium on Parallel and Distributed Processing*, Dallas, TX, 1991.

[26] K. Qiu, "The Star and Pancake Interconnection Networks: Properties and Algorithms", Ph.D Thesis, Department of Computing and Information Science, Queen's University, Kingston, ON, Canada, May 1992.

[27] K. Qiu, and S.G. Akl, "Load Balancing and Selection on the Star and Pancake Interconnection Networks", in *Proceedings of the 26th Hawaii International Conference on System Sciences*, Hawaii, vol. 2, pp. 235-242, 1993.

[28] K. Qiu, H. Meijer, and S.G. Akl, "On the Cycle Structure of Star Graphs" in *Proceedings of the 24th Southwestern Conference on Combinatorics, Graph theory and Computing*, Boca Raton, FL, 1993.

[29] K. Qiu, S.G. Akl, and H. Meijer, "On some Properties and Algorithms for the Star and Pancake Interconnection Networks", to appear in *Journal of Parallel and Distributed Computing.*

[30] K. Qiu, P. Fragopoulou, and S.G. Akl, "On the Tree Structure of the Star Graph", Technical Report 93-349, Dept. Comput. Info. Sc., Queen's Univ., Kingston, Canada, 1993.

Appendix A

Theorem 3: If x and y represent nodes of S_n, and x' and y' are produced by x and y respectively under cycle rotation then:

1. If x and y are connected then x' and y' are also connected (in other words connectivity is preserved under cycle rotation in S_n).
2. If (x, y) (the link connecting nodes x and y) is a link of type $t(x, y)$ then (x', y') is a link of type $t(x', y') = (t(x, y) - 1)\text{mod}(n - 1) + 2$.

Proof: This is an immediate consequence of the fact that cycle rotation induces an automorphism on the star graph. More analytically it can be proven as follows: Let us remind the definition of function r with domain and range $\{1, 2, ..., n\}$ in the definition of cycle rotation:

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

If $x = x_1 x_2 ... x_n$ is a node of S_n and $x' = x'_1 x'_2 ... x'_n$ is obtained from x under cycle rotation, then the following are true:

$$x'_i = \begin{cases} r(x_1), & \text{if } i = 1 \\ r(x_n), & \text{if } i = 2 \\ r(x_{i-1}), & \text{otherwise} \end{cases}$$

Assume that (x, y) is an edge of type i, and that x' and y' are nodes produced by x and y, respectively, under cycle rotation; if we write (x, y) and (x, y') as:

$$(x_1 x_2 ... x_{i-1} x_i x_{i+1} ... x_n, x_i x_2 ... x_{i-1} x_1 x_{i+1} ... x_n)$$

$$(r(x_1)r(x_n)r(x_2)...r(x_{i-2})r(x_{i-1})r(x_i)...r(x_{n-1}), r(x_i)r(x_n)r(x_2)...r(x_{i-2})r(x_{i-1})r(x_1)...r(x_{n-1}))$$

we see that when x and y differ in the 1^{st} and the i^{th} positions then x' and y' differ in the 1^{st} and the $((i-1)\text{mod}(n-1)+2)^{nd}$ positions. As a consequence, $t(x', y') = (t(x, y) - 1)\text{mod}(n - 1) + 2$. □

Appendix B

Theorem 8: Spanning tree $MBST$ satisfies the requirements of multinode broadcasting algorithm because:

1. All $n-1$ links that are used for message transmission at each time step of the algorithm are of different types.

2. Node x' produced by x following a link of type $v(x)$ is one link closer to node $12...n$; this guarantees that the path from the root of the tree to x has minimum length.

3. If x' is produced by x following a link with of $v(x)$ then $m(x) - m(x') \geq n-1$; this guarantees that x' receives the message before x.

Proof:

1. This is obvious from the way the $v(x)$'s were created.

2. By the way $v(x)$ was mapped to each node x this follows naturally.

3. The proof for this is separated into the following parts:

   (a) Each node x that belongs to some $N_{kj}, j \neq 1$, is connected through link $v(x)$ to some node x' in some $N_{k'j'}$, $k' < k$. Then $m(x) - m(x') \geq n-1$, since all the $n-1$ nodes of N_{k1} (theorem 6) are between x and x'.

   Nodes included in $D_{\lfloor \frac{3(n-1)}{2} \rfloor}$, for n odd, however need special treatment. Each of these nodes x starts with symbol 1 and as a consequence it is connected through $v(x)$ to a node in $D_{\lfloor \frac{3(n-1)}{2} \rfloor - 1}$, that does not start with symbol 1. If we impose the restriction that in each D_k all the nodes that start with symbol 1 are after all other nodes and since it is easy to show that $D_{\lfloor \frac{3(n-1)}{2} \rfloor - 1}$ has at least $n-1$ nodes that start with symbol 1, this guarantees that for every $x \in D_{\lfloor \frac{3(n-1)}{2} \rfloor}$, $m(x) - m(x') \geq n-1$.

   (b) We now prove that each node x in N_{k1} is connected through link $v(x)$ to node x' in $N_{(k-1)1}$. For $1 \leq k \leq n-1$ nodes in N_{k1} have exactly one cycle. The generator node x of N_{k1} has cycle notation $(k+1, k, ..., 2, 1)$ and $v(x) = k+1$. Starting from x and following link $v(x)$, the resulting node x' has symbol $k+1$ in its proper position and as a consequence its cycle notation is $(k, k-1, ..., 2, 1)$. But this is by definition the cycle notation of the generator node of $N_{(k-1)1}$. For $n \leq k \leq \lfloor \frac{3(n-1)}{2} \rfloor$ the cycle notation of the generator node x of N_{k1} has the form $(n, n-1)(n-2, n-3)...(i, i-1)(i-2, ...2, 1)$ and $v(x)$ equals the smallest symbol of the second cycle from the left ($i-1$ in this case). Starting from x and following $v(x)$, it is easy to see that in the cycle notation of the resulting node the two cycles to the right are merged into one $(n, n-1)(n-2, n-3), ..., (i, i-1, i-2, ..., 2, 1)$. But by definition this is nothing more than the cycle notation of the generator node of $N_{(k-1)1}$. Since connectivity is preserved under cycle rotation and a link type t is transformed into link type $t' = (t-1)\text{mod}(n-1) + 2$ (theorem 3), this guarantees that subsequent nodes x in N_{k1} are connected through $v(x)$ to subsequent nodes x' in $N_{(k-1)1}$. In order to guarantee that for each $x \in N_{k1}$, $m(x) - m(x') \geq n-1$, we have to prove that for each k there are at least $n-1$ nodes between $N_{(k-1)1}$ and N_{k1}. It is the same if we prove that for each $2 \leq k \leq D_{\lfloor \frac{3(n-1)}{2} \rfloor -1}$, D_k includes at least $2(n-1)$ nodes. We know that D_1 has $n-1$ nodes and D_2 has $2(n-1)$ nodes. If we show that $D_{\lfloor \frac{3(n-1)}{2} \rfloor -1}$ also has at least $2(n-1)$ nodes this will guarantee that each D_k in between has at least $2(n-1)$ nodes as well. For n odd, we take the special case of nodes that have cycle notations $(1,*)(5,6)...(n-1,n)$, where $*$ represents one of the 3! permutations of symbols $\{2,3,4\}$. It is easy to show that each of these 3! different cycle notations create a different necklace of $n-1$ nodes (theorem 6). As a consequence, for n even, there are at least $3!(n-1)$ nodes in $D_{\lfloor \frac{3(n-1)}{2} \rfloor -1}$. With the same reasoning, we can prove that for n odd there are at least $2!(n-1)$ nodes in $D_{\lfloor \frac{3(n-1)}{2} \rfloor -1}$. For example for $n=5$ the cycles that correspond to the above description and the necklaces they create are:

$$\begin{array}{ccccccc} (123)(45) & \rightarrow & (134)(52) & \rightarrow & (145)(23) & \rightarrow & (152)(34) \\ (132)(45) & \rightarrow & (143)(52) & \rightarrow & (154)(23) & \rightarrow & (125)(34) \end{array}$$

□