

Broadcasting Algorithms for the Star-Connected Cycles Interconnection Network

Marcelo Moraes de Azevedo and Nader Bagherzadeh*

Department of Electrical and Computer Engineering

University of California, Irvine – Irvine, CA 92717

email: mazevedo@ece.uci.edu, nader@ece.uci.edu

Phone: (714) 824-8720 FAX: (714) 824-2321

Shahram Latifi

Department of Electrical and Computer Engineering

University of Nevada, Las Vegas – Las Vegas, NV 89154-4026

email: latifi@jb.ee.unlv.edu

Phone: (702) 895-4016 FAX: (702) 895-4075

Abstract — *The star-connected cycles (SCC) graph was recently proposed as an alternative to the cube-connected cycles (CCC) graph, using a star graph to connect cycles of nodes rather than a hypercube.*

This paper presents an analysis of broadcasting algorithms for SIMD and MIMD SCCs, covering both one-port and multiple-port versions. We show that $O(n \log n)$ one-port broadcasting algorithms that have been proposed for the n -star cannot be efficiently extended to the case of the n -SCC graph. However, a simple but rather inefficient algorithm requiring $O(n^2)$ steps in the n -star graph can run in $O(n)$ time in the n -SCC if a proper combination of parallelism and transmission rates in the links connecting the nodes is selected. The result is that broadcasting in the n -SCC can be accomplished efficiently, requiring a running time better than or equal to that of an n -star containing $(n - 1)$ times fewer nodes.

Index terms — *Broadcasting, star-connected cycles graph, fixed-degree graphs, interconnection networks, parallel processing.*

1 Introduction

The steady progress observed in the field of VLSI technology continuously establishes new parameters and design choices in terms of cost, complexity, functionality and reliability of available devices and modules, making it viable to devise high performance computing systems targeted at different applications. One remarkable example of an architecture whose implementation was made possible by advances in VLSI technology are the massively parallel computing systems. Such systems can contain hundreds or thousands of processors connected via a particular type of interconnection network.

The choice of a proper interconnection network is a major decision in the design of a massively parallel system. This choice may affect several characteristics of the final system, such as performance, reliability, scalability, complexity of physical lay-out and cost.

Some interconnection networks that have been proposed for massively parallel systems are the hypercube [1] and the star graph [2]. These topologies provide advantages such as low diameter, simplicity of routing, symmetry and hierarchical structure. However, since both the hypercube and the star graph are variable-

*This research is supported in part by Conselho Nacional de Desenvolvimento Científico e Tecnológico (Brazil), under the grant No. 200392/92-1.

degree graphs, their applicability may be compromised when criteria such as scalability and complexity of physical lay-out are at premium.

As an alternative to overcome these difficulties, fixed-degree graphs such as the cube-connected cycles (CCC) [3] and the star-connected cycles (SCC) [4] have been proposed. An n -CCC graph is formed by substituting each node of an n -cube with a ring of n or more nodes. Accordingly, an n -SCC graph is formed by substituting each node of an n -star with a ring of $(n - 1)$ nodes. Fixed-degree interconnection networks may however present a longer diameter, which affects the performance of the system if proper communication techniques and well-devised algorithms presenting locality of operation are not used.

One basic problem that must be efficiently handled by most interconnection networks is broadcasting. Broadcasting is used by parallel algorithms that tackle problems such as matrix-matrix and matrix-vector multiplication, LU factorization, database queries and transitive closure of graphs.

In this paper, we investigate different broadcasting algorithms for the SCC graph, considering both the SIMD and the MIMD computational models. We initially analyze how efficient $O(n \log n)$ algorithms that have been proposed for the n -star graph can be extended to an n -SCC. We show that such algorithms do not find an efficient implementation on the SCC and therefore should have its applicability limited to the star graph. Rather surprisingly, we show that a simple but slow $O(n^2)$ broadcasting algorithm proposed in [2] for the n -star can be efficiently mapped onto the n -SCC graph. Actually, we show that both one-port and multiple-port broadcasting in an n -SCC graph can be accomplished in $O(n)$ running time, which is better than or equal to the running time required by an optimal one-port broadcasting algorithm targeted at an n -star containing $(n - 1)$ times fewer nodes.

We also show that for $4 \leq n \leq 8$ the number of steps required to run a multiple-port broadcasting algorithm in an n -SCC is at most 17.6% higher than the diameter of the graph, suggesting that the proposed broadcasting algorithm is very close to optimality.

Also included in this paper is a comparison between SCC, star and CCC graphs of similar size in respect to the number of steps required by their corresponding broadcasting algorithms.

Finally, we also show how broadcasting algorithms for the SCC graph can be extended to support transmission of a sequence of messages in pipelined fashion. The resulting pipelined broadcasting algorithms can be used in applications such as input and physical distribution of data values required by different parallel algorithms.

2 Background

2.1 The Star Graph

An n -star graph contains $n!$ nodes that are labeled with the $n!$ possible permutations of n distinct symbols. In this paper, we choose the digits $\{1, 2, \dots, n\}$ as the symbols used to label the nodes of an n -star.

A node labeled with permutation $P_i = i_1 i_2 \dots i_j \dots i_n$ is connected to $(n - 1)$ distinct nodes, respectively labeled with permutations $i_j i_2 \dots i_{j-1} i_1 i_{j+1} \dots i_n$, $2 \leq j \leq n$. In other words, a node labeled with permutation P_i is connected to other $(n - 1)$ nodes whose labels are the permutations resulting from exchanging the digit in position j in P_i with the first digit of P_i , where $2 \leq j \leq n$ [2], [5]. In addition, the link connecting node $P_i = i_1 i_2 \dots i_j \dots i_n$ to node $i_j i_2 \dots i_{j-1} i_1 i_{j+1} \dots i_n$ is labeled j to indicate a connection along the j th dimension of the star graph. A 4-star graph is shown in Figure 1.

An n -star graph is a regular graph with degree $\delta = n - 1$. The n -star graph exhibits vertex and edge symmetry, hierarchical structure and simple routing. In addition, the n -star presents a low diameter, given

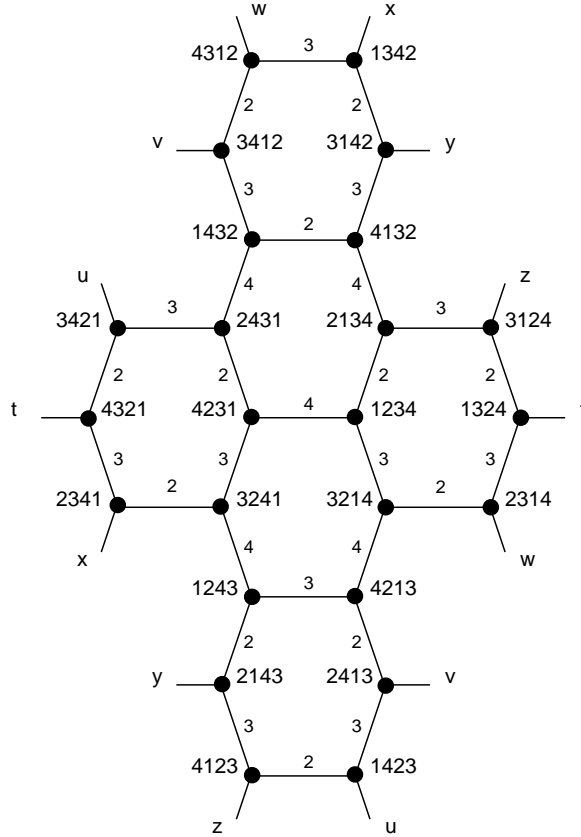


Figure 1: 4-star graph

by $d_{star} = \lceil 3(n - 1)/2 \rceil$ [2].

For a brief description of how routing can be accomplished in a star graph, assume that we want to route from P_s to P_d , where P_s is the source node and P_d is the destination node. If $P_s \neq P_d$, then there is a path from P_s to P_d with at least one link. To find the links connecting P_s to P_d , we can instead find the path from P_{ds} to the identity permutation [5], where $P_{ds} = P_d^{-1}P_s$. After calculating P_{ds} , the following routing algorithm applies:

1. If the first digit in permutation P_{ds} is 1, move it to any position not occupied by the correct digit.
2. If x (i.e. any digit other than 1) is first, move it to its position.

We may organize the digits of permutation P_{ds} as a set of cycles – i.e. cyclically ordered sets of digits with the property that each digit's desired position is that occupied by the next digit in the set. A permutation $P_{ds} = 26543187$ belonging to an 8-star graph, for instance, consists of the following cycles: (1 2 6), (3 5), (7 8), (4). Note that any digit already in its correct position appears as a 1-cycle. We assume in this paper that all cycles are written in canonical form [7], i.e. the smallest digit appears first in the representation of a cycle.

Let c be the number of cycles of length at least 2 and m the total number of digits in these cycles. Then the minimum number of links in the path from P_s to P_d is [2] :

$$d_{ds} = \begin{cases} c + m & , \text{ if 1 is the first digit in } P_{ds} \\ c + m - 2 & , \text{ if 1 is not the first digit in } P_{ds} \end{cases}$$

Let C_i be an r -cycle of the form $C_i = (a_1 a_2 \dots a_r)$, included in permutation P_{ds} ($2 \leq r \leq n$). The execution of an r -cycle C_i corresponds to a path R in the star graph and can be expressed as a sequence of links as follows [6]:

$$\begin{aligned} R &= (a_2, a_3, \dots, a_r) && , \text{ if } a_1 = 1 \\ R &= (a_{1+k \bmod r}, a_{1+(k+1) \bmod r}, \dots, a_{1+(k+r-1) \bmod r}, a_{1+k \bmod r}) && , \text{ if } a_1 \neq 1, \text{ for } 0 \leq k \leq r-1 \end{aligned}$$

Hence, there are N_i different ways to minimally execute C_i [4], [6], where:

$$N_i = \begin{cases} r & , \text{ if } a_1 \neq 1 \\ 1 & , \text{ if } a_1 = 1 \end{cases}$$

Let c be the number of r -cycles in permutation P_{ds} such that $2 \leq r \leq n$. Let also $C_k = (1 a_2 a_3 \dots a_b)$ be a b -cycle in P_{ds} which contains digit 1, $1 \leq b \leq n$. We may choose any order to execute the cycles in P_{ds} . Also, according to the routing algorithm for the star graph described earlier the execution of C_k can be interrupted at any point and interleaved with other cycles, being resumed afterwards. Therefore, the total number of disjoint paths that might be used to execute the cycles C_1, C_2, \dots, C_c in P_{ds} is:

$$T_p = \begin{cases} \frac{(c+b-2)!}{(b-1)!} \prod_{i=1}^c N_i & , \text{ if the first digit in } P_{ds} \text{ is not 1 } (2 \leq b \leq n) \\ c! \prod_{i=1}^c N_i & , \text{ if the first digit in } P_{ds} \text{ is 1 } (b = 1) \end{cases} \quad (1)$$

As an example, permutation 23154 has two cycles: $C_1 = (1 2 3)$ and $C_2 = (4 5)$. The execution paths for C_1 and C_2 are $R_1 = (2, 3)$ and $R_2 = (4, 5, 4) \equiv (5, 4, 5)$, respectively. We may therefore execute both cycles as the following sequences of links: $(2, 3, 4, 5, 4)$, $(2, 3, 5, 4, 5)$, $(4, 5, 4, 2, 3)$, $(5, 4, 5, 2, 3)$, $(2, 4, 5, 4, 3)$ or $(2, 5, 4, 5, 3)$.

2.2 The Star-Connected Cycles Graph

An n -SCC graph is obtained by replacing each node of an n -star with a ring of $(n-1)$ nodes. Each ring may be viewed as a *supernode* that can be implemented as a cluster of individual processors or as a single multiprocessor VLSI device. The connections between nodes inside the same supernode are referred to as *local links*. Also, each supernode is connected to $(n-1)$ adjacent supernodes, using *lateral links* according to the topology of the n -star graph. Each lateral link connects to exactly one of the $(n-1)$ nodes belonging to a supernode. The nodes in each ring are identified by a pair of labels (I_j, P_i) , where:

- P_i is a permutation obtained using the generators of the n -star graph [2], [5]. As in the case of the n -star, we assume that P_i is a permutation of the digits $\{1, 2, \dots, n\}$. P_i remains unchanged when the node of an n -star is replaced with $(n-1)$ nodes in an n -SCC graph, such that P_i does not vary among the nodes that belong to the same ring or supernode.
- I_j is a single digit that identifies each particular node inside a ring. The labeling method proposed for the n -SCC consists of assigning to each I_j a label in the range $\{2, 3, \dots, n\}$, such that I_j corresponds to the label of the lateral link used to connect each node within a ring to other rings in the n -SCC graph (i.e., I_j is a dimension of the n -star).

As an example, consider the 4-SCC graph shown in Figure 2. Node 1234 of a 4-star graph is replaced with 3 nodes, labeled respectively as $(2,1234)$, $(3,1234)$ and $(4,1234)$. These nodes form a supernode of a

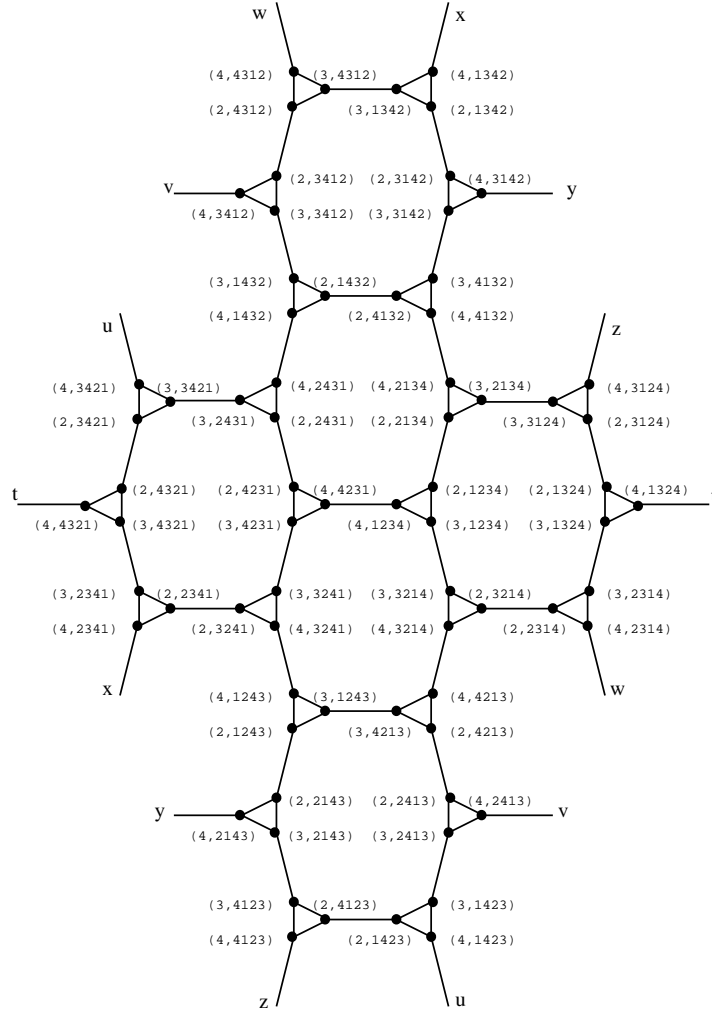


Figure 2: 4-SCC graph

4-SCC and are connected to other supernodes using lateral links 2, 3 and 4 (e.g. node (2,1234) is connected to node (2,2134) via lateral link 2).

An n -SCC graph can be seen as an n -star graph connecting $n!$ supernodes containing $(n - 1)$ nodes each. Therefore, the total number of nodes in an n -SCC graph is $N = (n - 1)n!$. Also, an n -SCC is a regular graph with degree $\delta = 3$, for $n > 3$.

The SCC graph is node-symmetric and also shows the property of edge symmetry if we separate the communication links into two distinct sets of edges (namely, local and lateral links). More specifically, every lateral link in an SCC graph is edge-symmetric with any other lateral link in the graph. The local links within a ring or supernode are also transitive with any other local link in the graph [4], [8].

The n -SCC graph can be represented as the product of two Cayley graphs (namely, the n -star and the $(n - 1)$ -ring) [5]. More specifically, an n -SCC graph is reducible to a quotient n -star graph if we substitute each of its $(n - 1)$ -ring subgraphs with a single node.

Routing in the SCC is an extension of routing in the star graph and can be seen as two different problems: routing in the lateral links and routing in the local links. In this paper we briefly describe the routing algorithm for the SCC graph. However, the reader may wish to refer to [4] for additional details.

A route between any pair of nodes in the SCC may involve both lateral and local links. Initially, the routing algorithm views the SCC as a quotient star graph and identifies all cycles that have to be executed in the path between the supernodes that contain the source and the destination node. The execution of those cycles is actually a sequence of lateral links in the SCC, and T_p different orderings containing the same minimal number of lateral links are allowed in the routing (Equation 1). However, as the order of traversal of the lateral links affects the number of local links in the route, a proper routing algorithm is used to select an optimal sequence of lateral links [4]. Once this order is chosen, the number of required local links is kept to a minimum.

As an example, consider the execution of a cycle $(a_i a_j)$ in an n -SCC graph (Figure 3), where $a_i, a_j \neq 1$. Such cycle can be executed as two possible orders of lateral links, namely (a_i, a_j, a_i) or (a_j, a_i, a_j) . To move between two consecutive lateral links, a maximum of $\lfloor (n-1)/2 \rfloor$ local links must be traversed. Therefore, execution of a 2-cycle $(a_i a_j)$ requires 3 lateral links and at most $2\lfloor (n-1)/2 \rfloor$ local links. Finally, when the execution of a cycle in the n -SCC is completed, a maximum of $\lfloor (n-1)/2 \rfloor$ local links are required to move into the first lateral link of the next cycle.

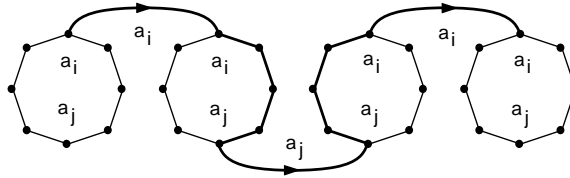


Figure 3: Execution of a cycle $(a_i a_j)$ in the n -SCC

In general, if the lateral links entering and leaving a supernode are respectively I_i and I_j , then the minimum number of local links required to traverse the supernode is [4] :

$$D_n = \min(D', n - 1 - D') \quad , \text{ where } D' = |I_j - I_i|$$

The diameter of an n -SCC graph can be calculated by identifying antipode¹ nodes in the graph and then evaluating the distance to the identity node using the routing algorithm described above. The resulting expression for the diameter of the n -SCC graph is given by [4]:

$$d_{SCC} = \begin{cases} 2 \left(\lfloor \frac{n-1}{2} \rfloor \right)^2 + \left\lfloor \frac{3(n-1)}{2} \right\rfloor + 2 \left\lfloor \frac{n}{2} \right\rfloor - 2 & , \text{ if } n \neq 3 \\ 6 & , \text{ if } n = 3 \end{cases} \quad (2)$$

3 Broadcasting in the SCC Graph

A broadcasting algorithm for the SCC graph consists of a sequence of transmissions over lateral and local links, such that a particular piece of information originated by a node is passed on to all other processors in the interconnection network.

At each step of the algorithm, every node communicates with one of its neighbors and compares notes on whether either of them has already received the information that is being broadcast. If only one node has received it, then additional communication takes place to relay the broadcast information to the uninformed node. If both or neither of the nodes have already received the information, then no additional messages are

¹An antipode is the farthest node from a given node along the shortest path [6].

exchanged between the nodes. We assume that node comparison between any pair of nodes is accomplished with full-duplex (i.e., bidirectional) communication links, both in the case of lateral and local links.

Broadcasting algorithms can be based on two distinct communication models, namely *one-port* communication or *multiple-port* communication. In the one-port communication model, each node sends messages using only one port at each step of the algorithm. With this scheme, the number of informed nodes can at most double at each step of the algorithm. Therefore, broadcasting in a graph with N nodes using a one-port communication algorithm requires at least $\log N$ steps². In the particular case of graphs of bounded degree $\delta \leq 3$, Liestman and Peters [9] showed that at least $1.4404 \log N - 1.769$ steps are required to accomplish one-port broadcasting.

In a multiple-port communication model, each node sends messages using two or more ports at each step of the algorithm. Assuming a regular graph with N nodes and degree δ , a broadcasting algorithm based on an m -port communication model ($1 \leq m \leq \delta$) requires at least $\log_{(m+1)} N$ steps.

Broadcasting also depends on the computational model being considered for the interconnection network. In the case of the SIMD model, the processors operate in a lock-step fashion, meaning that at any given time the same types of links must be used for communication by every processor. On the other hand, the MIMD model is more generic and allows utilization of different types of links at any step of the algorithm.

3.1 One-port Broadcasting in the SCC

Efficient one-port broadcasting algorithms have been proposed for different interconnection networks. One-port broadcasting in the n -cube, for instance, can be accomplished with an optimal algorithm in n steps [2], [9], [10]. The n -CCC also allows for efficient one-port broadcasting, which can be done in $\lceil 5n/2 \rceil - 1$ steps [9]. A one-port broadcasting algorithm for the n -star requiring at most $3(n \log n - n/2)$ steps was introduced in [2], followed by an optimal algorithm requiring $\sum_{i=2}^n (\lceil \log i \rceil + 1)$ steps [11]. In either case, the complexity of one-port broadcasting algorithms for the n -star is $O(n \log n)$.

Broadcasting in an SCC graph is basically an extension of the broadcasting algorithms already introduced for the star. Our goal is to find a sequence of lateral links σ such that for every pair of nodes in the graph there exists a subsequence that forms a path from one supernode to the other. As long as σ satisfies this condition, it constitutes a broadcasting algorithm. Of course, a proper choice of local links transmissions must be used between lateral link steps such that the information is correctly broadcast among the nodes inside each supernode.

Ideally, we should find an $O(\log N) = O(n \log n)$ broadcasting sequence for the n -SCC. However, if we recall that the diameter of the n -SCC contains a quadratic term (Equation 2), the following theorem holds:

Theorem 1 *One-port broadcasting in an n -SCC graph requires a sequence with at least $O(n^2)$ steps.*

Proof: The proof follows from the observation that any broadcasting sequence must include subsequences allowing the node originating the broadcast message to communicate with all other nodes in the graph. Clearly, the broadcasting sequence must be at least as long as the longest communication path existing in an n -SCC graph (i.e., the diameter). If we recall that the dominant term in the diameter of the n -SCC is $2(\lfloor (n-1)/2 \rfloor)^2 \approx 0.5n^2$, the theorem follows. \square

With that limitation in mind, different possible broadcasting sequences have been investigated [8]. The general approach for analyzing a broadcasting sequence for the SCC graph is done in two steps. First, we

²All logarithms in this paper are base 2, unless otherwise indicated.

must consider how many lateral link steps each sequence requires, and then we evaluate the number of local link steps. In any case, the sequence of lateral links required by the broadcasting algorithm is defined by the quotient star graph embedded in the SCC.

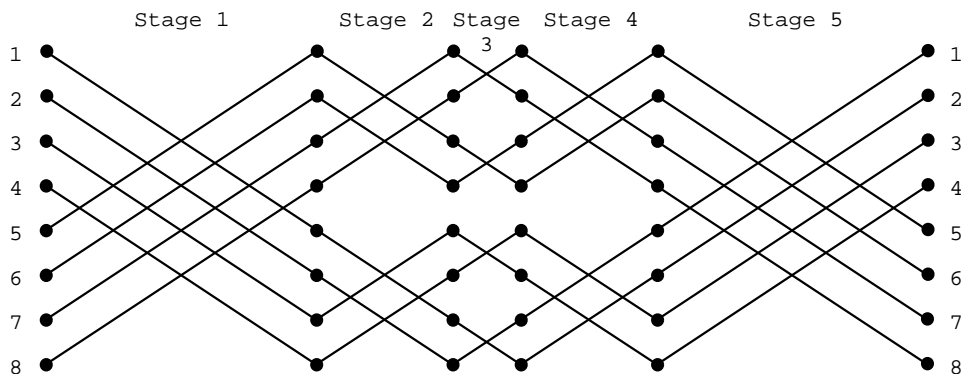


Figure 4: An $n \log n$ switching network (T_8)

Efficient broadcasting sequences for the star graph requiring $O(n \log n)$ steps were presented in [2], [11]. The algorithm presented in [2], for instance, uses a broadcasting sequence $\sigma_{star}(T_n)$ containing $(n \log n - n/2)$ pairwise interchanges of digits (a_i, a_j) chosen from a switching network T_n with $(2 \log n - 1)$ stages and $(n \log n - n/2)$ switches. As an example, consider the switching network shown in Figure 4 (T_8). The first stage consists of switches $(1, 5)$ $(2, 6)$ $(3, 7)$ $(4, 8)$ and can be represented by the following sequence of star operations:

$$5 - 6 - 2 - 6 - 7 - 3 - 7 - 8 - 4 - 8$$

A similar analysis over all stages of T_n yields a broadcast sequence $\sigma_{star}(T_n)$ for an n -star graph of length $|\sigma_{star}(T_n)| \leq 3(n \log n - n/2)$ [2].

The sequence resulting from T_n can be shown to contain all subsequences of links that are required to properly carry a piece of information between any pair of nodes in the star graph [2], and therefore can be used for broadcast purposes. However, in order to extend $\sigma_{star}(T_n)$ to the case of the n -SCC graph, local link steps must be properly inserted between lateral link transmissions, yielding a one-port broadcasting algorithm with $|\sigma_{SCC}(T_n)|$ steps [8], where:

$$|\sigma_{SCC}(T_n)| < 4n \left\lfloor \frac{n}{2} \right\rfloor + (5n - 4)(\log n - 1) - \frac{5n}{2} \quad (3)$$

A comparison of Equation 3 and Theorem 1 reveals that the one-port broadcasting algorithm resulting from an $n \log n$ switching network, although efficient when used in the star graph ($\approx 3n \log n$ steps), is far from being optimal in the case of the SCC graph ($\approx 2n^2$ steps). This conclusion follows from the observation that the number of steps required by a broadcasting algorithm in any interconnection network should be as close as possible to its diameter.

An $n^2/2$ switching network (T_n) was also investigated in [8], resulting in a one-port broadcasting algorithm requiring $|\sigma_{star}(T_n)| = 1.5n^2 - 3.5n + 2$ steps in the case of the star graph and $|\sigma_{SCC}(T_n)| = 3.5n^2 - 9.5n + 6$ steps in the case of the SCC graph. Such a broadcasting algorithm also proves to be inefficient.

A major limitation of broadcasting sequences obtained from switching networks is the requirement of a sequential execution of lateral link steps. We recall that such broadcasting sequences should contain all

subsequences of links possibly needed for communication between any two nodes in the quotient star graph. Therefore, any attempt to reduce the number of steps in the broadcasting algorithm by executing multiple steps of the original broadcast sequence in parallel should not violate that requirement. An analysis of sequences such as $\sigma_{star}(T_n)$ and $\sigma_{star}(\mathcal{T}_n)$ reveals that these sequences have an intrinsically sequential nature that makes the introduction of parallelism difficult [8]. The same limitation holds for the optimal broadcasting algorithm proposed for the star graph in [11] when extended to the SCC graph.

Interestingly, an analysis of broadcasting sequences presented in [8] shows that an efficient algorithm for the SCC can be obtained from a cyclic sequence of digits σ_c requiring $O(n^2)$ steps in the star graph. Such a sequence is formed by repeating a cyclic pattern of lateral links represented by the digits $(2\ 3\ \dots\ n)$ as follows:

$$\sigma_c = 2\ 3\ 4\ \dots\ (n-1)\ n\ 2\ 3\ 4\ \dots\ (n-1)\ n\ \dots\ (d_{star}\ \text{times})$$

Note that σ_c repeats the pattern $(2\ 3\ \dots\ n)$ d_{star} times, where d_{star} is the diameter of the quotient n -star embedded in the n -SCC graph. Therefore, σ_c includes all possible paths between any two supernodes in the n -SCC, and as long as a proper sequence of local links is also chosen, σ_c can be used as a broadcasting algorithm for the n -SCC graph.

Although the length of the above sequence is $O(n^2)$, σ_c can actually run in $O(n)$ lateral link steps in an n -star graph by using parallel transmissions in those links:

Theorem 2 *The use of a δ -port communication model in an n -star graph with degree $\delta = n - 1$ and diameter $d_{star} = \lceil 3(n - 1)/2 \rceil$ yields an optimal broadcasting algorithm requiring only d_{star} steps.*

Proof: At each step, a node running a δ -port broadcasting algorithm uses all its ports to propagate the information to be broadcast. Suppose that at the beginning of the algorithm, only node P_i holds the information. After the first step of the algorithm, all nodes within a distance of one lateral link from P_i will also have received the information. After the second step, the information has been propagated to all nodes within two lateral links of P_i , and so on. After d_{star} steps, all nodes in the n -star graph have received the broadcast information.

Optimality results from the observation that no broadcasting algorithm can use a shorter sequence of steps than the longest distance between any pair of nodes in the n -star (i.e., the diameter). Since a δ -port broadcasting uses exactly d_{star} steps, the algorithm is optimal. \square

Note that the technique described in Theorem 2 for the n -star graph results in a multiple-port algorithm with $O(n)$ steps, while a one-port broadcasting algorithm in the same graph requires $O(n \log n)$ steps [2].

A main disadvantage of using such a δ -port algorithm in a star graph is that we may impose severe communication overhead on the nodes. The algorithm may even be difficult to implement or require special hardware support for δ -port communication, specially on large degree star graphs with high transmission rates in the lateral links.

These restrictions do not apply to the n -SCC graph, since the task of simultaneous communication over the lateral links is equally distributed over $(n - 1)$ nodes belonging to the same supernode. Therefore, we may take advantage of parallel transmissions in the lateral links to implement a faster broadcasting algorithm in the n -SCC graph.

An analysis of sequence σ_c shows that such a sequence can be efficiently executed by using all lateral links of each supernode simultaneously. To illustrate this reasoning, Figure 5 shows the initial steps required to run σ_c in a δ -SCC graph. This approach allows a fast execution of σ_c , since it initially broadcasts the

information inside the supernode and then uses all lateral links simultaneously to pass the information on to adjacent supernodes. The full broadcasting algorithm requires this operation to be repeated d_{star} times. Note that we still have a one-port broadcasting algorithm, since at each step every node tries to compare notes using a single communication port. Of course, the nodes may also receive a communication request in a second port while transmitting in another port. Another interesting observation is that this technique actually runs $\delta = (n - 1)$ lateral link steps of σ_c in parallel, therefore reducing the number of steps that would be required if we used only one lateral link per supernode at a time by a factor of $(n - 1)$.

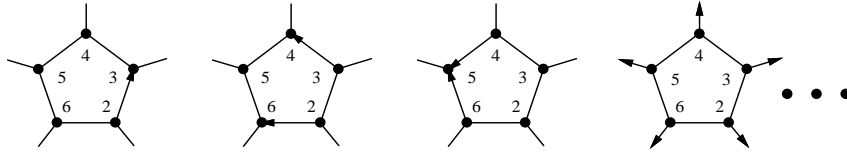


Figure 5: One-port broadcasting in a MIMD δ -SCC (cyclic sequence)

Note that the concept of parallelism is also used inside the ring by forcing one of the nodes to use different local links in the first two steps of a subsequence of local link transmissions. Each remaining node in the ring simply propagates the information using the same direction chosen by their informed neighbors. A total of $\lfloor n/2 \rfloor$ local link steps is required to broadcast inside a supernode. The particular node that starts the broadcasting in a ring is either a node that originated a piece of information to be broadcast or a node that has just been informed of it via a lateral link. In this case, the node may be assumed as the first informed node in a ring, and therefore proceeds with transmissions over different local links in the next two steps. Note that the MIMD model is implicit in Figure 5 since the nodes use different local links in some steps of the algorithm.

Theorem 3 *A one-port broadcasting algorithm for a MIMD n -SCC graph based on the cyclic sequence σ_c and using parallel transmissions over both lateral and local links requires $|\sigma_c|$ steps, where:*

$$|\sigma_c| = \left\lfloor \frac{n+2}{2} \right\rfloor \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

Proof: The number of lateral link steps in σ_c is $|\sigma_c(lat)| = d_{star} = \lfloor 3(n-1)/2 \rfloor$. The number of local link steps in σ_c is $|\sigma_c(loc)| = \lfloor n/2 \rfloor d_{star} = \lfloor n/2 \rfloor \lfloor 3(n-1)/2 \rfloor$. The total number of steps required to run sequence σ_c is then: $|\sigma_c| = |\sigma_c(lat)| + |\sigma_c(loc)| = \lfloor (n+2)/2 \rfloor \lfloor 3(n-1)/2 \rfloor$. \square

In the case of a SIMD n -SCC graph, broadcasting inside a supernode can be accomplished in $n - 2$ steps by forcing all nodes to use either the left or the right local link in a lock-step fashion (Figure 6). The number of steps required by a one-port broadcasting algorithm for a SIMD n -SCC graph can be obtained similarly as done for a MIMD n -SCC in the previous theorem, being equal to $(n - 1) \lfloor 3(n - 1)/2 \rfloor$. In the remaining of this paper, we will consider only the MIMD model of computation. Extensions of our results to the case of a SIMD n -SCC can be easily obtained.

Table 1 lists the number of steps required by a one-port broadcasting algorithm using the cyclic sequence σ_c , according to Theorem 3. The efficiency of the broadcasting algorithm is measured by comparing the required number of steps with the diameter of the graph. We present such a comparison in Table 1 in terms of percentages. More specifically, we define the *relative distance to the diameter* as:

$$\frac{|\sigma_c| - d_{SCC}}{d_{SCC}} \times 100\%$$

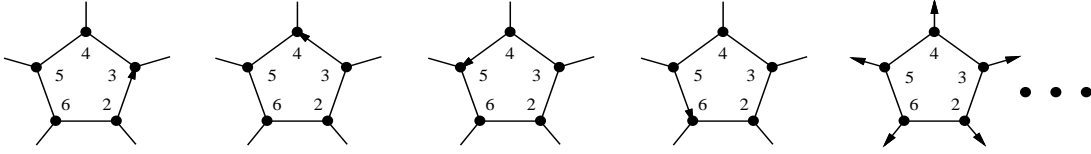


Figure 6: One-port broadcasting in a SIMD 6-SCC (cyclic sequence)

Note that the broadcasting algorithm based on σ_c contains about $1.5n$ lateral link steps and about $0.75n^2$ local link steps (Theorem 3). Such an algorithm is optimal from the viewpoint of lateral links steps and is also close to the minimal number of local link steps required by an ideal one-port broadcasting algorithm for the n -SCC graph (about $0.5n^2$ steps). As a matter of fact, for $4 \leq n \leq 8$ the total number of steps required by the algorithm based on σ_c is just 12.5% to 50% greater than the diameter of the n -SCC graph, which suggests that the algorithm is also very close to optimality regarding the number of local links steps.

Optimality from the viewpoint of lateral links is particularly desired in implementations using faster transmission rates in the local links than in the lateral links. In such cases, broadcasting algorithms with a quadratic number of lateral link steps would perform poorly.

n	Size of n -SCC graph	Diameter of n -SCC graph	Lateral link steps	Local link steps	Total number of steps	Relative distance to the diameter
4	72	8	4	8	12	50%
5	480	16	6	12	18	12.5%
6	3600	19	7	21	28	47.4%
7	30240	31	9	27	36	16.1%
8	282240	34	10	40	50	47.1%

Table 1: Number of steps required by the broadcasting sequence σ_c (one-port version)

We now present a synchronous algorithm to accomplish one-port broadcasting in an SCC graph using sequence σ_c . Each node keeps a set of local variables that are required for proper operation of the algorithm. These variables are:

- INFORMED - a boolean variable that is set to TRUE if the node has already received the broadcast message. It is assumed that the node originating the message has its variable INFORMED initialized to TRUE, while the remaining nodes in the graph have INFORMED initialized to FALSE.
- DONE_WITH_LATERALS - a boolean variable that is set to TRUE if an informed node has already accomplished all lateral link transmissions required to broadcast a particular message to its neighbors.
- DONE_WITH_LOCALS - a boolean variable that is set to TRUE if an informed node has already accomplished all local link transmissions required to broadcast a particular message to its neighbors.
- MESSAGE_RECEIVED_THROUGH - a variable that indicates the port through which a previously uninformed node first received the broadcast message. Three possible values may be assigned to this variable: lateral_link, right_local_link or left_local_link. The node originating the broadcast message has MESSAGE_RECEIVED_THROUGH initialized to lateral_link.

The algorithm also uses procedures to send and receive messages, namely:

- SEND(port) - this procedure is called by an informed node to send the broadcast message using one of 3 possible ports: lateral_link, right_local_link or left_local_link.
- MESSAGE_RECEIVED - this procedure checks the reception of the broadcast message by an uninformed node. If no message is received, the procedure returns FALSE. If the message is received, the procedure returns TRUE and sets MESSAGE_RECEIVED_THROUGH to indicate the port that first brought the message to the uninformed node. If the node receives the message simultaneously in more than one port, then MESSAGE_RECEIVED_THROUGH is arbitrarily set to any of the ports currently bringing the broadcast message to the node.

Algorithm 1 (One-port broadcasting in the SCC):

```
DONE_WITH_LATERALS := FALSE;
DONE_WITH_LOCALS := FALSE;
for  $i := 1$  to  $\lfloor 3(n-1)/2 \rfloor$  do
begin
  for  $j := 1$  to  $\lfloor n/2 \rfloor$  do
  begin
    if (not DONE_WITH_LOCALS) and (INFORMED) then
    begin
      if ( $j = 1$ ) then SEND(right_local_link)
      else
      begin
        case (MESSAGE_RECEIVED_THROUGH) of
          lateral_link: SEND(left_local_link);
          right_local_link: SEND(left_local_link);
          left_local_link: SEND(right_local_link)
        end;
        DONE_WITH_LOCALS := TRUE
      end
    end;
    if (not INFORMED) then
      if (MESSAGE_RECEIVED) then INFORMED := TRUE
    end;
  if (not DONE_WITH_LATERALS) and (INFORMED) then
  begin
    SEND(lateral_link);
    DONE_WITH_LATERALS := TRUE
  end;
  if (not INFORMED) then
    if (MESSAGE_RECEIVED) then INFORMED := TRUE
  end;
end;
```

Message Pipelining with Algorithm 1

A straightforward modification of Algorithm 1 allows broadcasting of B consecutive messages in the SCC graph in a pipeline fashion. Among other applications, pipelined broadcast can be used to input a set of data values required by a parallel algorithm running on the SCC graph. Such can be done by choosing any node in the graph as the source for the data stream. After the execution of the algorithm, all nodes must select among the stream of broadcast messages the correct value sent along with it. Once such a selection is done, each node in the graph contains the proper value required by the particular parallel algorithm being considered.

A pipelined version of Algorithm 1 can be implemented as follows. The variables used by the algorithm can be kept in arrays INFORMED[1.. B], DONE_WITH_LATERALS[1.. B], DONE_WITH_LOCALS[1.. B] and MESSAGE_RECEIVED_THROUGH[1.. B]. The external loop of the algorithm must also be modified to run for $\lfloor 3(n-1)/2 \rfloor + B - 1$ iterations. Thus, during each of the first B iterations of the main loop the node originating the broadcasting will input one different message in the graph.

The MESSAGE_RECEIVED procedure must in this case provide proper memory allocation mechanisms to store the incoming messages at different positions. A simple implementation could be an array with the capacity to store B messages. The index of the array can be easily implemented as a count of received messages (k), initially set to 0 and incremented at each new message arrival. Such an index could be passed as a parameter to MESSAGE_RECEIVED, which would store the currently received message and also set MESSAGE_RECEIVED_THROUGH[k] accordingly. Message counting is also used as the index for variable arrays DONE_WITH_LATERALS[], DONE_WITH_LOCALS[] and INFORMED[]. Finally, the SEND(port) procedure may be modified to accept an additional parameter (e.g. the index k) indicating which message should be transmitted.

With this approach, broadcasting of B pipelined messages using a one-port communication model can be accomplished in $\lfloor (n+2)/2 \rfloor \lfloor B-1+3(n-1)/2 \rfloor$ steps. Hence, there is a latency of $\lfloor (n+2)/2 \rfloor \lfloor 3(n-1)/2 \rfloor$ steps before the broadcasting of the first message is completed. After that, broadcasting of the remaining messages in the sequence is concluded at a rate of $\lfloor (n+2)/2 \rfloor$ steps/message.

3.2 Multiple-port Broadcasting in the SCC

Multiple-port broadcasting algorithms for the SCC can be built as an extension of the previous one-port algorithms. We recall that for $n > 3$, the n -SCC graph has a fixed degree $\delta = 3$, i.e. in an n -SCC we can have at most a 3-port broadcasting algorithm. Another concern that arises while running a multiple-port broadcasting algorithm in the SCC is a possible difference between the transmission rate in the lateral and local links. In a particular broadcasting algorithm requiring simultaneous use of lateral and local links, the amount of time required to run each step of the algorithm is determined by the type of link with lowest transmission rate.

Notably, an efficient multiple-port execution of σ_c can be achieved by using both local links of each node simultaneously while broadcasting the information inside each supernode. Figure 7 shows such a technique for one of the supernodes of a 6-SCC. Note that the approach used for multiple-port broadcasting fits the SIMD computational model, since every node uses exactly the same links at every step of the algorithm. Naturally, the approach depicted on Figure 7 can be used in a MIMD n -SCC in an equally efficient manner.

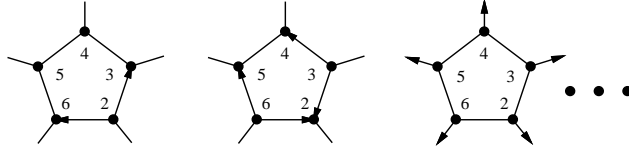


Figure 7: Multiple-port broadcasting in a 6-SCC (cyclic sequence)

Theorem 4 *A multiple-port broadcasting algorithm for a SIMD or MIMD n -SCC graph based on the cyclic sequence σ_c and using parallel transmissions over both lateral and local links requires $|\sigma_c(mult)|$ steps, where:*

$$|\sigma_c(mult)| = \left\lfloor \frac{n+1}{2} \right\rfloor \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

Proof: The number of lateral link steps in a multiple-port execution of σ_c is the same as in the one-port case, i.e. $|\sigma_c(lat, mult)| = |\sigma_c(lat)| = d_{star} = \lfloor 3(n-1)/2 \rfloor$. The number of local link steps is however reduced to $|\sigma_c(loc, mult)| = \lfloor (n-1)/2 \rfloor d_{star} = \lfloor (n-1)/2 \rfloor \lfloor 3(n-1)/2 \rfloor$. The total number of steps required to run sequence σ_c using a multiple-port communication model is therefore $|\sigma_c(mult)| = |\sigma_c(lat, mult)| + |\sigma_c(loc, mult)| = \lfloor (n+1)/2 \rfloor \lfloor 3(n-1)/2 \rfloor$. \square

Table 2 lists the number of steps required by a multiple-port broadcasting algorithm using the cyclic sequence σ_c , according to Theorem 4. Note that the total number of steps required by the algorithm based on sequence σ_c ($\approx 0.75n^2$) is very close to the diameter of the n -SCC graph (actually, the relative distance to the diameter is at most 17.6% for $4 \leq n \leq 8$). We have proved that σ_c is optimal from the viewpoint of lateral link steps, and by inspecting Table 2 we note that optimality from the viewpoint of local link steps is very likely to have already been achieved.

n	Size of n -SCC graph	Diameter of n -SCC graph	Lateral link steps	Local link steps	Total number of steps	Relative distance to the diameter
4	72	8	4	4	8	0%
5	480	16	6	12	18	12.5%
6	3600	19	7	14	21	10.5%
7	30240	31	9	27	36	16.1%
8	282240	34	10	30	40	17.6%

Table 2: Number of steps required by the broadcasting sequence σ_c (multiple-port version)

A comparison of Tables 1 and 2 shows that for odd n , the one-port broadcasting algorithm based on sequence σ_c performs as well as its multiple-port counterpart. However, for even n the number of steps required by one-port broadcasting is about 50% greater than the diameter of the n -SCC graph. Therefore, it is more efficient to use multiple-port broadcasting in this case.

A synchronous algorithm to perform multiple-port broadcasting in an SCC graph using sequence σ_c follows. The variables and functions used by the multiple-port algorithm have the same functionality previously defined for the one-port version. However, an additional procedure is used for multiple-port broadcasting, namely SEND_MULTIPLE(port1,port2). This procedure is called by an informed node to send the broadcast message simultaneously in two of 3 possible ports: lateral_link, right_local_link or left_local_link. As a matter

of fact, this procedure is always called as `SEND_MULTIPLE(right_local_link, left_local_link)` in the proposed multiple-port broadcasting algorithm.

Also, note that the variable `MESSAGE_RECEIVED_THROUGH` is not used by the multiple-port broadcasting algorithm. Therefore, the `MESSAGE_RECEIVED` procedure is also simpler in this case, since recording of the port through which the broadcast message has been received is not required.

Algorithm 2 (Multiple-port broadcasting in the SCC):

```

DONE_WITH_LATERALS := FALSE;
DONE_WITH_LOCALS := FALSE;
for  $i := 1$  to  $\lfloor 3(n-1)/2 \rfloor$  do
begin
  for  $j := 1$  to  $\lfloor (n-1)/2 \rfloor$  do
  begin
    if (not DONE_WITH_LOCALS) and (INFORMED) then
    begin
      SEND_MULTIPLE(right_local_link, left_local_link);
      DONE_WITH_LOCALS := TRUE
    end;
    if (not INFORMED) then
      if (MESSAGE_RECEIVED) then INFORMED := TRUE
    end;
    if (not DONE_WITH_LATERALS) and (INFORMED) then
    begin
      SEND(lateral_link);
      DONE_WITH_LATERALS := TRUE
    end;
    if (not INFORMED) then
      if (MESSAGE_RECEIVED) then INFORMED := TRUE
    end;
end;

```

Message Pipelining with Algorithm 2

Broadcasting of B pipelined messages can be supported in a multiple-port communication model using the same mechanisms previously discussed for one-port broadcasting. In addition to the modifications proposed for one-port broadcasting, the `SEND_MULTIPLE(port1,port2)` procedure may also be modified to accept an index or pointer to one of the messages in the pipeline. With this approach, broadcasting of B pipelined messages using a multiple-port communication model can be accomplished in $\lfloor (n+1)/2 \rfloor \lfloor B-1+3(n-1)/2 \rfloor$ steps. Therefore, there is a latency of $\lfloor (n+1)/2 \rfloor \lfloor 3(n-1)/2 \rfloor$ steps before the broadcasting of the first message is completed. After that, broadcasting of the remaining messages in the sequence is concluded at a rate of $\lfloor (n+1)/2 \rfloor$ steps/message.

4 $O(n)$ Broadcasting Algorithms for the n -SCC

Both versions of the broadcasting algorithm based on sequence σ_c require $O(n)$ lateral link steps and $O(n^2)$ local link steps. We can actually accomplish broadcasting in $O(n)$ running time by making proper assumptions on the time spent by the algorithm on lateral and local link steps. As an example, assume that we want to have the total running time of a broadcasting algorithm equally divided between lateral and local link steps. This results in the following theorem:

Theorem 5 *Both one-port and multiple-port broadcasting using sequence σ_c in an n -SCC graph have linear running time if the transmission rate in the local links is $O(n)$ times faster than the transmission rate in the lateral links.*

Proof: Suppose that the transmission rates on the lateral and the local links are respectively $TR(lat)$ and $TR(loc)$. If we want an even distribution of time over lateral and local link steps, then we may choose:

$$\frac{\sigma_c(lat)}{TR(lat)} = \frac{\sigma_c(loc)}{TR(loc)}$$

The ratio between $TR(loc)$ and $TR(lat)$ is then:

$$\frac{TR(loc)}{TR(lat)} = \begin{cases} \lfloor \frac{n}{2} \rfloor & , \text{ for one-port broadcasting} \\ \lfloor \frac{n-1}{2} \rfloor & , \text{ for multiple-port broadcasting} \end{cases}$$

In both cases, the result is that the time spent on a quadratic number of local link steps can be made equal to that spent on a linear number of lateral link steps, as long as the transmission rate in the local links is $O(n)$ times faster than the transmission rate in the lateral links. If we suppose that the broadcasting algorithm spends most of the time transmitting data (i.e., the overhead or start-up time associated with the messages is small when compared to the time required to transmit them), then the resulting running time is linear with n and equal to $2d_{star}/TR(lat)$. \square

5 Comparison of Broadcasting Algorithms for the SCC and the Star Graphs

A comparison of broadcasting algorithms for the SCC and the star graphs is presented in Table 3. Table 3 lists both one-port and multiple-port algorithms, assuming the broadcasting sequence σ_c in the case of the SCC graph. We also assume in this case that the transmission rates in the local and lateral links of the graph meet the conditions described in Theorem 5.

One-port broadcasting in the n -star uses the optimal $O(n \log n)$ algorithm proposed in [11]. For multiple-port broadcasting in the n -star, we use the δ -port communication model proposed in Theorem 2.

From the viewpoint of the relative distance to the diameter, it seems that one-port broadcasting is accomplished more efficiently in the SCC than in the star graph. However, we recall that the longer diameter of the SCC graph limits the efficiency of broadcasting in the graph. Due to this reason, the star graph actually outperforms a SCC graph of similar size in respect to the total number of steps required by a one-port broadcasting algorithm.

In an implementation where the assumptions of Theorem 5 hold, one-port broadcasting in the n -SCC graph can be accomplished in $O(n)$ running time. In this case, Table 3 indicates that one-port broadcasting in an

n -SCC graph requires a running time better than or equal to that of an n -star containing $(n - 1)$ times fewer nodes.

Multiple-port broadcasting can be accomplished in $O(n)$ running time in both graphs, but the star graph requires fewer steps than does an SCC graph of similar size. However, a clear advantage provided by the n -SCC graph is that an $O(n)$ multiple-port broadcasting algorithm requires each node to transmit over at most two links at a time. On the other hand, the $O(n)$ multiple-port broadcasting algorithm pictured in Table 3 for the n -star graph may impose excessive overhead on the nodes, since it requires simultaneous transmissions over the $(n - 1)$ ports of each node.

Broadcasting algorithm & graph type	n	Graph size	Graph diameter	Lateral link steps	Local link steps	Total number of steps	Running time in lateral link steps	Relative distance to the diameter
<i>One-port broadcasting in the n-SCC</i>	4	72	8	4	8	12	8	50%
	5	480	16	6	12	18	12	12.5%
	6	3600	19	7	21	28	14	47.4%
	7	30240	31	9	27	36	18	16.1%
	8	282240	34	10	40	50	20	47.1%
<i>One-port broadcasting in the n-star</i>	5	120	6	12	-	12	12	100%
	6	720	7	16	-	16	16	129%
	7	5040	9	20	-	20	20	122%
	8	40320	10	24	-	24	24	140%
	9	362880	12	29	-	29	29	142%
<i>Multiple-port broadcasting in the n-SCC</i>	4	72	8	4	4	8	8	0%
	5	480	16	6	12	18	12	12.5%
	6	3600	19	7	14	21	14	10.5%
	7	30240	31	9	27	36	18	16.1%
	8	282240	34	10	30	40	20	17.6%
<i>Multiple-port broadcasting in the n-star</i>	5	120	6	6	-	6	6	0%
	6	720	7	7	-	7	7	0%
	7	5040	9	9	-	9	9	0%
	8	40320	10	10	-	10	10	0%
	9	362880	12	12	-	12	12	0%

Table 3: Comparison of broadcasting algorithms for the SCC and the star graphs

6 Comparison of Broadcasting Algorithms for the SCC and the CCC Graphs

A comparison of one-port broadcasting algorithms for the SCC and the CCC graphs is presented in Table 4. It is assumed that Algorithm 1 is used in the case of the SCC graph. A one-port broadcasting algorithm requiring $\lceil 5n/2 \rceil - 1$ steps [9] is used for the CCC graph. We assume that each ring in the CCC graph contains n nodes, resulting in a graph of size $n2^n$. In addition, the diameter of an n -CCC graph is given by $2n + \lfloor n/2 \rfloor - 2$ [12].

Broadcasting in the CCC is initially accomplished with alternated transmissions on local and lateral

links³. After $2n - 1$ such steps, reception of the broadcast message by at least one node in every ring of the CCC is guaranteed. The remaining steps of the algorithm use only local link transmissions to ensure proper distribution of the broadcast message to all nodes in each ring of the CCC. A total of n lateral link steps and $\lceil 3n/2 \rceil - 1$ local link steps is used by the algorithm.

For a more complete comparison with the SCC graph, we also consider the possibility of having the transmission rate in the local links of the n -CCC to be $O(n)$ times faster than the transmission rate in the lateral links. For similarity with the SCC graph, we assume that the ratio between the transmission rates in local and lateral links of the n -CCC graph is $\lfloor n/2 \rfloor$.

<i>Graph</i>	<i>n</i>	<i>Graph size</i>	<i>Graph diameter</i>	<i>Lateral link steps</i>	<i>Local link steps</i>	<i>Total number of steps</i>	<i>Running time in lateral link steps</i>	<i>Relative distance to the diameter</i>
<i>n-SCC</i>	4	72	8	4	8	12	8	50%
	5	480	16	6	12	18	12	12.5%
	6	3600	19	7	21	28	14	47.4%
	7	30240	31	9	27	36	18	16.1%
	8	282240	34	10	40	50	20	47.1%
<i>n-CCC</i>	4	64	8	4	5	9	6.5	12.5%
	6	384	13	6	8	14	8.7	7.7%
	9	4608	20	9	13	22	12.3	10%
	11	22528	25	11	16	27	14.2	8%
	14	229376	33	14	20	34	16.9	3%

Table 4: Comparison of one-port broadcasting algorithms for the SCC and the CCC graphs

One-port broadcasting is accomplished more efficiently in the CCC graph than in the SCC. Note that for graphs of similar size as listed in Table 4 the relative distance to the diameter of one-port broadcasting algorithms range from 12.5% to 50% in the case of the SCC graph, while for the CCC this range is 3% to 12.5%. Also, one-port broadcasting is accomplished in fewer steps in the CCC when compared to a SCC of similar size. The difference in performance is more evident for even n in the case of the n -SCC. Such can be explained by the fact that the quotient graph of the CCC (i.e., the hypercube) allows for an optimal broadcasting algorithm [2], [9], [10] which can be nicely extended to the case of the CCC [9]. The same is not true in the case of the SCC and its quotient graph (the star).

If an $O(n)$ times faster transmission rate is used in the local links of the SCC and the CCC when compared to the transmission rate in the lateral links, a smaller difference in performance results. However, even in this case the CCC is still superior to the SCC in respect to the running time measured in lateral link steps.

Using multiple-port broadcasting reduces the difference in performance between the CCC and the n -SCC for even n . It is also likely that multiple-port communication may reduce the number of steps required by broadcasting in the CCC. However, the amount of improvement should be small since one-port broadcasting in the CCC already requires only 1 or 2 steps more than the diameter of the graph.

Although the CCC shows some superiority in respect to the number of steps required for broadcasting, note that the SCC requires less nodes per supernode than a CCC graph of similar size does. Implementation

³By analogy with the SCC, *local links* in a CCC graph connect nodes inside a ring or supernode, while *lateral links* correspond to a dimension link of the quotient hypercube.

of supernodes as single multiprocessor VLSI devices is therefore less complex in the case of the SCC graph. Such can be observed in Table 4 by recalling that an n -SCC has $(n - 1)$ nodes/supernode while an m -CCC has m or more nodes/supernode. In addition, we have $m > n$ if an n -SCC is being compared with an m -CCC of similar size.

7 Conclusion

In this paper, we have presented one-port and multiple-port broadcasting algorithms for the star-connected cycles (SCC) graph, considering both the SIMD and the MIMD computational models.

We have shown that efficient $O(n \log n)$ one-port broadcasting algorithms devised for the n -star graph cannot be efficiently mapped onto an n -SCC graph due to their sequential nature of execution.

Interestingly, we have shown that an $O(n^2)$ one-port cyclic broadcasting sequence originally proposed for the n -star graph can be efficiently executed in the n -SCC by using parallel transmissions over the lateral and local links of the graph. Also, if the transmission rate in the local links of an n -SCC graph is $O(n)$ times faster than the transmission rate in the lateral links, then it is possible to accomplish both one-port and multiple-port broadcasting in $O(n)$ time.

The proposed broadcasting algorithms are optimal from the viewpoint of lateral link steps and also seem to be close to optimality from the viewpoint of local link steps. Particularly for $4 \leq n \leq 8$, the total number of steps required by a multiple-port broadcasting algorithm based on a cyclic sequence of digits is at most 17.6% greater than the diameter of the corresponding n -SCC graph.

We have also compared the efficiency of broadcasting algorithms for the n -star and the n -SCC graph and concluded that one-port broadcasting in an n -SCC graph can be accomplished in running time better than or equal to that of an n -star containing $(n - 1)$ times fewer nodes.

The running time required by the SCC is greater than that of the star in the case of multiple-port broadcasting algorithms. However, we may still claim that the SCC is superior to the star graph in regard to multiple-port broadcasting if aspects such as the communication overhead at each node are taken into account.

A comparison between SCC and CCC graphs of similar size indicates that one-port broadcasting requires less steps in the case of the CCC graph. However, the difference in performance is reduced if a faster transmission rate in the local links is used in both graphs when compared to the transmission rates in the lateral links. In addition, the SCC requires less nodes per supernode than a CCC graph of similar size does, which may simplify the implementation of supernodes.

Finally, we have shown that the proposed broadcasting algorithms can be easily extended to support transmission of a sequence of messages in pipelined fashion.

Acknowledgements

Thanks are due to Referee C for bringing Reference [9] to our attention.

References

- [1] Y. Saad and M. H. Schultz, "Topological Properties of Hypercubes," *IEEE Transactions on Computers*, Vol. 37, No. 7, July 1988, pp. 867-872.

Appears in *Journal of Parallel and Distributed Computing*, 25, 209-222 (1995).

- [2] S. B. Akers, D. Harel and B. Krishnamurthy, "The Star graph: An Attractive Alternative to the n -Cube," *Proc. Int'l Conf. on Parallel Processing*, 1987, pp. 393-400.
- [3] F. P. Preparata and J. Vuillemin, "The Cube-Connected Cycles: A Versatile Network for Parallel Computation," *Communications of the ACM*, Vol. 24, No. 5, May 1981, pp. 300-309.
- [4] S. Latifi, M. M. Azevedo and N. Bagherzadeh, "The Star-Connected Cycles: a Fixed-Degree Interconnection Network for Parallel Processing," *Proc. Int'l Conf. on Parallel Processing*, 1993, Vol. 1, pp. 91-95.
- [5] S. B. Akers and B. Krishnamurthy, "A Group-Theoretic Model for Symmetric Interconnection Networks," *IEEE Transactions on Computers*, Vol. 38, No. 4, April 1989, pp. 555-566.
- [6] S. Latifi, "Parallel Dimension Permutations on Star Graph," *IFIP Transactions A: Computer Science and Technology*, 1993, A23, pp. 191-201.
- [7] D. E. Knuth, *The Art of Computer Programming, Vol. 1*, Addison-Wesley, 1968, pp. 73, pp. 176-177.
- [8] M.M. Azevedo, N. Bagherzadeh and S. Latifi, *The Star-Connected Cycles: a Fixed-Degree Interconnection Network for Massively Parallel Systems*, Department of Electrical and Computer Engineering, University of California, Irvine, Technical Report ECE 93-02, March 1993.
- [9] A. L. Liestman and J. G. Peters, "Broadcast Networks of Bounded Degree," *SIAM Journal on Discrete Mathematics*, Vol. 1, No. 4, November 1988, pp. 531-540.
- [10] S. L. Johnson and C. T. Ho, "Optimum Broadcasting and Personalized Communications in Hypercubes," *IEEE Transactions on Computers*, Vol. 38, No. 9, September 1989, pp. 1249-1268.
- [11] V.E. Mendia and D. Sarkar, "Optimal Broadcasting on the Star Graph," *IEEE Transactions on Parallel and Distributed Systems* Vol. 3, No. 4, July 1992, pp. 389-396.
- [12] D.S. Meliksetian and C.Y.R. Chen, "Communication Aspects of the Cube-Connected Cycles," *Proc. Int'l. Conf. Parallel Processing*, Vol. 1, 1990, pp. 579-580.