

**The Star-Connected Cycles: a Fixed-Degree
Interconnection Network for Massively Parallel Systems**

Marcelo Moraes de Azevedo and Nader Bagherzadeh
Department of Electrical and Computer Engineering
University of California, Irvine – Irvine, CA 92717

Shahram Latifi
Department of Electrical and Computer Engineering
University of Nevada, Las Vegas – Las Vegas, NV 89154-4026

Technical Report ECE 93-02 - March 1993

The Star-Connected Cycles: a Fixed-Degree Interconnection Network for Massively Parallel Systems

Marcelo Moraes de Azevedo and Nader Bagherzadeh*

Department of Electrical and Computer Engineering
University of California, Irvine – Irvine, CA 92717
email: mazevedo@balboa.eng.uci.edu, nader@balboa.eng.uci.edu
Phone: (714) 856-8720 FAX: (714) 856-4152

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 — *This technical report describes a new interconnection network for massively parallel systems referred to as star-connected cycles (SCC) graph. The SCC presents an I/O-bounded, fixed-degree structure that results in several advantages over variable-degree graphs like the star graph and the n -cube.*

The description of the SCC graph given in this report includes issues such as labeling of nodes, degree, diameter, symmetry, fault tolerance and Cayley graph representation. This report also presents an optimal routing algorithm for the SCC and efficient broadcasting algorithms with $O(n)$ running time. A comparison with the cube-connected cycles (CCC) and other interconnection networks is included, indicating that for even n , an n -SCC and a CCC of similar sizes have about the same diameter. Also, we show that one-port broadcasting in an n -SCC graph can be accomplished with a running time better than or equal to that required by an n -star containing $(n - 1)$ times fewer nodes.

Index terms — *Fixed-degree graphs, I/O-Bounding, Interconnection networks, Parallel processing, Routing, Broadcasting, Star graph.*

1 Introduction

Over the past years, many interesting graphs such as the n -star [1] and the n -cube [2] have been proposed as interconnection networks for parallel processing applications. Some important properties shown by these graphs are node and edge symmetry, hierarchical structure, maximal fault tolerance and strong resilience [1], [3]. However, the n -star is superior to the n -cube as far as the degree and diameter are concerned. The n -star also has a shorter average distance and fault diameter than does a hypercube with similar number of nodes.

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

Most graphs studied so far offer a high processor density while keeping the diameter as low as possible. Nevertheless, graphs such as the n -star and the n -cube present a variable-degree structure and have low scalability from the viewpoint of network growth. More specifically, since the degree of the n -star and the n -cube is respectively equal to $(n - 1)$ and n , a growing number of communication links is required as n increases. Hence, one disadvantage posed by variable-degree interconnection networks is the large number of I/O communication ports required at each processor in massively parallel systems. Variable-degree interconnection networks also present more complex physical layouts and require additional communication ports at each processor to be expanded. In other words, if we want to increase the number of nodes of an existing variable-degree parallel system, it might be necessary to use processors with additional I/O ports, unless unused communication ports are available at each node.

To overcome these difficulties, we propose a new type of interconnection network: the *star-connected cycles (SCC)* graph [4]. The SCC offers an I/O-bounded, fixed-degree structure and can be viewed as an evolution of its counterpart, the cube-connected cycles or CCC [5].

The SCC and CCC graphs are formed by connecting cycles or rings of nodes through a particular network communication topology. The underlying topology used to connect the cycles in an n -SCC graph is an n -star, while that of the n -CCC graph is the n -cube. As expected, this results in a fixed-degree interconnection network that is superior to the CCC with respect to several characteristics.

In this report, we define the SCC graph and present issues related to the labeling of nodes, diameter, symmetry, fault tolerance, Cayley graph representation and routing. It is shown that although the diameter of the SCC graph is greater than that of an n -star and a hypercube of similar size, it is possible to use a selection of communication links that yields low message transmission delays.

Also included in this report is an analysis of different broadcasting algorithms for the SCC graph. We initially analyze how an efficient $O(n \log n)$ algorithm that has been proposed for the n -star graph can be extended to an n -SCC. We show that such algorithm does 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 [1] 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 show that for $4 \leq n \leq 8$ the number of steps required to run an efficient 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. In addition, we also show how broadcasting algorithms for the SCC graph can be extended to support transmission of a sequence of messages in a pipeline fashion.

Finally, we compare the SCC with the hypercube, the star and CCC graphs and claim that the SCC is an interesting alternative for parallel systems.

2 Background - The Star Graph

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

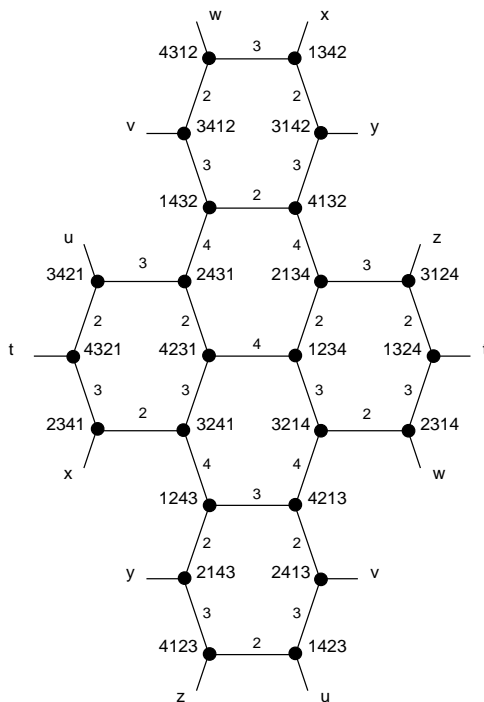


Figure 1: 4 -star graph

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 $2 \leq j \leq n$ [1], [6]. A 4 -star graph is shown in Figure 1.

An n -star graph is a regular graph with degree $\delta = n - 1$ and fault tolerance $f = \delta - 1 = n - 2$. The n -star graph is maximally fault-tolerant and strongly resilient [3]. Also, the n -star graph shows vertex and edge symmetry, hierarchical structure and simple routing. Finally, the n -star presents a low diameter, given by $d_{star} = \lceil 3(n - 1)/2 \rceil$ [1].

3 Description of the SCC

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 [6], [7]. We consider that the nodes in an n -star are labeled with a permutation of the digits $\{1, 2, \dots, n\}$, which allows the labeling of $n!$ different nodes in the n -star. The P_i permutation 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 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. The label of the lateral link is chosen so as to represent the position of the digit in the permutation P_i that is swapped with the first digit of P_i when the lateral link is traversed (i.e., I_j is a dimension of the n -star).

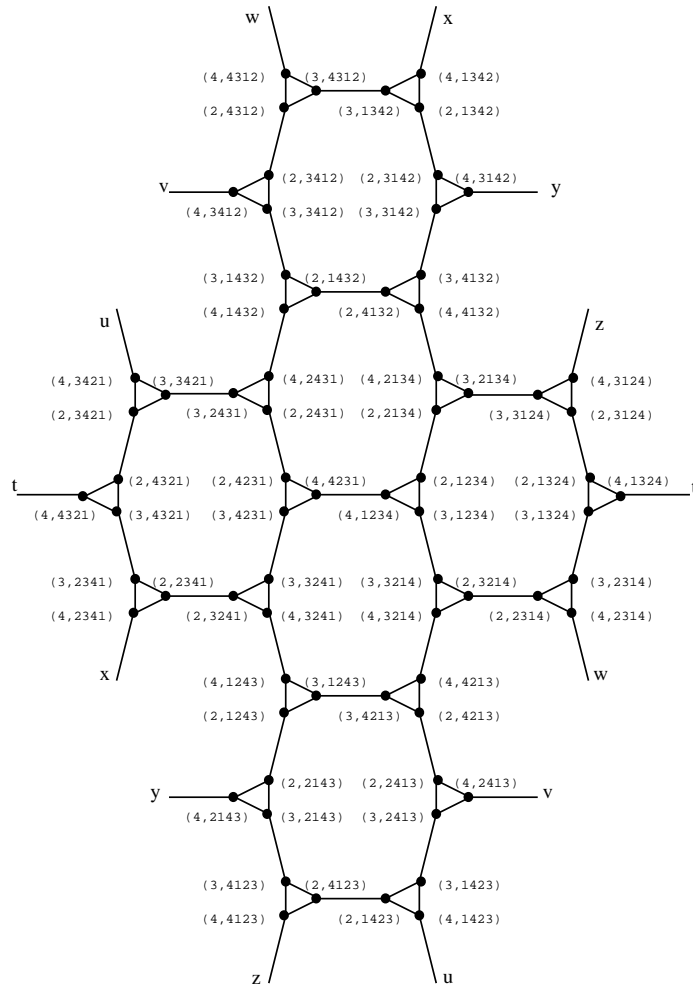


Figure 2: 4-SCC graph

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 4-SCC and are connected to other supernodes using lateral links 2, 3 and 4 as follows:

- (2,1234) is connected to (2,2134) via lateral link 2
- (3,1234) is connected to (3,3214) via lateral link 3
- (4,1234) is connected to (4,4231) via lateral link 4

4 Properties of the SCC

Number of nodes

An n -SCC graph can be seen as an n -star graph connecting $n!$ supernodes. Since each supernode contains $(n - 1)$ nodes, the total number of nodes in an n -SCC graph is $N = (n - 1)n!$.

Degree

The n -SCC graph has a degree of:

$$\delta = \begin{cases} 3 & , \text{ for all } n > 3 \\ 2 & , \text{ for } n = 3 \\ 1 & , \text{ for } n = 2 \end{cases}$$

Since the degree of every node in the graph is the same, the n -SCC graph is regular. For $n > 3$, every node in a n -SCC graph connects to exactly 3 adjacent nodes using two local links within the same supernode and one lateral link to a node belonging to an adjacent supernode.

Keeping the degree low and constant reduces the communication costs at each node. If each node is implemented as an individual processor, we can use a standard building block with 3 communication links to build any n -SCC graph. In an n -star graph, the number of communication links required at each processor changes linearly with n .

Fault tolerance

A graph is *f-fault-tolerant* if it remains connected when any set of f or fewer nodes are removed from the graph [3], [8]. The *fault tolerance* of a graph corresponds to the largest f for which the graph is f -fault tolerant.

The fault tolerance of a graph with degree δ can be at most equal to $(\delta - 1)$, since if we remove all the neighbors of a node the graph will be disconnected. A graph whose fault tolerance is exactly $(\delta - 1)$ is said to be *maximally fault-tolerant*.

The fault tolerance of an n -SCC graph is:

$$f = \begin{cases} 2 & , \text{ for all } n > 3 \\ 1 & , \text{ for all } n = 3 \\ 0 & , \text{ for } n = 2 \end{cases}$$

It is clear that the n -SCC graph is maximally fault-tolerant.

Symmetry

The SCC graph is node symmetric. This is true both from the viewpoint of a single node and from the viewpoint of a supernode. So, given any two nodes a and b there is an automorphism of the graph that maps a to b [9]. Node symmetry is an interesting property since it allows the utilization of identical processors in a SCC-based parallel computer.

Since not all SCC edges look the same, the SCC graph is not edge symmetric. This implies that the communication load is not uniformly distributed over all communication links. However, if we consider only the lateral links and view the n -SCC as an n -star of supernodes, the edge symmetry properties of the n -star still hold.

Thus, every lateral link in an n -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.

Therefore, we may view the n -SCC graph as having two different types of communication links, which allows for an efficient implementation. The local links, for instance, can be implemented as a high-speed bus if the processors that belong to the same supernode are kept physically close to each other. Each supernode can also be implemented as a single multiprocessor VLSI device.

The lateral links used to connect the supernodes can either use a high speed bus, a LAN or other type of serial communication technique, depending on the distance between the supernodes and other criteria such as cost and required performance.

Cayley Graph Representation

Lemma 1: The n -SCC is not a Cayley graph.

Proof: The generators of a Cayley graph labeled with n digits can either generate S_n or a subgroup of S_n [10], where S_n is the set of all $n!$ possible permutations that can be created with n digits. Each node in an n -SCC graph is actually labeled with $(n + 1)$ digits, which means that the labels of an n -SCC graph actually belong to S_{n+1} .

S_{n+1} has $(n + 1)!$ different permutations and the n -SCC graph uses only $(n - 1)n!$ permutations of S_{n+1} . The relation between these numbers is equal to $(n + 1)/(n - 1)$.

According to LaGrange's theorem on finite groups, the order of any subgroup (the number of its elements) always divides the order of the group [10], [11]. That is not the case with the ratio above. \square

Although the n -SCC graph is not a Cayley graph, its node symmetry property allows that any n -SCC graph can be represented as the quotient of two Cayley graphs [6]. More specifically, we may obtain the quotient graph of an n -SCC graph by identifying subgraphs in the n -SCC and reducing such subgraphs to nodes. The nodes in the resulting quotient graph are connected iff there existed an edge between elements of the corresponding subgraphs. In the case of the n -SCC, each subgraph corresponds to a ring of nodes $(2, I_i)$, $(3, I_i)$, \dots , (n, I_i) (i.e., a *supernode*). If we label each node within a subgraph as a permutation of the digits $\{2, 3, \dots, n\}$, then we can identify each subgraph as a Cayley graph that is described by two generators: $3 \dots n2$ and $n2 \dots (n - 1)$. Reducing each subgraph to a node results in a quotient graph. We can easily see that such quotient graph is a well known Cayley graph: the n -star. Thus, we may obtain a compact representation of the n -SCC graph by listing the generators of its subgraph and the corresponding quotient graph (i.e. respectively an $(n-1)$ -ring and an n -star).

5 Routing in the n -SCC

Routing in the n -SCC is an extension of the routing in the n -star graph and can be seen as two different problems: routing in the lateral links and routing in the local links.

5.1 Routing in the Lateral Links

Routing in the lateral links uses the same routing techniques already developed for the n -star. Although the routing algorithm for the n -star can be found in [1] and [12], we repeat it here for clarity.

Suppose that we want to route from P_s to P_d in an n -star, 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 lateral link. To find the lateral links connecting P_s to P_d , we can instead find the path from P_{ds} to the identity permutation [6], where:

$$P_{ds} = P_d^{-1}P_s$$

After calculating P_{ds} , the following routing algorithm applies:

Algorithm 1 (Routing in the n -star):

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.

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 lateral links in the path from P_s to P_d is [1] :

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

Let $C = (i_1 i_2 \dots i_k)$ be a cycle of length $k \leq n$ in P_{ds} , where $1 \leq i_1 < i_2 < \dots < i_k \leq n$. The *execution of cycle C* corresponds to a path R in the n -star and can be expressed as a sequence of lateral links as follows [13]:

$$\begin{aligned} R &= (i_2, i_3, \dots, i_k) & , \text{ if } i_1 = 1 \\ R &= (i_1, i_2, \dots, i_{k-1}, i_k, i_1) & , \text{ if } i_1 \neq 1 \end{aligned}$$

Note that in an n -star there are¹ $N_c = c!$ different choices that can be used for an optimal order of execution of cycles of length at least 2 in P_{ds} . If the number of digits in cycle C_i is K_i , $K_i \geq 2$, then there are also N_i different ways to minimally execute C_i , where:

$$N_i = \begin{cases} K_i & , \text{ if } C_i \text{ does not include digit } 1 \\ 1 & , \text{ if } C_i \text{ includes the digit } 1 \end{cases}$$

If we order the cycles C_i for which $K_i \geq 2$ as C_1, C_2, \dots, C_c then the total number of optimal routing paths in the n -star from P_s to P_d is:

¹Note that this equation is valid even if the first digit in P_{ds} is not 1. Although Algorithm 1 indicates that the cycle including digit 1 should be executed first, we may actually choose any order to execute the cycles, one at a time. This is possible because the execution of any cycle leaves the position of digits that do not belong to that cycle unchanged.

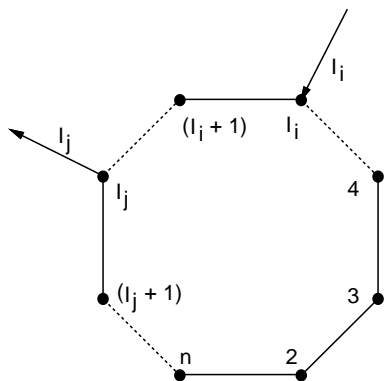


Figure 3: Routing in a supernode

$$T_p = c! \prod_{i=1}^{i=c} N_i$$

As an example, permutation 21453 has two cycles: $C_1 = (1\ 2)$ and $C_2 = (3\ 4\ 5)$. Cycle C_1 is executed with a single swap (2). On the other hand, cycle C_2 may be executed with three different sequences of swaps: (3, 4, 5, 3), (4, 5, 3, 4) or (5, 3, 4, 5). We may therefore execute both cycles as the following sequences of lateral links: (2, 3, 4, 5, 3), (2, 4, 5, 3, 4), (2, 5, 3, 4, 5), (3, 4, 5, 3, 2), (4, 5, 3, 4, 2) or (5, 3, 4, 5, 2).

5.2 Routing in the Local Links

To describe the routing algorithm for the local links, we refer the reader to Figure 3. Let D_n be the minimum number of local links between two nodes I_i and I_j that belong to the same supernode. Then:

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

We assume that the nodes belonging to the same ring are labeled in an ascending order from 2 to n , going counterclockwise (Figure 3). We also assume that the lateral link entering the supernode is I_i and the lateral link leaving the supernode is I_j . Then, the following routing algorithm applies to the local links:

Algorithm 2 (Routing in the local links):

1. Evaluate $L = I_j - I_i$ and $R = |I_j - I_i| \operatorname{div} \lfloor \frac{n+1}{2} \rfloor$, where div is the integer division operator.
2. If $(L > 0 \text{ and } R = 0)$ or $(L < 0 \text{ and } R = 1)$, then take the ring counterclockwise traversing D_n local links.
3. If $(L > 0 \text{ and } R = 1)$ or $(L < 0 \text{ and } R = 0)$, then take the ring clockwise traversing D_n local links.

Notice that the ordering of nodes inside each ring of an n -SCC depends on the physical lay-out of the interconnection network. It is possible, for instance, to lay-out the n -SCC such that some

supernodes have a counterclockwise ordering while other supernodes use a clockwise ordering (as an example, consider the 4-SCC graph in Figure 2). Algorithm 2 can be easily modified for n -SCC graphs that have different types of node orderings if we store in the nodes a clockwise/counterclockwise flag bit. Testing that bit indicates whether Algorithm 2 should proceed as stated above or whether the opposite direction should be taken. In both cases, the number of local links to be traversed is D_n .

5.3 Routing Algorithm for the n -SCC

We now present an algorithm for routing in the n -SCC graph. Such algorithm is actually a combination of Algorithms 1 and 2 and provides a sequence of lateral and local links as a result.

We recall that Algorithm 1 allows for T_p different optimal paths in an n -star graph. The edges of the n -star graph are the lateral links of the corresponding n -SCC graph. However, not all of the T_p different optimal paths that exist in the n -star result in T_p optimal paths in the n -SCC, since the order of execution of the lateral links affects the number of local links in the routing.

The routing algorithm for the n -SCC graph performs a depth-first search on a weighted tree structure. The algorithm builds the tree by expanding at each step those cycle orderings that seem to result in a minimal number of local links. Backtracking is also performed to enable expansion of previous cycle orderings that seem to be equivalent to or better than the most recently expanded orderings.

In the description of the routing algorithm for the n -SCC, we denote the source node by (I_s, P_s) and the destination node by (I_d, P_d) . We also define the following items:

- P_s can be mapped into P_d by a sequence of lateral links or generators of the quotient n -star graph embedded in the n -SCC graph (i.e., $P_s g_1 g_2 \dots g_p = P_d$). Alternatively, we can find the path from P_s to P_d by routing from P_{ds} to the identity permutation, where $P_{ds} = P_d^{-1} P_s$.
- S_c is the set of cycles of length at least 2 that exist in P_{ds} .
- S_d is a subset of the digits included in the cycles of S_c , such that:
 - If $(1 \ i_2 \ i_3 \ \dots \ i_k)$ is a cycle of S_c , then $i_2 \in S_d$ and $1, i_3, \dots, i_k \notin S_d$.
 - If $(i_1 \ i_2 \ \dots \ i_k)$ is a cycle of S_c that does not include digit 1, then $i_1, i_2, \dots, i_k \in S_d$.

The tree structure generated by the routing algorithm has the following characteristics:

- If the number of cycles in S_c is c , then the height of the tree is $(c + 1)$. The first level of the tree is 0 and the deepest level is $(c + 1)$.
- The label of any vertex in the tree is a pair of digits (f, ℓ) , belonging either to S_d or to $\{I_s, I_d\}$. Each vertex located between levels 1 and c in the tree represents one of the cycles of S_c . The label (f, ℓ) is chosen so as to represent the first (f) and last (ℓ) lateral link used during the execution of the cycle.

If the cycle represented by the vertex is $(1 \ i_k)$, then the vertex is labeled $(f, \ell) = (i_k, i_k)$. If the cycle represented by the vertex is $(1 \ i_2 \ i_3 \ \dots \ i_k)$, then the vertex is labeled $(f, \ell) = (i_2, i_k)$.

If the cycle represented by the vertex does not include digit 1, then the vertex may be labeled as $(f, \ell) = (i_i, i_i)$, where i_i is any of the digits of the cycle.

- The weight of an edge connecting any two vertices (f_i, ℓ_i) and (f_j, ℓ_j) corresponds to the number of local links required to move from ℓ_i to f_j within the same supernode and is given by Equation 1.
- Each vertex (f, ℓ) has an associated data structure consisting of its distance to the root (D_r) and a reduced set of digits $S_d^c = S_d^p - S_i$.
The distance D_r is obtained by summing the weights of all edges in the path from the root to the vertex. S_d^p is the set of digits stored in the parent of each vertex and S_i is the set of digits belonging to the cycle that includes digit f .
- The root vertex is $(f, \ell) = (I_s, I_s)$ and has $D_r = 0$ and $S_d^c = S_d$. The vertices located at level $(c + 1)$ in the tree are labeled $(f, \ell) = (I_d, I_d)$ and have $S_d^c = \{\}$. The vertices located at level c in the tree have $S_d^c = \{I_d\}$.
- Each vertex also stores an enable/disable bit that informs whether the tree should continue to be expanded from that vertex or not. The root vertex is created with an enabled bit, but all other child vertices are created with a disabled bit. Intermediate vertices (i.e., vertices that have already been expanded) also have a disabled bit.

Given the definitions above, the routing algorithm for the n -SCC is as follows :

Algorithm 3 (Routing in the n -SCC):

1. If $P_s = P_d$, then route inside the ring using Algorithm 2 and exit.
2. If $P_s \neq P_d$, then calculate permutation P_{ds} such that $P_{ds} = P_d^{-1}P_s$.
3. Identify the cycles of length at least 2 that exist in P_{ds} and create the sets S_c and S_d .
4. Create an enabled root vertex labeled (I_s, I_s) such that $S_d^c = S_d$ and $D_r = 0$.
5. Generate child vertices for all enabled vertices, such that the label f for each child corresponds to exactly one of the digits stored in the set S_d^p of each parent vertex. The label ℓ for each child vertex is chosen according to the definitions of the tree structure given earlier in this section.
6. Evaluate D_r and S_d^c for all child vertices. Check if there is any recently generated child vertex that has a distance $D_n = 0$ to its parent. If such child vertex exists, enable it. Otherwise, enable all recently generated child vertices that have a distance $D_n = 1$ to its parent vertices.

If none of the recently generated child vertices has a distance $D_n \leq 1$ to its parent vertices, then it is necessary to perform a backtracking search in the tree.

The backtracking search enables all vertices that have the smallest *virtual distance* (D_v) to the end of the tree, where:

$$D_v = D_r + c + 1 - h$$

and h is the level at which the vertex is located in the tree ($0 \leq h \leq c + 1$).

7. If all enabled vertices are at level $c + 1$ of the tree, then an optimal order of execution for the cycles in S_c has already been found. Otherwise, return to Step 5.

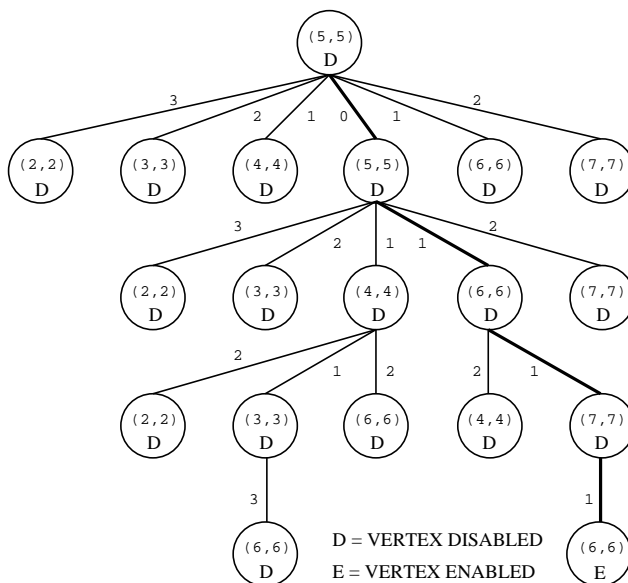


Figure 4: Example of routing tree in the n -SCC

8. The optimal order of execution for the cycles in S_c is given by the intermediate vertices existing between the root and any enabled vertex at level $c + 1$ in the tree.

This optimal order of execution is actually a sequence of lateral links. Additional routing is required to move between lateral links, but that can be easily done with Algorithm 2.

As an example, consider the routing in an 8-SCC for the case $P_{ds} = (1\ 5)(2\ 3\ 6)(4\ 7)(8)$, $I_s = 5$ and $I_d = 6$. Figure 4 shows the tree built by the routing algorithm. An optimal order of execution for P_{ds} is therefore (5, 6, 3, 2, 6, 7, 4, 7).

Algorithm Complexity

The complexity of the routing algorithm depends on the depth of the tree and on the amount of backtracking that might be required during its execution. As at each of the $(c + 1)$ levels of the tree we might have to compare at most n vertices, the complexity of the routing algorithm is equal to or better than $O(cn)$, if no backtracking is performed during its execution.

The algorithm usually executes faster than estimated for the following reasons:

1. Although there might be different optimal paths from the source to the destination node, the execution is stopped as soon as the algorithm finds one optimal path. This behavior is dictated by searching the tree depth-first.
2. The number of cycles that remain to be executed is decremented while the tree is built. This reduces the number of child vertices under each parent as we move deeper in the tree.
3. Many child vertices are created with high weighted edges. Such vertices generally result in longer paths and are kept disabled by the routing algorithm. Even if backtracking occurs during the execution of the algorithm, only part of the vertices located in the upper levels of the tree are enabled.

6 Diameter

Before calculating the diameter of the n -SCC graph, we recall that the diameter of the n -star can be found by evaluating the number of edges between the identity permutation and one of its antipodes. An antipode is the farthest node from a given node along the shortest path. In the n -star, the antipodes are [13]:

$$\begin{aligned} & (1 \ a_2)(a_3 \ a_4) \dots (a_{n-1} \ a_n) & , \text{ for even } n \\ & (1)(a_2 \ a_3 \ a_4)(a_5 \ a_6) \dots (a_{n-1} \ a_n) & , \text{ for even } n \\ & (1)(a_2 \ a_3)(a_4 \ a_5) \dots (a_{n-1} \ a_n) & , \text{ for odd } n \end{aligned}$$

In an n -star, an antipode permutation P_a can be formed with digits a_2, a_3, \dots, a_n , chosen from the set $\{2, 3, \dots, n\}$, such that $\lfloor n/2 \rfloor$ 2-cycles $(a_i \ a_j)$ (i.e, in P_a digit a_i occupies the position of digit a_j and vice-versa) are formed. For even n , the antipode permutation may also contain a 3-cycle and $\lfloor (n-3)/2 \rfloor$ 2-cycles. In any case, the distance from an antipode P_a to the identity permutation is equal to the diameter of the n -star, i.e. $\lfloor 3(n-1)/2 \rfloor$.

Similarly, the diameter of the n -SCC graph can be calculated by finding its antipode permutations. We recall that in the n -SCC graph there are $(n-1)$ nodes (I_i, P_1) , where P_1 is the identity permutation $123 \dots n$ and I_i is a digit of the set $\{2, 3, 4, \dots, n\}$. We must therefore choose one of these $(n-1)$ nodes as the identity node for the n -SCC graph. Let such node be $(I_1, P_1) = (2, 123 \dots n)$. Now, we define an antipode (I_a, P_a) in the n -SCC graph as follows:

- P_a is a permutation chosen such that the node (I_a, P_a) is located $\lfloor 3(n-1)/2 \rfloor$ lateral links away from (I_1, P_1) , while keeping the number of local links in the path to the identity node to a maximum.
- I_a is a digit chosen so as to include a maximum number of additional local links in the path to (I_1, P_1) .

Considering the above requirements, we state the following Lemma:

Lemma 2: The following nodes are antipodes in the n -SCC graph:

For odd n : $(2, (1)(2 \ a+1)(3 \ a+2) \dots (a-1 \ n-1)(a \ n))$

For even n : $(2, (1 \ b+1)(2 \ b+2) \dots (b-1 \ n-1)(b \ n))$

where $a = (n+1)/2$ and $b = n/2$.

A proof of correctness for the above antipodes is given in Appendix A. From the above antipodes, we obtain the following Theorem:

Theorem 1 *The diameter of the n -SCC, $n \geq 2$, is :*

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

Proof: Routing from any of the above antipodes requires $d_{lat} = \lfloor 3(n-1)/2 \rfloor$ lateral links. The number of local links can be calculated as follows:

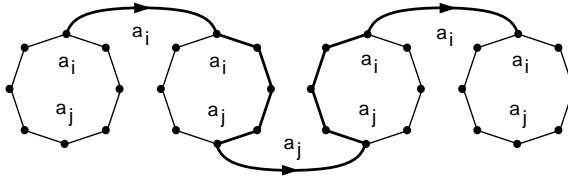


Figure 5: Execution of a cycle $(a_i a_j)$ in the n -SCC

- Permutation P_a has $\lfloor (n-1)/2 \rfloor$ cycles that does not include the digit 1. Execution of each of these cycles requires $2\lfloor (n-1)/2 \rfloor$ local links, as shown in Figure 5. Thus, the total number of local links required for execution of all cycles in P_a that do not include digit 1 is $d_{loc}(1) = 2(\lfloor (n-1)/2 \rfloor)^2$.
- Permutation P_a has $\lfloor n/2 \rfloor$ cycles of length 2 that must be executed in the route to the identity node. The cycles in P_a may be ordered such that only one local link is required to move between the execution of adjacent cycles. This adds $d_{loc}(2) = (\lfloor n/2 \rfloor - 1)$ local links to the routing from the antipode to the identity.
- Digit I_a in the antipode (I_a, P_a) is such that a maximum of $d_{loc}(3) = (\lfloor n/2 \rfloor - 1)$ local links may be added to the routing.
- The diameter of the n -SCC graph is the sum of d_{lat} , $d_{loc}(1)$, $d_{loc}(2)$ and $d_{loc}(3)$, so the result follows. \square

7 Broadcasting in the n -SCC Graph

A broadcasting algorithm for the n -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 broadcasted. If only one node has received it, then additional communication takes place to relay the broadcasted information to the uninformed node. If both or neither of the nodes have already received the information, then no additional messages are exchanged between the nodes. We assume that note 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 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.

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

8 One-port broadcasting algorithms for the n -SCC

One-port broadcasting in the n -cube can be accomplished with an optimal algorithm in n steps [1], [14]. A one-port broadcasting algorithm for the n -star requiring at most $3(n \log n - n/2)$ steps was introduced in [1], followed by an optimal algorithm requiring $\sum_{i=2}^n (\lceil \log i \rceil + 1)$ steps [15]. In either case, the complexity of one-port broadcasting algorithms for the n -star is $O(n \log n)$.

Broadcasting in an n -SCC graph is basically an extension of the broadcasting algorithms already introduced for the n -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 broadcasted 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 2 *One-port broadcasting in an n -SCC graph requires a sequence with $O(n^2)$ 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, we shall analyze different possible broadcasting sequences. The general approach for analyzing a broadcasting sequence for the n -SCC graph consists of two steps. First, we 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 n -star graph embedded in the n -SCC.

8.1 An algorithm based on an $n \log n$ switching network

The first broadcasting algorithm we are going to analyze is an extension of the $O(n \log n)$ algorithm introduced in [1] for the n -star graph. Such algorithm 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 6 (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 at most [1]:

$$\overline{\sigma_{star}(T_n)} = 3(n \log n - n/2)$$

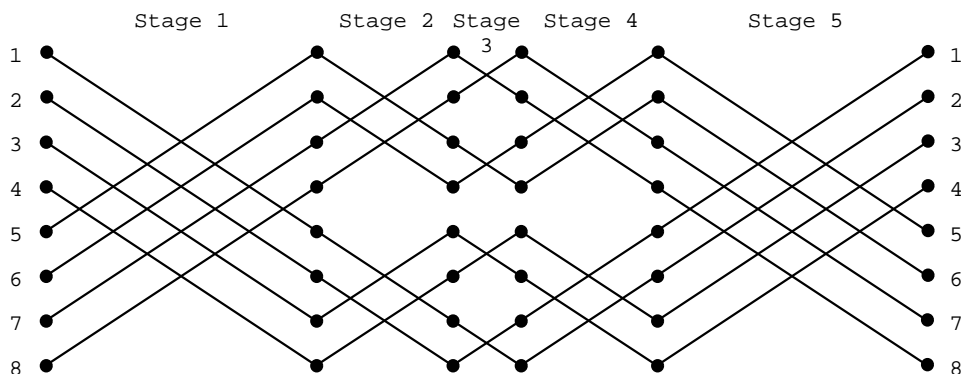


Figure 6: An $n \log n$ switching network (T_8)

A broadcast sequence for an n -SCC graph can be built from the $\sigma_{star}(T_n)$ broadcast sequence of the quotient n -star embedded in the n -SCC. To do so, the local links transmissions that are required to move between adjacent lateral link steps in $\sigma_{star}(T_n)$ must be taken into account. That yields the following theorem:

Theorem 3 *An $n \log n$ switching network T_n generates a one-port broadcasting sequence for the n -SCC graph with length at most:*

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

Proof: An analysis of Figure 6 shows that each stage of T_n has 1 switch that includes digit 1 and at most $\lfloor (n-2)/2 \rfloor$ switches that do not include digit 1. The switches in each stage of T_n can be ordered such that their corresponding star operations are just 1 digit apart from each other. Therefore, if we use T_n to build a broadcasting algorithm for an n -SCC, only 1 local link will be required to move between switches belonging to the same stage. Moving between adjacent stages requires from 0 to $\lfloor (n-1)/2 \rfloor$ local links. We will assume the worst case to evaluate the length of a broadcasting sequence for the n -SCC, originated from $\sigma_{star}(T_n)$. Therefore, the number of local links required to move between all switches in T_n is:

$$\overline{\sigma_{SCC}^1(T_n, loc)} \leq \left\lfloor \frac{n-2}{2} \right\rfloor (2 \log n - 1) + \left\lfloor \frac{n-1}{2} \right\rfloor (2 \log n - 2) = (n-2)(2 \log n - 1) - \left\lfloor \frac{n-1}{2} \right\rfloor$$

We must now consider the local links required to move between digits inside the same switch. The first and last stages of T_n have at most $\lfloor (n-2)/2 \rfloor$ switches involving 2 digits $a_i \neq 1$, $a_j \neq 1$ that are $\lfloor (n-1)/2 \rfloor$ local links apart from each other. Those switches require $2 \lfloor (n-1)/2 \rfloor$ local links each to be executed.

The second stage and the stage preceding the last one have at most $\lfloor (n-2)/2 \rfloor$ switches involving 2 digits $a_i \neq 1$, $a_j \neq 1$ that are $\lfloor n/4 \rfloor$ local links apart from each other. Those switches require $2 \lfloor n/4 \rfloor$ local links each to be executed.

Examining the remaining stages, we conclude that the total number of local links required to individually execute all switches in T_n is:

$$\overline{\sigma_{SCC}^2(T_n, loc)} \leq 2 \left(2 \left\lfloor \frac{n-2}{2} \right\rfloor \left\lfloor \frac{n-1}{2} \right\rfloor + 2 \left\lfloor \frac{n-2}{2} \right\rfloor \left\lfloor \frac{n}{4} \right\rfloor + \dots + 2 \left\lfloor \frac{n-2}{2} \right\rfloor \left\lfloor \frac{n}{n} \right\rfloor \right) - \left\lfloor \frac{n-2}{2} \right\rfloor \left\lfloor \frac{n}{n} \right\rfloor$$

$$< (4n - 1) \left\lfloor \frac{n-2}{2} \right\rfloor$$

Hence, the total number of local links required by the broadcasting sequence is:

$$\overline{\sigma_{SCC}(T_n, loc)} = \overline{\sigma_{SCC}^1(T_n, loc)} + \overline{\sigma_{SCC}^2(T_n, loc)} < 4n \left\lfloor \frac{n-2}{2} \right\rfloor + 2(n-2)(\log n - 1)$$

Taking now into account the lateral links, we obtain the total length of a broadcasting sequence $\sigma_{SCC}(T_n)$, built from a switching network T_n :

$$\overline{\sigma_{SCC}(T_n)} = \overline{\sigma_{star}(T_n)} + \overline{\sigma_{SCC}(T_n, loc)} < 4n \left\lfloor \frac{n}{2} \right\rfloor + (5n - 4)(\log n - 1) - \frac{5n}{2}$$

□

Although Figure 6 was used to illustrate the case $n = 8$, a proper broadcasting sequence may be obtained for generic n from a $T_{\hat{n}}$ switching network, where $\hat{n} = \lceil \log n \rceil$. For instance, a broadcasting sequence for a δ -SCC graph may be obtained from a T_8 switching network by listing only those pairwise interchanges of digits (a_i, a_j) in T_8 that do not include either digit 6 or 7 [1].

Table 1 lists upper bounds in the number of steps required by a broadcasting algorithm based on an $n \log n$ switching network, according to Equation 3.

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	18	20	38	375%
5	480	16	27	27	54	238%
6	3600	19	37	60	97	410%
7	30240	31	48	74	122	293%
8	282240	34	60	120	180	429%

Table 1: Maximum number of steps required by the broadcasting sequence $\sigma_{SCC}(T_n)$

The efficiency of broadcasting algorithms for interconnection networks can be measured by comparing the required number of steps with the diameter of the graph. We present such comparison in Table 1 for the broadcasting sequence $\sigma_{SCC}(T_n)$ in terms of percentages. More specifically, the *relative distance to the diameter* of a generic broadcasting algorithm requiring $\overline{\sigma_{SCC}}$ steps is defined for a SCC graph as:

$$\frac{\overline{\sigma_{SCC}} - d_{SCC}}{d_{SCC}} \times 100\%$$

For large n , $\overline{\sigma_{SCC}(T_n)}$ approximates to $2n^2$ and d_{SCC} approximates to $0.5n^2$. Therefore, the higher order terms in the expressions of $\overline{\sigma_{SCC}(T_n)}$ and d_{SCC} indicate that the number of steps required by the broadcasting sequence $\sigma_{SCC}(T_n)$ is approximately 300% higher than the diameter of the n -SCC graph. If we take into account the lower order terms, we notice that the relative distance to the diameter can be as high as 429% for $4 \leq n \leq 8$.

A broadcasting algorithm for the n -SCC graph based on an $n \log n$ switching network seems to be far from the minimum number of steps stated in Theorem 2. One possible approach to reduce the number of steps required by this broadcasting algorithm is to execute some of the steps of sequence $\sigma_{SCC}(T_n)$ in parallel. However, such approach is not feasible in this case, since the algorithm uses star operations (pairwise interchanges of digits) that must be executed sequentially rather than in parallel. More specifically, $\sigma_{SCC}(T_n)$ consists of a sequence of lateral link steps that must be necessarily intercalated with local link transmissions that will forward the broadcast message to the nodes that are supposed to perform the next lateral link transmission. This limitation dictates a sequential approach for the execution of $\sigma_{SCC}(T_n)$.

8.2 An algorithm based on an $n^2/2$ switching network

Referring again to Theorem 2, a simple analysis of the expression of the diameter of the n -SCC graph (d_{SCC}) reveals that ideally a one-port broadcasting algorithm should require $d_{star} = \lfloor 3(n-1)/2 \rfloor$ lateral link steps and $d_{SCC} - d_{star} = 2(\lfloor (n-1)/2 \rfloor)^2 + 2\lfloor n/2 \rfloor - 2$ local link steps. The dominant terms for the ideal number of lateral and local link steps can be approximated by $1.5n$ and $0.5n^2$, respectively.

Clearly, our first broadcasting algorithm is far from these goals, since it requires about $3n \log n$ lateral link steps and $2n^2$ local link steps. Also, by inspecting Table 1 we notice that the number of local link steps required by the algorithm based on an $n \log n$ switching network is a major contribution to its inefficiency. As an approach to verify whether a more efficient algorithm can be found, we might perform some additional analysis on the $n \log n$ switching network (T_n) in order to understand its drawbacks and check whether it is possible to use a different type of switching network, particularly one with the ability of generating a broadcasting algorithm containing about $0.5n^2$ local link steps.

Without loss of generality, a simple analysis shows that the $4n\lfloor n/2 \rfloor \approx 2n^2$ term in Equation 3 is due to:

- Each stage of the switching network has $n/2$ switches.
- While we approach the inner stage of the switching network, each switch requires about $2n/2, 2n/4, \dots, 2n/n$ local links to be executed.
- Therefore, the total number of local links required to execute all switches in T_n (not taking into account local links that are required to move between switches) is approximately:

$$\frac{n}{2} \left(\frac{2n}{2} + \frac{2n}{4} + \dots + \frac{2n}{n} + \dots + \frac{2n}{4} + \frac{2n}{2} \right) \approx 2n^2$$

This analysis reveals that a major cause for the presence of the $2n^2$ term in $\overline{\sigma_{SCC}(T_n)}$ is the existence of switches that require many local links to be executed, particularly in the initial and final stages of T_n (i.e., switches on these stages require $2n/2, 2n/4, \dots$ local links as we go deeper into T_n). Therefore, it seems worthwhile to examine other type of switching network, particularly one containing only switches requiring $2n/n = 2$ local links to be executed. Such switches allow pairwise interchanges of digits located at adjacent positions (p_i, p_j) of a permutation of n digits, such that $|p_i - p_j| \bmod (n-2) = 1$. These requirements result in an $n^2/2$ switching network (\mathcal{T}_n). Figure 7 shows \mathcal{T}_8 .

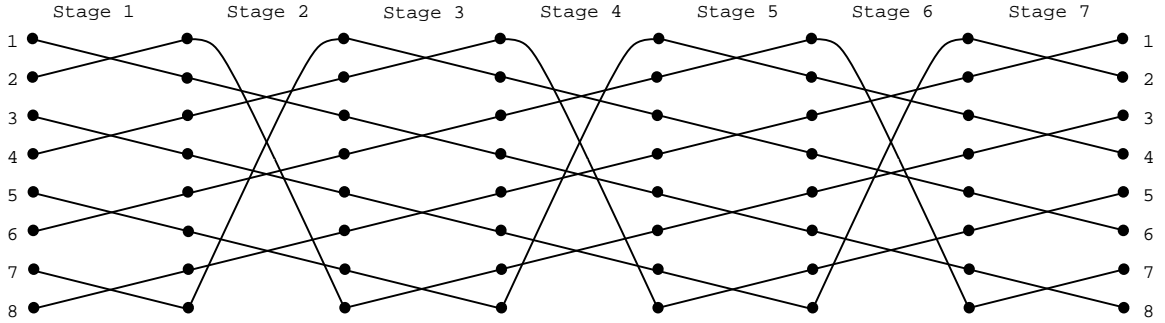


Figure 7: An $n^2/2$ switching network (\mathcal{T}_8)

To sort a permutation through \mathcal{T}_n , we compare at each stage k of \mathcal{T}_n every pair of digits (a_x, a_y) located at adjacent positions (p_i, p_j) and connected by a switch S_{ij} . Considering that at the beginning of stage k position p_i is located above p_j (i.e., $p_i < p_j$), we set switch S_{ij} (i.e., exchange a_x with a_y) if $a_x > a_y$. It is easily seen that:

Theorem 4 *An $n^2/2$ switching network \mathcal{T}_n has $(n - 1)$ stages and $n(n - 1)/2$ switches.*

We now show:

Theorem 5 *Considering even n , every permutation of n digits can be sorted through an appropriate setting of the switches in \mathcal{T}_n .*

Proof: A permutation of n digits has at most 2 digits that are $(n - 1)$ positions apart from their correct position. Since \mathcal{T}_n operates by exchanging the positions of a pair of adjacent digits, a maximum of $(n - 1)$ switch settings are required to place one of these digits in its correct position.

We are therefore left with a permutation containing at most 2 digits that are $(n - 2)$ positions apart from their correct position. An additional setting of $(n - 2)$ switches can be used to place one of these digits in its correct position.

By extending this reasoning, the maximum number of switch settings that may be required to sort the permutation is $(n - 1) + (n - 2) + \dots + 1 = n(n - 1)/2$, which is exactly the number of switches in \mathcal{T}_n . \square

A direct consequence of Theorem 5 is that a sequence containing all switches in \mathcal{T}_n can be used as a broadcasting algorithm for an n -star graph (and therefore, also for an n -SCC graph). Basically, a broadcasting algorithm for an n -star graph can be obtained by listing all switches in \mathcal{T}_n and transforming every switch in an star operation as described before. Using this methodology, we obtain the following theorems:

Theorem 6 *An $n^2/2$ switching network \mathcal{T}_n generates a one-port broadcasting sequence for an n -star graph with length:*

$$\overline{\sigma_{star}(\mathcal{T}_n)} = 1.5n^2 - 3.5n + 2 \quad (4)$$

Proof: Each stage of \mathcal{T}_n has a switch that can be represented by a star operation consisting of just one lateral link and $(n-2)/2$ switches that require 3 lateral links when transformed in star operations. The total number of lateral links is therefore:

$$\overline{\sigma_{star}(\mathcal{T}_n)} = (n-1) \left(1 + 3 \frac{n-2}{2} \right) = 1.5n^2 - 3.5n + 2$$

□

Theorem 7 *An $n^2/2$ switching network \mathcal{T}_n generates a one-port broadcasting sequence for an n -SCC graph with length:*

$$\overline{\sigma_{SCC}(\mathcal{T}_n)} = 3.5n^2 - 9.5n + 6$$

Proof: The switches of \mathcal{T}_n can be ordered such that at each stage only one local link is required to move from the first to the second switch. Moving to each switch remaining in the same stage requires two local links. Also, one local link is required to move to the first switch in the next stage. As an example, consider the first 3 stages of \mathcal{T}_8 , after their transformation to star operations:

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

Therefore, the number of local links required to move between switches in \mathcal{T}_n is:

$$\overline{\sigma_{SCC}^1(\mathcal{T}_n, loc)} = 2(n-1) + 2(n-1) \left(\frac{n}{2} - 2 \right) = n^2 - 3n + 2$$

Each stage of \mathcal{T}_n has $(n-2)/2$ switches that require 2 local links for their internal execution. The total number of local links required to internally execute all switches in \mathcal{T}_n is therefore:

$$\overline{\sigma_{SCC}^2(\mathcal{T}_n, loc)} = 2(n-1) (n-2)/2 = n^2 - 3n + 2$$

Hence, the total number of local links required by a broadcasting algorithm based on \mathcal{T}_n is:

$$\overline{\sigma_{SCC}(\mathcal{T}_n, loc)} = \overline{\sigma_{SCC}^1(\mathcal{T}_n, loc)} + \overline{\sigma_{SCC}^2(\mathcal{T}_n, loc)} = 2n^2 - 6n + 4$$

The total length of the broadcasting sequence for the n -SCC graph, when obtained from \mathcal{T}_n is therefore:

$$\overline{\sigma_{SCC}(\mathcal{T}_n)} = \overline{\sigma_{star}(\mathcal{T}_n)} + \overline{\sigma_{SCC}(\mathcal{T}_n, loc)} = 3.5n^2 - 9.5n + 6$$

□

Table 2 lists the number of steps required by a one-port broadcasting algorithm based on an $n^2/2$ switching network, according to Theorem 7.

Table 2 indicates that the algorithm based on an $n^2/2$ switching network is more efficient than the previous algorithm, for $4 \leq n \leq 8$. However, Theorem 7 shows that the length of the broadcasting sequence generated by \mathcal{T}_n approximates to $3.5n^2$ as n grows. This means that, for larger values of n , \mathcal{T}_n tends to generate a broadcasting algorithm requiring more steps than the broadcasting algorithm based on the $n \log n$ switching network (\mathcal{T}_n).

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	12	12	24	200%
5	480	16	22	24	46	188%
6	3600	19	35	40	75	295%
7	30240	31	51	60	111	258%
8	282240	34	70	84	154	353%

Table 2: Number of steps required by the broadcasting sequence $\sigma_{SCC}(\mathcal{T}_n)$

The additional stages introduced in \mathcal{T}_n can be blamed for this result, as they cause the number of switches to be quadratic and correspondingly increase the number of lateral links to approximately $3n^2/2 = 1.5n^2$. Furthermore, a one-port broadcasting algorithm for the n -star generated by \mathcal{T}_n is not optimal, since the length of such sequence has $1.5n^2$ as its dominant term (Equation 4) and the n -star graph allows for an $O(n \log n)$ one-port broadcasting algorithm.

The number of steps required by the broadcasting sequence $\sigma_{SCC}(\mathcal{T}_n)$ can not be reduced by introducing parallelism in its execution. The reasoning for this statement is the same previously used for the algorithm generated by \mathcal{T}_n .

It is also interesting to notice that \mathcal{T}_n does not even reduce significantly the requirements on local links when compared to \mathcal{T}_n . Although the number of local links required to internally execute all switches in \mathcal{T}_n is reduced to about n^2 , the number of local links required to move between switches is increased to about n^2 . The result is that the total number of local links in the algorithm generated by \mathcal{T}_n is approximately the same found in the algorithm generated by \mathcal{T}_n (i.e., about $2n^2$).

8.3 Algorithms based on cyclic sequences of digits

The sequential nature of broadcasting algorithms obtained from switching networks suggests that broadcasting sequences that allow parallel execution of steps should be investigated.

One such sequence was proposed in [1] for the n -star graph, and basically is formed by repeating a cyclic pattern of the digits $(2\ 3\ \dots\ n)$ as follows. Each digit in $\sigma_{star}(C)$ actually represents a lateral link. Notice that in $\sigma_{star}(C)$ the pattern $(2\ 3\ \dots\ n)$ is repeated d_{star} times, where d_{star} is the diameter of the quotient n -star embedded in the n -SCC graph. Therefore, $\sigma_{star}(C)$ 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, $\sigma_{star}(C)$ can be used as a broadcasting algorithm for the n -SCC graph.

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

The length of the above sequence is $\overline{\sigma_{star}(C)} = (n-1)d_{star} = (n-1)\lfloor 3(n-1)/2 \rfloor$. Therefore, a one-port broadcasting algorithm for an n -star graph based on $\sigma_{star}(C)$ has a complexity of $O(n^2)$.

Let us now examine how $\sigma_{star}(C)$ can be extended to accomplish one-port broadcasting in an n -SCC graph. A simple approach consists of inserting a local link step between every two adjacent lateral link steps in $\sigma_{star}(C)$, resulting in a sequential broadcasting sequence $\sigma_{SCC}^{seq}(C)$. Execution of such broadcasting sequence is illustrated in Figure 8 for a 5-SCC graph.

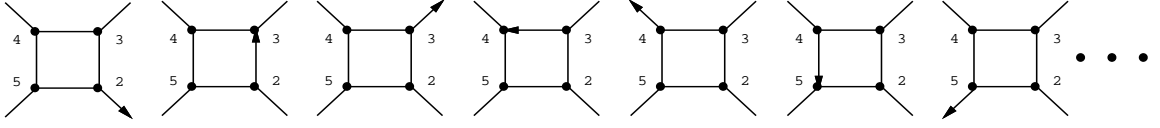


Figure 8: One-port broadcasting in a 5-SCC graph ($\sigma_{star}^{seq}(C)$)

It can be easily seen that $\sigma_{SCC}^{seq}(C)$ requires $\overline{\sigma_{star}(C)}$ lateral link steps and $\overline{\sigma_{SCC}^{seq}(C, loc)} = \overline{\sigma_{star}(C)} - 1$ local link steps. Therefore, the total number of steps required by $\sigma_{SCC}^{seq}(C)$ is $\overline{\sigma_{SCC}^{seq}(C)} = \overline{\sigma_{star}(C)} + \overline{\sigma_{SCC}^{seq}(C, loc)} = 2(n-1) \lfloor 3(n-1)/2 \rfloor - 1$.

Notice that due to its sequential nature, $\sigma_{SCC}^{seq}(C)$ requires $O(n^2)$ lateral link steps and $O(n^2)$ local link steps. However, $\sigma_{star}(C)$ can actually run in $O(n)$ lateral link steps by using parallel transmissions in those links. Moreover, while in an n -star such technique actually constitutes a multiple-port broadcasting algorithm, in an n -SCC we can benefit from parallel transmissions in the lateral links and still have a one-port broadcasting algorithm.

Before adding more details on the advantages of using parallel transmissions in the lateral links of an n -SCC graph, we state the following theorem:

Theorem 8 *The use of a δ -port communication model in an n -star graph with degree $\delta = n - 1$ and diameter $d_{star} = \lfloor 3(n - 1)/2 \rfloor$ 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 broadcasted. Suppose that at the beginning of the algorithm, only node N_i holds the information. After the first step of the algorithm, all nodes within a distance of one lateral link from N_i will also have received the information. After the second step, the information has been propagated to all nodes within two lateral links from N_i , and so on. After d_{star} steps, all nodes in the n -star graph have received the broadcasted 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

Also notice that δ -port communication results in an algorithm with $O(n)$ steps, while a one-port broadcasting algorithm in the n -star requires $O(n \log n)$ steps [1], [15]. A main disadvantage of using such δ -port algorithm in an n -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 one-port broadcasting algorithm in the n -SCC graph.

An efficient mapping of sequence $\sigma_{star}(C)$ onto the SCC graph can be obtained with a sequence $\sigma_{SCC}^{par}(C)$ that uses all lateral links of each supernode simultaneously. To illustrate this reasoning, Figure 9 shows the initial steps required to run $\sigma_{SCC}^{par}(C)$ in a 6-SCC graph. Clearly, $\sigma_{SCC}^{par}(C)$ can be executed faster than $\sigma_{SCC}^{seq}(C)$, since the information is initially broadcasted inside the supernode and then all lateral links are used simultaneously to pass the information on to adjacent supernodes.

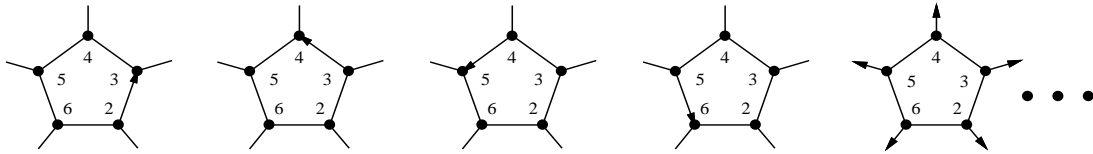


Figure 9: One-port broadcasting in a 6 -SCC ($\sigma_{SCC}^{par}(C)$)

The full broadcasting algorithm requires this operation to be repeated d_{star} times. Notice 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 $(n-1)$ lateral link steps of the previous sequence $\sigma_{SCC}^{seq}(C)$ 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 $(n-1)$.

Notice that in Figure 9 $(n-2)$ local link steps are required to accomplish the broadcasting inside each supernode, since the information flows in just one direction. Therefore, $\sigma_{SCC}^{par}(C)$ requires d_{star} lateral link steps and $(n-2)d_{star}$ local link steps, for a total of $(n-1)d_{star} = (n-1) \lfloor 3(n-1)/2 \rfloor$ steps.

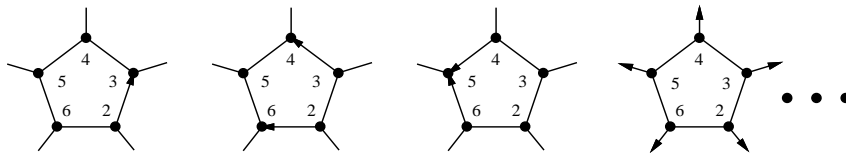


Figure 10: One-port broadcasting in a 6 -SCC ($\sigma_{SCC}(C)$)

A better approach consists of adopting the concept of parallel transmissions in the local links as well (Figure 10). This can be done in a one-port broadcasting algorithm by forcing one of the nodes in the ring 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. The result is that now only $\lfloor n/2 \rfloor$ local link steps are required to accomplish one-port broadcasting inside a supernode. The particular node that initiates the broadcasting in a ring is either a node that originated a piece of information to be broadcasted 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. Let us call this particular broadcasting sequence as $\sigma_{SCC}(C)$.

Theorem 9 *A one-port broadcasting algorithm for an n -SCC graph based on the cyclic sequence $\sigma_{SCC}(C)$ and using parallel transmissions over both lateral and local links requires a total of $\overline{\sigma_{SCC}(C)}$ steps, where:*

$$\overline{\sigma_{SCC}(C)} = \left\lfloor \frac{n+2}{2} \right\rfloor \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

Proof: The number of steps using lateral link transmissions in $\sigma_{SCC}(C)$ is:

$$\overline{\sigma_{SCC}(C, lat)} = d_{star} = \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

Also, the number of local link steps in $\sigma_{SCC}(C)$ is:

$$\overline{\sigma_{SCC}(C, loc)} = \left\lfloor \frac{n}{2} \right\rfloor d_{star} = \left\lfloor \frac{n}{2} \right\rfloor \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

The total number of steps required to run sequence $\sigma_{SCC}(C)$ is therefore:

$$\overline{\sigma_{SCC}(C)} = \overline{\sigma_{SCC}(C, lat)} + \overline{\sigma_{SCC}(C, loc)} = \left\lfloor \frac{n+2}{2} \right\rfloor \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

□

Table 3 lists the number of steps required by a one-port broadcasting algorithm using the cyclic sequence $\sigma_{SCC}(C)$, according to Theorem 9. The results shown in this table indicate that the algorithm based on $\sigma_{SCC}(C)$ clearly outperforms the previous algorithms, since the relative distance to the diameter of the n -SCC graph is at most 50% for $4 \leq n \leq 8$. Also, the algorithm based on sequence $\sigma_{SCC}(C)$ is optimal from the viewpoint of lateral link steps, since it requires exactly $d_{star} = \lfloor 3(n-1)/2 \rfloor$ such steps.

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 3: Number of steps required by the broadcasting sequence $\sigma_{SCC}(C)$

We now present a synchronous algorithm to accomplish one-port broadcasting in an n -SCC graph using sequence $\sigma_{SCC}(C)$. 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 4 (One-port broadcasting in the n -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 4

A straightforward modification of Algorithm 4 allows broadcasting of B consecutive messages in the SCC graph in a pipeline fashion. In this case, each node can keep the required control variables 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 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 INFORMED[], DONE_WITH_LATERALS[] and DONE_WITH_LOCALS[]. 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. Therefore, 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 per message.

8.4 Comparison of one-port broadcasting algorithms

Table 4 summarizes the characteristics of the one-port broadcasting algorithms presented so far. The broadcasting algorithm generated by cyclic sequences of digits and using parallelism in both lateral and local link steps clearly outperforms the remaining algorithms, yielding a sequence $\sigma_{SCC}(C)$ containing about $1.5n$ lateral link steps and $0.75n^2$ local link steps. Such 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 $\sigma_{SCC}(C)$ is just 12.5% to 50% greater than the diameter of the n -SCC graph.

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.

It is also interesting to notice that although the broadcasting algorithm generated by an $n \log n$ switching network runs efficiently in the star graph, it does not find such an efficient implementation in the SCC. On the other hand, the one-port implementation of the broadcasting sequence $\sigma_{star}(C)$ requires $O(n^2)$ steps in the star graph, but can be optimally implemented from the viewpoint of lateral link steps in the n -SCC graph.

As a final comment on one-port broadcasting algorithms for the n -SCC graph, we add that a modified version T'_n of the switching network T_n shown in Figure 6 could actually result in a sequence containing about n^2 local links and $3n \log n$ lateral links. T'_n can be obtained by using at each stage switches connecting digit positions that are $1 \dots n/4 \ n/2 \ n/4 \dots 1$ apart from each other. However,

<i>Broadcasting sequence</i>	<i>Dominant term for lateral link steps</i>	<i>Dominant term for local link steps</i>	<i>Relative distance to the diameter ($4 \leq n \leq 8$)</i>
$\sigma_{SCC}(C)$	$1.5n$	$0.75n^2$	12.5% to 50%
$\sigma_{SCC}^{par}(C)$	$1.5n$	$1.5n^2$	50% to 106%
$\sigma_{SCC}^{seq}(C)$	$1.5n^2$	$1.5n^2$	188% to 309%
$\sigma_{SCC}(T_n)$	$1.5n^2$	$2n^2$	188% to 353%
$\sigma_{SCC}(T_n)$	$3n \log n$	$2n^2$	238% to 429%

Table 4: Comparison of one-port broadcasting algorithms

T'_n can not be used to generate a broadcasting algorithm for the n -star graph (and consequently neither for the n -SCC), since it does not have the ability to sort a permutation of n digits.

Therefore, it seems that the algorithm based on sequence $\sigma_{SCC}(C)$ is not only optimal from the viewpoint of lateral link steps but is also very close to optimality regarding the number of local links steps. Other approaches seem to increase both the number of local and lateral link steps.

9 Multiple-port broadcasting in the n -SCC

Multiple-port broadcasting algorithms for the n -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 $\delta = 3$, so that 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 n -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.

9.1 Algorithms based on sequential broadcasting sequences

Use of a multiple-port communication model is supposed to reduce the number of steps required by a particular broadcasting sequence. To verify such possibility in the case of the SCC graph, we initially investigate broadcasting sequences that have a sequential execution behavior (e.g. $\sigma_{SCC}(T_n)$, $\sigma_{SCC}(T_n)$ and $\sigma_{SCC}^{seq}(C)$).

An algorithm based on cyclic sequences of digits

For the sake of simplicity, let us consider sequence $\sigma_{SCC}^{seq}(C)$ first. If we recall that $\sigma_{SCC}^{seq}(C)$ interleaves local and lateral link steps, a possible solution to execute $\sigma_{SCC}^{seq}(C)$ under a multiple-port communication model seems to join a local and a lateral link transmission into a single step. The local link forwards the broadcasted information to a node that is connected to the next lateral link to be used in the execution of $\sigma_{SCC}^{seq}(C)$. Such concept is depicted in Figure 11 for a supernode belonging to a 5-SCC.

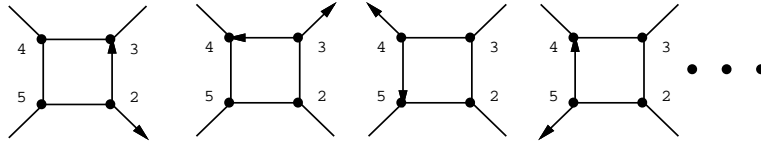


Figure 11: Multiple-port broadcasting in a 5-SCC graph ($\sigma_{SCC}^{seq}(C)$)

The simultaneous use of a lateral and a local link to execute $\sigma_{SCC}^{seq}(C)$ seems to reduce the total number of steps to about half that required by a one-port broadcasting algorithm. However, a simple inspection of the communication model shown in Figure 11 reveals that $\sigma_{SCC}^{seq}(C)$ can not be properly executed by such model. Suppose that a partially informed supernode a is passing on the broadcasted information to an uninformed supernode b through lateral link I (Figure 12). The data forwarding from a to b is actually performed by nodes I_a and I_b , which simultaneously also communicate with nodes J_a and J_b . Although I_a succeeds to broadcast the information to both J_a and I_b , I_b was still uninformed while comparing notes with J_b . Therefore, after completion of this step of the algorithm J_b will remain uninformed. As in the next step of the execution of $\sigma_{SCC}^{seq}(C)$ J_b is supposed to pass the information on to K_b (the next node in b) and J_c (a peer node in a third supernode c), the execution of $\sigma_{SCC}^{seq}(C)$ under the communication model shown in Figure 11 clearly fails. In other words, $\sigma_{SCC}^{seq}(C)$ seems to require interleaved execution of lateral and local link steps to assure proper forwarding of the broadcasted information inside the supernodes. Nevertheless, the use of interleaved execution of lateral and local link steps actually constitutes a one-port broadcasting algorithm.

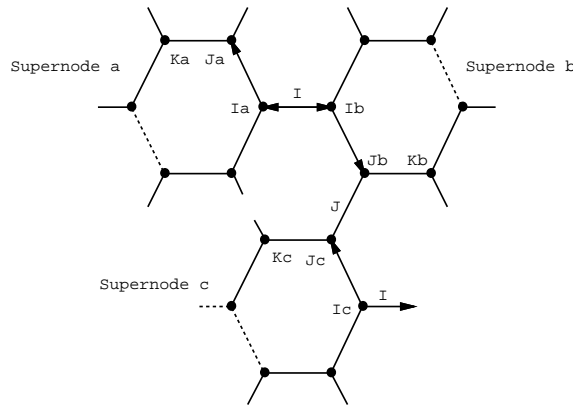


Figure 12: A detail of multiple-port broadcasting in an n -SCC graph ($\sigma_{SCC}^{seq}(C)$)

Another approach to run $\sigma_{SCC}^{seq}(C)$ consists of interleaving simultaneous transmissions on both local links with each lateral link step. Therefore, the information inside each supernode is broadcasted in both clockwise and counterclockwise directions. The main benefit from this approach seems to be a possible reduction of the number of local link steps, since a node in charge of performing a lateral link step may possibly have received the broadcasted information by either of its local link ports.

Although such approach correctly executes $\sigma_{SCC}^{seq}(C)$, it fails to reduce the number of local link steps. After each lateral link step, supernodes located at increasing distances from the source of the broadcasted information will be reached. These uninformed supernodes will require interleaved execution of the next lateral and local link steps to forward the information among each of their internal nodes and still execute $\sigma_{SCC}^{seq}(C)$ correctly. As simultaneous use of both local links results in a faster broadcasting inside the supernodes, at the time a supernode is completely informed there are still some lateral link steps remaining to be executed. Although these lateral link steps can easily skip any local link steps between them, there are farther supernodes in the graph still requiring interleaved execution of lateral and local link steps. This actually imposes a limit on the total number of steps required to execute $\sigma_{SCC}^{seq}(C)$.

Our conclusion is that a multiple-port version of the broadcasting sequence $\sigma_{SCC}^{seq}(C)$ does nothing more than increasing the number of node comparisons between the nodes of an n -SCC graph, without actually reducing the total number of steps when compared with the one-port version.

Algorithms based on switching networks

An analysis of one-port broadcasting sequences derived from the switching networks shown in the previous section (namely, $\sigma_{SCC}(T_n)$ and $\sigma_{SCC}(\mathcal{T}_n)$) reveals that:

- Both $\sigma_{SCC}(T_n)$ and $\sigma_{SCC}(\mathcal{T}_n)$ require that each supernode use exactly one lateral link during each lateral link step (i.e., there is no parallelism in the lateral links).
- A proper selection of local link steps must be inserted between the execution of any two consecutive lateral link steps.

If we now try to derive a multiple-port version for the one-port broadcasting algorithms based on switching networks, we notice that the same restriction previously pointed for the cyclic sequence $\sigma_{SCC}^{seq}(C)$ holds for $\sigma_{SCC}(T_n)$ and $\sigma_{SCC}(\mathcal{T}_n)$. In other words, $\sigma_{SCC}(T_n)$ and $\sigma_{SCC}(\mathcal{T}_n)$ must be executed sequentially and can not have their number of steps reduced by utilization of a multiple-port communication model.

9.2 An algorithm based on parallel broadcasting sequences

Let us now turn our attention to broadcasting sequences that allow parallel execution of steps, such as $\sigma_{SCC}^{par}(C)$ and $\sigma_{SCC}(C)$. An efficient multiple-port version of sequence $\sigma_{SCC}(C)$ (namely, $\sigma_{SCC}^m(C)$) can be achieved by using both local links simultaneously while broadcasting the information inside each supernode. Figure 13 shows such technique for one of the supernodes of a 6-SCC.

Theorem 10 *A multiple-port broadcasting algorithm for an n -SCC graph based on the cyclic sequence $\sigma_{SCC}^m(C)$ and using parallel transmission over both lateral and local links requires a total of $\overline{\sigma_{SCC}^m(C)}$ steps, where:*

$$\overline{\sigma_{SCC}^m(C)} = \left\lfloor \frac{n+1}{2} \right\rfloor \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

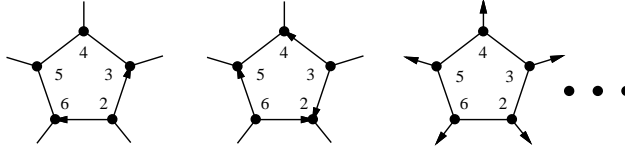


Figure 13: Multiple-port broadcasting in a 6-SCC ($\sigma_{SCC}^m(C)$)

Proof: Our approach for multiple-port communication does not affect the number of steps using lateral link transmissions compared to the one-port broadcasting sequence $\sigma_{SCC}(C)$. Hence:

$$\overline{\sigma_{SCC}^m(C, lat)} = d_{star} = \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

On the other hand, simultaneous use of both local links reduces the number of local link steps to:

$$\overline{\sigma_{SCC}^m(C, loc)} = \left\lfloor \frac{n-1}{2} \right\rfloor d_{star} = \left\lfloor \frac{n-1}{2} \right\rfloor \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

The total number of steps required to run sequence $\sigma_{SCC}^m(C)$ using the multiple-port communication model depicted in Figure 13 is then:

$$\overline{\sigma_{SCC}^m(C)} = \overline{\sigma_{SCC}^m(C, lat)} + \overline{\sigma_{SCC}^m(C, loc)} = \left\lfloor \frac{n+1}{2} \right\rfloor \left\lfloor \frac{3(n-1)}{2} \right\rfloor$$

□

Table 5 lists the number of steps required by a multiple-port broadcasting algorithm using the cyclic sequence $\sigma_{SCC}^m(C)$, according to Theorem 10. Notice that the total number of steps required by the algorithm based on sequence $\sigma_{SCC}^m(C)$ 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 $\sigma_{SCC}^m(C)$ is optimal from the viewpoint of lateral link steps, and by inspecting Table 5 we notice 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 5: Number of steps required by the broadcasting sequence $\sigma_{SCC}^m(C)$

A comparison of Tables 3 and 5 shows that for odd n , the one-port broadcasting sequence $\sigma_{SCC}(C)$ performs as well as its multiple-port counterpart ($\sigma_{SCC}^m(C)$). 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 n -SCC graph using sequence $\sigma_{SCC}^m(C)$ follows. The variables and functions used by the multiple-port algorithm have the same functionality previously defined for its one-port counterpart. 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, notice 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 5 (Multiple-port broadcasting in the n -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;
    end;
  end;

```

Message Pipelining Algorithm 5

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.

10 $O(n)$ Broadcasting Algorithms for the n -SCC

A broadcasting algorithm using either sequence $\sigma_{SCC}(C)$ or sequence $\sigma_{SCC}^m(C)$ requires $O(n)$ 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 over lateral and local link steps. This results in the following theorem:

Theorem 11 *Broadcasting in an n -SCC graph using either sequence $\sigma_{SCC}(C)$ or sequence $\sigma_{SCC}^m(C)$ can be accomplished in 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 for one-port broadcasting:

$$\frac{\overline{\sigma_{SCC}(C, lat)}}{TR(lat)} = \frac{\overline{\sigma_{SCC}(C, loc)}}{TR(loc)}$$

For one-port broadcasting, the ratio between $TR(loc)$ and $TR(lat)$ is then:

$$\frac{TR(loc)}{TR(lat)} = \left\lfloor \frac{n}{2} \right\rfloor$$

A similar reasoning for multiple-port broadcasting results in:

$$\frac{TR(loc)}{TR(lat)} = \left\lfloor \frac{n-1}{2} \right\rfloor$$

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

11 Comparison of Broadcasting Algorithms for the n -SCC and the n -star Graphs

A comparison of broadcasting algorithms for the n -SCC and the n -star graph is presented in Table 6. Table 6 lists both one-port and multiple-port algorithms, assuming the broadcasting sequences $\sigma_{SCC}(C)$ and $\sigma_{SCC}^m(C)$ in the case of the n -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 11.

One-port broadcasting in the n -star assumes that the optimal $O(n \log n)$ algorithm proposed in [15] is used. For multiple-port broadcasting in the n -star, we assume that the δ -port communication model proposed in Theorem 8 is used.

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>	4	24	4	8	-	8	8	100%
	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%
<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>	4	24	4	4	-	4	4	0%
	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%

Table 6: Comparison of broadcasting algorithms for the n -SCC and the n -star graphs

One-port broadcasting is accomplished more efficiently in the n -SCC graph than in the n -star. Notice that for $4 \leq n \leq 8$ the relative distance to the diameter of one-port broadcasting algorithms range from 12.5% to 50% in the case of the n -SCC graph, while for the n -star this range is 100% to 140%. Most interestingly, notice that one-port broadcasting in the n -SCC graph can be accomplished in $O(n)$ running time, while the n -star requires an $O(n \log n)$ running time. Such difference in performance can also be verified quantitatively in Table 6 by observing that one-port broadcasting 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. For $4 \leq n \leq 8$, the relative distance to the diameter of multiple-port broadcasting algorithms range from 0% to 17.6% in the case of the n -SCC graph, while for the n -star this relative distance can be made equal to 0% by using a δ -port broadcasting algorithm. Also, if we assume an n -star and an n -SCC graph with the same lateral link transmission rate, the running time of multiple-port broadcasting in an n -SCC is twice as much greater than that of an n -star containing $(n - 1)$ times fewer nodes. However, if the criterion (running time)/(graph size) is used to compare the efficiency of multiple-port broadcasting algorithms, then the n -SCC graph can also be considered superior to the n -star graph in regards to this aspect, for $n \geq 3$.

Another 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 6 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

12 Comparison to other Graphs

A comparison between different interconnection network graphs is shown in Table 7. The n -SCC graph offers a fixed-degree structure with bounded I/O requirements at the node level, which allows the construction of interconnection networks of different sizes with higher scalability in network growth than the n -cube and the n -star. In other words, processors with just 3 communication links can be used to build any n -SCC graph. Variable-degree graphs such as the n -cube and the n -star require a growing number of communication links at each processor as we increase the number of nodes in the graph. The result is increased complexity and higher pin count at each processor than required by fixed-degree graphs.

Graph	n	Topological properties			One-port broadcasting			
		Size	Degree	Diameter	Lateral link steps	Local link steps	Total number of steps	Running time in lateral link steps
n -cube	7	128	7	7	7	-	7	7
	8	256	8	8	8	-	8	8
	9	512	9	9	9	-	9	9
n -star	5	120	4	6	12	-	12	12
	6	720	5	7	16	-	16	16
	7	5040	6	9	20	-	20	20
n -CCC	4	64	3	8	4	8	12	18
	5	160	3	10	5	15	20	10
	6	384	3	13	6	18	24	12
	7	896	3	15	7	28	35	14
	8	2048	3	18	8	32	40	16
n -SCC	4	72	3	8	4	8	12	8
	5	480	3	16	6	12	18	12
	6	3600	3	19	7	21	28	14
	7	30240	3	31	9	27	36	18

Table 7: Comparison of interconnection network graphs

One of the trade-offs of fixed-degree graphs is an increased diameter. However, the n -SCC can be built with very high speed buses in the local links. Therefore, the n -SCC can present communication delays comparable to the n -star, if we consider that the lateral links often use serial links for making their lay-out simpler. In addition, many practical algorithms present locality of operation and require just a limited region of the interconnection network to run. This reduces even more the requirements for long communication paths in the graph and contributes to high performance in parallel computers.

Another disadvantage of fixed-degree graphs is a reduced fault tolerance in comparison to variable-degree graphs. The fault tolerance of the n -SCC graph is 2, while the fault tolerance of the n -star and the n -cube is respectively equal to $(n - 2)$ and $(n - 1)$. However, since the underlying topology

connecting the cycles of the n -SCC is the n -star, we are at least left with a richness of disjoint paths that might be taken in case of node failures [3], [12]. Of course, choosing between disjoint paths in an n -SCC graph containing faulty nodes requires a dynamic fault-tolerant routing algorithm. Such algorithm is actually an extension of the routing algorithm presented in this report and can be based on dynamic fault-tolerant routing and broadcasting algorithms already developed for the n -star [3], [16], [17] and the cube-connected cycles [18].

Table 7 also shows another type of I/O-bounded interconnection network, namely the cube-connected cycles or CCC. An n -CCC graph can be built by replacing each node of an n -cube with a ring of n or more nodes. Table 7 shows typical values for n -CCC graphs containing n nodes in each ring. The number of nodes and diameter of an n -CCC graph formed under such structure are given respectively by $N = n2^n$ and $d_{CCC} = 2n + \lfloor n/2 \rfloor - 2$ [19].

Compared to a CCC graph of similar size, the n -SCC graph presents about the same diameter for the cases where n is even. The diameter of the n -SCC graph shows a sharp discontinuity when n changes from an even to an odd value. Such behavior is due to the presence of the quadratic component in the n -SCC diameter expression.

The diameter of the CCC compares favorably with the n -SCC graph for odd n . However, the underlying topology or quotient Cayley graph used to connect the cycles in the n -SCC (i.e., the n -star) has several advantages over that used in the CCC (i.e., the n -cube) [1]. Among these advantages, we may cite a smaller degree from the viewpoint of the supernodes, as well as a shorter average distance and fault diameter. Also, an n -SCC graph requires fewer lateral links and fewer nodes at each ring than a CCC graph with similar number of nodes. Such characteristic reduces the complexity of the supernodes and makes their implementation simpler.

Other aspects such as average distance and fault diameter have not been formally derived for the n -SCC. However, it has been shown that the n -star outperforms the n -cube on these aspects [1], [3]. If we recall that the n -SCC not only uses the n -star as its quotient Cayley graph but also has fewer nodes in each ring, then it seems that we should expect favorable results when compared with the CCC.

Table 7 also compares the n -SCC with other graphs from the viewpoint of one-port broadcasting algorithms. Theorem 11 shows that the selection of a proper value for the ratio of the transmission rates in the local and lateral links of an n -SCC graph can reduce the running time of a broadcasting algorithm to about twice the time spent in the lateral link steps. This results in an $O(n)$ running time, while a one-port broadcasting in an n -star has a $O(n \log n)$ running time.

The values shown for one-port broadcasting in the n -star in Table 7 are optimal and have been extracted from [15]. By inspecting Table 7, we notice that the one-port broadcasting in an n -SCC graph can be accomplished with running time better than or equal to that of an n -star containing $(n - 1)$ times fewer nodes.

For $n = 4$ and $n = 6$, the total number of steps required by the one-port broadcasting algorithm is about 50% greater than the diameter of the corresponding SCC graph. For $n = 5$ and $n = 7$, the total number of steps is less than 16.1% greater than the diameter. Therefore, the higher discontinuity observed in the diameter of the n -SCC graph for odd n is somehow compensated by the increased efficiency of one-port broadcasting algorithms, what may be beneficial to different parallel processing applications.

Solutions to the broadcasting problem in the cube-connected cycles network have been presented in [20]. The approach used in that reference consists of using three arc-disjoint spanning trees (ADST's) with depth at most $4n$. An algorithm based on such ADST's could have been used to picture the time complexity of broadcasting in the CCC graph. However, the number of steps shown in Table 7 for the CCC graph refer to a broadcasting algorithm based on the same technique that we have introduced for the n -SCC. By doing so, we show that the CCC graph can also benefit from using parallel transmissions both on the lateral and the local links. We also use for the CCC graph the assumption of a higher transmission rate in the local links. Hence, a one-port broadcasting algorithm in an n -CCC exploiting parallel transmissions on the lateral and local links requires $\sigma_{CCC}(lat)$ lateral links steps and $\sigma_{CCC}(loc)$ local links steps, where:

$$\sigma_{CCC}(lat) = d_{cube} = n$$

$$\sigma_{CCC}(loc) = \left\lfloor \frac{n+1}{2} \right\rfloor d_{cube} = n \left\lfloor \frac{n+1}{2} \right\rfloor$$

The total number of steps required to perform one-port broadcasting in the n -CCC under this approach is then:

$$d(\sigma_{CCC}) = \sigma_{CCC}(lat) + \sigma_{CCC}(loc) = n \left\lfloor \frac{n+3}{2} \right\rfloor$$

Multiple-port broadcasting in an n -CCC using two local links at a time at each node and parallel transmission over all lateral links of a supernode results in:

$$\sigma_{CCC}(lat)_m = n$$

$$\sigma_{CCC}(loc)_m = n \left\lfloor \frac{n}{2} \right\rfloor$$

$$d(\sigma_{CCC})_m = n \left\lfloor \frac{n+2}{2} \right\rfloor$$

As in the n -SCC, we can choose a proper ratio for the transmission rates in the local and lateral links of the n -CCC in order to achieve an $O(n)$ running time for the broadcasting algorithm. Table 7 assumes that an equal amount of time is spent in the lateral and local link steps while running a one-port broadcasting algorithm in both the n -SCC and the n -CCC. Such condition can be obtained in the n -CCC by making $TR(loc)/TR(lat) = \lfloor (n+1)/2 \rfloor$. For a multiple-port broadcasting algorithm in the n -CCC, the same condition can be achieved by making $TR(loc)/TR(lat) = \lfloor n/2 \rfloor$.

Although one-port broadcasting can be accomplished within $O(n)$ running time in both the n -SCC and the n -CCC graph, the n -SCC shows better performance than an n -CCC graph with similar size. This is because in both cases the running time can be made proportional to twice the diameter of the underlying quotient graph (i.e., the n -cube in the case of the n -CCC and the n -star in the case of the n -SCC). Since the diameter of an n -star is less than that of a hypercube of similar size, we simply extend this result to conclude that broadcasting in the n -SCC graph can be achieved in shorter running time than in a CCC of similar size. As a matter of fact, an inspection of Table 7 allows us to confirm that.

13 Conclusion

We have described in this technical report a new interconnection network - the star-connected cycles or SCC. The SCC is an I/O-bounded graph and has been proposed as an evolution of the previous cube-connected cycles or CCC.

Aspects such as labeling of nodes, degree, diameter, symmetry, fault tolerance and Cayley graph representation have been presented. We have also developed an optimal routing algorithm for the n -SCC.

We have shown that an efficient $O(n \log n)$ one-port broadcasting algorithm devised for the n -star graph can not be efficiently mapped onto an n -SCC graph. We have proved that a major limitation of both this algorithm and a second algorithm extracted from an $n^2/2$ switching network is their sequential nature of execution.

Interestingly, we have shown that an $O(n^2)$ one-broadcasting cyclic 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, we have shown that 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 optimal 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 requires a running time better than or equal to that of an n -star containing $(n - 1)$ times fewer nodes.

In the case of multiple-port broadcasting algorithms, the running time required by an n -SCC is twice as much greater than that of an n -star containing $(n - 1)$ times fewer nodes. However, we may also claim that the n -SCC is superior to the n -star graph in regards to multiple-port broadcasting if criteria such as the communication overhead at each node and the ratio (running time)/(graph size) are used.

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

We have compared the n -SCC with variable-degree graphs such as the star graph and the n -cube. We claim that the n -SCC is an attractive alternative for parallel systems, since it overcomes some disadvantages of variable-degree graphs such as the number of communication ports required for massively parallel systems and the issue of scalability.

We have also compared the n -SCC with another fixed-degree graph, namely the CCC graph. We have shown that the diameter of the n -SCC is close to that of a CCC graph containing a similar number of nodes whenever n is even. However, even for the cases where n is odd, the n -SCC shows some superiority over the CCC, due to the use of the n -star as its quotient graph.

Some disadvantages of I/O-bounded interconnection networks are an increased diameter and a reduced fault tolerance. Nevertheless, the n -SCC shows some symmetry properties that allow the selection of different communication techniques between its nodes. A proper selection of transmission rates can be done to reduce the communication delay to levels comparable to those obtained in an n -star. Particularly for one-port broadcasting, we have shown that the total running time of the

algorithm in the n -SCC graph is better than or equal to that required by an n -star containing $(n - 1)$ times fewer nodes.

Finally, the disadvantage of a reduced fault tolerance is somehow alleviated in the n -SCC due to the richness of disjoint paths provided by its quotient graph (the n -star).

References

- [1] 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.
- [2] M. C. Pease, "The Indirect Binary n -Cube Microprocessor Array," *IEEE Transactions on Computers*, Vol. C-26, No. 5, May 1977, pp. 458-473.
- [3] S. B. Akers and B. Krishnamurthy, "The Fault Tolerance of Star Graphs," *2nd Int'l. Conf. on Supercomputing*, 1987, pp. 270-276.
- [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. Parallel Processing*, 1993.
- [5] 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.
- [6] 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.
- [7] S. B. Akers and B. Krishnamurthy, "Group Graphs as Interconnection Networks," *Proc. 14th Int'l Conf. on Fault-Tolerant Computing*, 1984, pp. 422-427.
- [8] R. J. Wilson, *Introduction to Graph Theory*, Longman, 3rd Edition, 1985, pp. 28-29.
- [9] N. Biggs, *Algebraic Graph Theory*, Cambridge University Press, 1974, pp. 101-118.
- [10] W. Ledermann, *Introduction to the Theory of Finite Groups*, Oliver and Boyd, London, 1964, pp. 34-64.
- [11] I. N. Herstein, *Topics in Algebra*, Xerox College Publishing, 2nd Edition, 1974, pp. 41-42.
- [12] S. G. Akl, K. Qiu and I. Stojmenovic, "Data Communication and Computational Geometry on the Star and Pancake Interconnection Networks," *Third IEEE Symposium on Parallel and Distributed Processing*, December 1991, pp. 415-422.
- [13] S. Latifi, "Parallel Dimension Permutations on Star Graph," *1993 Working Conf. on Architectures and Compilation Techniques for Fine and Medium Grain Parallelism*, January 1993.
- [14] S. L. Johnsson and C. T. Ho, "Optimum Broadcasting and Personalized Communications in Hypercubes," *IEEE Transactions on Computers*, Vol. 38, No. 9, September 1989, pp. 1249-1268.
- [15] 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.
- [16] S. Sur and P.K. Srimani, "A Fault-Tolerant Routing Algorithm in Star Graph Interconnection Networks," *Proc. Int'l. Conf. Parallel Processing*, 1991, Vol. 3, pp. 267-270.

- [17] N. Bagherzadeh, N. Nassif and S. Latifi, "A Routing and Broadcasting Scheme on Faulty Star Graphs," *IEEE Transactions on Computers* (to appear).
- [18] J.E. Jang, "Optimal Fault-Tolerant Broadcast Algorithm in a Cube-Connected Cycles Network," *Proc. Int'l Conf. on Database, Parallel Architectures and Their Applications (PARBASE-90)*, March 1990, pp. 206-215.
- [19] 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.
- [20] P. Fraigniaud and C.T. Ho, "Arc-Disjoint Spanning Trees on Cube-Connected Cycles Networks," *Proc. Int'l. Conf. Parallel Processing*, 1991, Vol. 1, pp. 225-229.

A Proof of Correctness for the n -SCC Antipodes

A common point in the antipodes proposed for the n -SCC graph is that they have a maximum number of 2-cycles. Such characteristic leads to a maximum number of lateral links in the path from an antipode to the identity. However, it is not obvious that the inclusion of a 3-cycle or a 4-cycle, for instance, does not result in a larger number of links (i.e., it may reduce the number of lateral links but increase the number of local links).

The 2-cycles included in the proposed antipodes (I_a, P_a) are chosen so as to represent nodes that are located at opposite sites in the supernodes of the n -SCC. This requires $2 \lfloor (n-1)/2 \rfloor$ local links and 3 lateral links for the execution of each 2-cycle.

Also, only one additional local link is required to move from a 2-cycle to the next. This condition always holds for the proposed antipodes, as long as a proper order of execution is chosen. For instance, an antipode $P_a = (1\ 5)(2\ 6)(3\ 7)(4\ 8)$ for an 8-SCC can be executed with the following order of lateral links: (5, 6, 2, 6, 7, 3, 7, 8, 4, 8).

By inspecting the structure of the n -SCC graph, it can be seen that the execution of a r -cycle that does not include digit 1 requires c_{lat} lateral links and at most c_{loc} local links, where:

$$c_{lat} = r + 1$$

and

$$c_{loc} = \begin{cases} (r-1) \lfloor \frac{n-1}{2} \rfloor & , \text{ for odd } r \ \& \ n \\ (r-1) \lfloor \frac{n-1}{2} \rfloor + \lfloor \frac{r-1}{2} \rfloor & , \text{ for odd } r \ \& \ \text{even } n \\ r \lfloor \frac{n-1}{2} \rfloor - \lfloor \frac{r-1}{2} \rfloor & , \text{ for even } r \ \& \ n \\ r \lfloor \frac{n-1}{2} \rfloor - 2 \lfloor \frac{r-1}{2} \rfloor & , \text{ for even } r \ \& \ \text{odd } n \end{cases}$$

The above equations hold for cycles of length at least two that do not include digit 1. The equations for c_{loc} are derived from r -cycles chosen so as to result in a maximum number of local links during its execution.

For a r -cycle including digit 1, the following equations hold:

$$c_{lat} = r - 1$$

and

$$c_{loc} = \begin{cases} (r-2) \lfloor \frac{n-1}{2} \rfloor & , \text{ if } n \text{ is even} \\ (r-2) \lfloor \frac{n-1}{2} \rfloor - \lfloor \frac{r-2}{2} \rfloor & , \text{ if } n \text{ is odd} \end{cases}$$

For the special case of a 2-cycle not including digit 1, we have $c_{lat} = 3$ and $c_{loc} = 2 \lfloor (n-1)/2 \rfloor$.

We now consider the effect of the inclusion of 3-cycles in (I_a, P_a) . To do so, we may substitute three 2-cycles of the proposed antipodes with two 3-cycles and check whether such modification increases the number of links in the path to the identity.

An antipode $P_a = (1\ 6)(2\ 7)(3\ 8)(4\ 9)(5\ 10)$ of a 10-SCC, for example, might be substituted with permutation $(1\ 6)(7\ 2\ 3)(4\ 9\ 8)(5\ 10)$. The 3-cycles in this example have been chosen so as to require a maximum number of local links to be executed.

The inclusion of two 3-cycles in (I_a, P_a) reduces the total number of cycles in the antipodes by 1. Therefore, we have one lateral link less in the path to the identity. Let this variation in the number of lateral links be $\Delta_a = -1$. We must now check the corresponding effect over the number of local links.

For even n and $r = 3$, each 3-cycle requires at most $2 \lfloor (n-1)/2 \rfloor + 1$ local links to be executed. We define the *non-concatenated execution* of a set of cycles S as the total sequence of local links required for the isolated execution of each cycle in S (i.e., additional cycles required to move between cycles in S are not taken into account).

Accordingly, we define the *concatenated execution* of a set of cycles S as the sequence of additional local links required to move between adjacent cycles in S . If S does not include all cycles in (I_a, P_a) , then the concatenated execution of S must also include additional local links that may be needed to enter and exit the execution of S .

The non-concatenated execution of two 3-cycles in (I_a, P_a) requires at most $4 \lfloor (n-1)/2 \rfloor + 2$ local links. If the digits in these two 3-cycles were organized as three 2-cycles of the proposed antipodes (I_a, P_a) , a total of $6 \lfloor (n-1)/2 \rfloor$ local links would be required.

Therefore, the substitution of three 2-cycles in (I_a, P_a) with two 3-cycles reduces the number of local links required for the non-concatenated execution of these cycles by $\Delta_b = -2 \lfloor (n-1)/2 \rfloor + 2$, if n is even.

We must now take into account the number of local links required to move from one cycle to another, i.e. additional local links are required for the concatenated execution of cycles in P_a .

The concatenated execution of three 2-cycles of the proposed antipodes (I_a, P_a) requires 4 additional local links. If we substitute these cycles with two 3-cycles, $\lfloor (n-1)/2 \rfloor + 2$ additional local links are required at most for the concatenated execution of the 3-cycles.

To justify this, we recall that the digits in the 3-cycles are selected from the digits originally belonging to three 2-cycles of the proposed antipodes (I_a, P_a) . We do so in order to study separately the effect of including 3-cycles in the antipodes, and therefore must leave the remaining cycles in (I_a, P_a) unchanged.

Considering that the 3-cycles must be chosen so as to maximize the number of local links required for their non-concatenated execution, we must have in the 3-cycles digits that represent nodes located in opposite sites in the supernodes. In other words, each of the two 3-cycles formed from three 2-cycles originally belonging to (I_a, P_a) must include the digits of one of these 2-cycles plus another digit from a third 2-cycle. For clarity, we refer the reader to the example already presented for the 10-SCC.

After doing this observation, we recall that a proper order of execution for the 3-cycles can be chosen, such that a minimum number of additional links is required for the concatenated execution

of these cycles. Let l_a , l_b and l_c be the number of additional local links needed to switch from one cycle to the other in the boundaries of the execution of the β -cycles. Then, it is possible to have at least two of these values (e.g. l_a and l_c) equal to 1 and a third value (e.g. l_b) that is at most equal to $\lfloor (n-1)/2 \rfloor$.

Therefore, the number of additional local links required for the concatenated execution of the β -cycles is at most equal to $\lfloor (n-1)/2 \rfloor + 2$. Comparing this with the number of additional local links required for the concatenated execution of three β -cycles in the proposed antipodes, we have therefore a variation $\Delta_c = \lfloor (n-1)/2 \rfloor - 2$ in the number of additional local links.

For even n , the total variation in the number of links due to the substitution of three β -cycles with two β -cycles in the proposed antipodes (I_a, P_a) is therefore:

$$\Delta = \Delta_a + \Delta_b + \Delta_c = -1 - 2 \left\lfloor \frac{n-1}{2} \right\rfloor + 2 + \left\lfloor \frac{n-1}{2} \right\rfloor - 2 = -1 - \left\lfloor \frac{n-1}{2} \right\rfloor$$

Hence, we conclude that the utilization of β -cycles in the proposed antipodes for even n actually reduces the number of links in the path to the identity. Correspondingly, the antipodes for the n -SCC cannot include β -cycles if n is even.

We leave as an exercise to the reader to confirm that a similar effect occurs if any r -cycle ($r > 3$) is included in the proposed antipodes. In any case, it is possible to confirm that the proposed antipodes are correct. \square