Circuit-Switched Structured Communications on Toroidal Meshes

by

Curtis C. Spencer

B.C.Sc., Carleton University, 1992

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the School
of
Computing Science

SIMON FRASER UNIVERSITY
February 1994

APPROVAL

Name: Curtis C. Spencer

Degree: Master of Science

Title of thesis: Circuit-Switched Structured Communications on Toroidal Meshes

Examining Committee: Dr. Binay Bhattacharya
Chair

Dr. Joseph G. Peters
Senior Supervisor

Dr. Thomas C. Shermer
Supervisor

Dr. Selim G. Akl
Examiner

Date Approved: 4 Feb. 1994

PARTIAL COPYRIGHT LICENSE

I hereby grant to Simon Fraser University the right to lend my thesis, project or extended essay (the title of which is shown below) to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users. I further agree that permission for multiple copying of this work for scholarly purposes may be granted by me or the Dean of Graduate Studies. It is understood that copying or publication of this work for financial gain shall not be allowed without my written permission.

Title of Thesis/Project/Extended Essay

CIRCUIT - SWITCHED STRUCTURED COMMUNICATIONS ON TOROIDAL MESHES

Author: _____
(signature)

CURTIS C. SPENCER.
(name)

10 February 1994
(date)

Abstract

Standard communication patterns may be grouped into two broad classifications, information disseminations or 'to-all' operations, and information permutations or 'one-to-one' operations. Information collections or 'from-all' operations are grouped with disseminations since they are simply the inverse operations. In this thesis, we develop algorithms for each of the data movement patterns within both classifications on cycles, and 2- and 3-dimensional toroidal meshes. Our algorithms take advantage of a multiple port model with circuit- switched routing and virtual channels. A linear cost model is employed in our analysis of these algorithms which takes into consideration start-up, switching and propagation costs.

The operations in the information dissemination classification that we develop algorithms for are: broadcasting, scattering (gathering), gossiping and multi-scattering. Those which we develop algorithms for in the information permutation classification include: all the global 1-, 2- and 3-dimensional permutations (e.g. reflections, rotations, translations, and transpositions), as well as several translation-based permutations. In addition, lower bounds to these problems are set forth and a comparison is made between the multiple port algorithms and their single port counterparts. The techniques used to perform the dissemination operations are based upon those which were developed for use with the one-port model. The technique used for the permutation operations is based on breaking each transformation down into simpler transformations. We show that 1-dimensional transpositions and translations can be efficiently combined or used to perform all global permutations as well as each of the special case permutations described in this thesis.

Acknowledgments

I would like to thank my senior supervisor, Dr. Joseph Peters, for the direction and assistance he has given me in researching and in writing this thesis. I would also like to thank the member of my examining committee, Dr. Thomas Shermer and Dr. Selim Akl, for their questions and comments which have improved the quality of this thesis. The faculty members and staff in the School of Computing Science have made working here at Simon Fraser University an enjoyable experience (as has the weather). Finally, my thanks to my wife, Christine, for her continued support of me in this endeavor.

To my wife
and our first child
who gave me the
incentive to finish

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In any multiple processor system or multi-computer, a key issue that must be addressed is that of providing an efficient means of communicating between processors. Building systems in which processors are directly connected to all other processors is prohibitively expensive and compromises have been made in which processors are connected via some interconnection network to all other processors. A popular interconnection network which has been used in many cases is that of the torus, or toroidal mesh. The toroidal networks have received much attention due to their low degree and ease of layout. One disadvantage of the torus is its large diameter. However, using circuit-switched routing we are able to overcome many of the limitations placed on our communications because of the diameter.

Communications patterns which occur most often can be broken down into a set of smaller structured communication problems. Providing good algorithms for these problems is key to providing an efficient means of communicating in the multi-computer. The structured communication problems that have received the most attention are broadcasting and gossiping. Several survey papers have been written which cover much of the known information about these two problems. A sampling of these surveys includes: Fraigniaud and Lazard [5], Hromkovič et al. [9], Hedetniemi, Hedetniemi, and Liestman [7], and Krumme, Cybenko, and Venkataraman [11]. In each, the authors look at the problems based upon a group of interconnection networks and a specific cost model. Most of the previous work on these problems also

used the store-and-forward approach to routing as opposed to circuit-switched routing which is now available on many systems.

This thesis looks at two main classifications of problems and analyzes them based upon a specific network and communications model. The models we use are based on those used by Fraigniaud and Peters [6] with the exception that each processor in the network has the capability of using all of its ports simultaneously as opposed to a single port. The details of our network and communications model are covered in the next chapter.

Specifically, the information dissemination problems we study are:

- *Broadcasting*: A single processor sends the same message to all other processors.
- *Scattering*: A single processor sends a personalized message to every processor.
- *Gossiping*: All processors broadcast a message to all other processors
- *Multi-scattering*: All processors scatter personalized messages to all other processors.

An additional problem, Gathering, is also studied as a part of this classification even though it is not a dissemination problem. *Gathering* (the inverse of Scattering) requires that a single node receive a personalized message from every other processor. Each of these five operations may also be performed on a subset of the network, allowing us to classify them as 'to-many' and 'from-many' operations as well. One reason why these problems have received so much attention is their generality to all interconnection schemes.

A set of less studied structured communications are problems which rely on the network topology in order to make sense. Information permutation problems or 'one-to-one' problems fit into this category. Examples of the permutation problems we provide algorithms for include:

- *Transpositions* on cycles and 2- and 3-dimensional toroidal meshes.
- *Translations* on cycles and combined translations to performs various other operations.

- *Rotations* in 2 dimensions ($\pm 90°$ and $180°$) and in 3 dimensions ($120°$ and $180°$).
- *Reflections* through the mid-point in a d-dimensional toroidal mesh.

In Chapter 2 we explain the network model including definitions of the toroidal mesh and circuit-switched routing. We also explain the communication model and provide citations to other work which use the same model.

Chapter 3 examines the lower bounds of the problems in the information dissemination classification as well as presents algorithms for each of the information dissemination problems. Our results which use an all-ports model are then compared with the results given by Fraigniaud and Peters [6] which use the one-port version of the same network and communications model.

Chapter 4 examines operations in the information permutation classification. We first present the two most basic permutations, namely the 1-dimensional transposition and translation. Using a standard form of these two operations we are able to provide algorithms to solve all other information permutations. In general we divide the permutation problems we study into two types, those which are global permutations and those which are a combination of position dependent permutations. The lower bounds for single and multi-dimensional permutations are used in comparison to show the efficiency of our algorithms. Finally, in Chapter 5 we summarize the results of this thesis.

Chapter 2

Explanation of Model

In this chapter we set forth the basics of our model including the network model on which the algorithms are based, the restrictions we place on the main components of the network, and the communication routing strategy employed. We also describe the linear cost communications model which we use to analyze our algorithms.

Network Model

The communication algorithms proposed in this thesis are based upon a d-dimensional toroidal mesh interconnection network. Toroidal meshes are not new to this thesis and have been studied by several authors in the presentation of communication algorithms [1, 2, 3, 5, 6, 10, 11, 13, 14, 15]. The d-dimensional toroidal mesh can be embedded on the surface of a torus. The basic units of a toroidal mesh are nodes and communication links. These can be modeled as the vertices and edges of a graph. Each node is represented as a vertex and consists of a processor, memory, and a router. Each link is represented by an edge and provides a connection between two nodes.

A 1-dimensional *toroidal mesh* is a linear arrangement of n nodes with links in between and an additional link joining the first node to the last node. In graph-theoretic terms, a 1-dimensional toroidal mesh is a *cycle* where each node j has neighbours labeled $(j-1) \bmod n$ and $(j+1) \bmod n$. A 2-dimensional toroidal mesh is the direct product of two 1-dimensional toroidal meshes. We define the direct product of two

graphs, $G = (V_1, E_1)$ and $H = (V_2, E_2)$, denoted by $G \times H$, as follows. The vertex set is the cartesian product $V_1 \times V_2$. There is an edge between (v_1, v_2) and (v_1', v_2') when $v_1 = v_1'$ and $(v_2, v_2') \epsilon E_2$, or $v_2 = v_2'$ and $(v_1, v_1') \epsilon E_1$. Using our definition of direct product we define a d-dimensional *toroidal mesh* as the direct product of d 1-dimensional toroidal meshes.

As an example, we illustrate in Figure 2.1 the formation of a 2-dimensional toroidal mesh as the direct product of two 1-dimensional toroidal meshes (cycles), A and B. Each node is connected both horizontally and vertically to two neighbours. The pattern that emerges is one of a square mesh with additional edges which connect the left side to the right and the top to the bottom. The 3-dimensional toroidal mesh can be viewed as a cubic mesh with edges which wrap around from the front to the back, the left to the right, and the top to the bottom. In some cases our algorithms apply to d-dimensional toroidal meshes but for the most part we restrict our attention to 1-, 2-, and 3-dimensional toroidal meshes. Since each toroidal mesh can be formed as the direct product of cycles, algorithms on cycles play a key role in the development of our algorithms for multi-dimensional toroidal meshes.

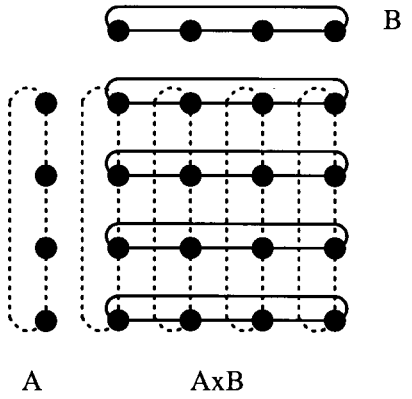


Figure 2.1: 2-dimensional Toroidal Mesh

For our study, we assume that the cycles in each dimension of the toroidal mesh have the same number of nodes, n, and that n is even unless otherwise stated. The total number of nodes for a toroidal mesh of d dimensions can then be written as N, where $N = n^d$. Having the same number of nodes in each dimension is not a

necessary constraint in the dissemination algorithms but is useful in simplifying our presentation and analysis. This constraint, however, is necessary in the permutation algorithms since a number of permutations, rotations for example, only make sense on toroidal meshes with this property. The *degree*, or number of neighbours a node has, is twice the number of dimensions. The *diameter*, or maximum distance between any two nodes, in a d-dimensional toroidal mesh is $d\lfloor\frac{n}{2}\rfloor$.

Communication algorithms depend on the capacities of the nodes and links. In particular for this thesis we assume *multi-port* communications which are also referred to in the literature as *link-bounded* [3] communications or *shouting* [7]. Multi-port communications allow each node to use all of its communication links simultaneously. In contrast, *one-port* (*processor-bounded* or *whispering*) communications permit the use of only one link at any given time. We also assume that the communication links are *full-duplex* so that messages can travel across the same link in both directions simultaneously. For each link that is connected to a node we consider the node to have an input port and an output port, such that messages received over the link enter the node through the input port and message to be transmitted over the link exit the node through the output port. Using this model, in any given round of operation a single node in a 2-dimensional toroidal mesh can send up to four messages out through its output ports and receive up to four messages through its input ports. The input and output ports of a node if not used in this manner may be used to route messages through the node. We assume that a node can *switch through* a message by connecting an input port to an output port. In the 2-dimensional toroidal mesh as many as four messages can be switched through a node.

As part of this thesis, we compare information dissemination algorithms based on multi-port communications with those algorithms obtained by Fraigniaud and Peters [6] using one-port communications. We justify being able to make the comparison since the additional memory control costs required by multi-port communications are considered negligible [6] in comparison to other costs.

Interconnection networks such as toroidal meshes are used because connecting each node to all other nodes is prohibitively expensive. Consequently, communications between non-neighbouring nodes require messages to be switched through intermediate

nodes. Two main strategies are used for this routing between nodes: *store-and-forward* and *circuit-switched.*

In *store-and-forward* routing messages are stored in buffers at each intermediate node along the path to the destination node. Once the entire message is received by an intermediate node the message is forwarded on to the next node along the path. A variation on this is *pipelined* store-and-forward routing which uses the links more effectively by partitioning the message into packets that are sent one after the other along the path using store-and-forward routing. Many of the communication problems studied in this thesis have been studied previously using store-and-forward routing on the torus. Store and-forward algorithms for broadcasting and gossiping have been presented in [1, 3, 5, 7, 11, 15, 16] and store-and-forward algorithms for transpositions have been presented in [2, 10].

In *circuit-switched* routing a header containing the destination address is sent to "build" a path. At each intermediate node on the path, the input port and output port used by the header are connected. Eventually, the circuit is complete and an acknowledgment is sent back informing the source node to begin sending the message. The message is sent in packets in a pipeline fashion and the final packet disconnects the input and output ports in the intermediate nodes as it passes through. In circuit-switched routing the message is not examined or stored at the intermediate nodes as it is switched through. A derivative of this circuit-switched routing is *wormhole* routing where a header is sent to build a path and instead of waiting for an acknowledgment the message packets are pipelined behind the header with the last packet releasing the switches as it passes through.

Peters and Syska point out in [14] that the large diameters of the toroidal meshes are a disadvantage when store-and-forward routing is used because the communication time for store-and-forward routing is proportional to the diameter of the network. In many recent multi-computers such as the Intel Touchstone Delta, AP100, Symult 2010, nCUBE-2 and iWARP store-and-forward routing has been replaced by circuit-switched routing. Circuit-switched routing is less dependent on the diameter of the network and for this reason is a practical choice. Circuit-switched routing can emulate store-and-forward routing by creating circuits of length one only. Store-and-forward

routing cannot emulate circuit-switched routing.

When using circuit-switched routing for *non-structured* communications one of the most important characteristics is that the routing is deadlock-free. To implement simple deadlock-free routing on most topologies, several *virtual channels* or links can be multiplexed onto a single physical link. Fraigniaud and Peters [6] discuss one implementation of virtual channels and use virtual channels in their study of *structured* communications. In this thesis our communication patterns are also structured and are deadlock-free. Virtual channels are useful even with structured communicaion patterns since they allow for more uniform communications. Links in high demand make full use of the virtual channels available while links in low demand require only a partial or minimal use of virtual channels. In our network we assume a constant number of virtual channels, q. When q virtual channels are in use we can "simultaneously" transmit q different messages over the same link. The communication time to transmit q messages simultaneously is approximately q times greater than the time to transmit a single message. Our algorithms (with gossiping and multi-scattering as exceptions) can also be performed without the use virtual channels.

Communication Model

For each algorithm we present, we analyze the communication time using a linear cost model. Several of the previously cited papers using store-and-forward routing perform their analysis with a linear cost model [1, 2, 3, 5]. The linear cost model we use for our circuit-switched algorithms has been described previously in [6, 14].

For each of our algorithms the cost of the algorithm is the time required to perform the algorithm. In general, each algorithm consists of synchronous rounds of communication where the total time is the sum of the maximum communication times in each round.

Using circuit-switched routing, the time required to send a message of length L along i links takes time, $\alpha + i\delta + L\tau$. α is the *start-up time* or the time required to set up the communications. It may include a hand-shaking protocol to insure that the receiver has enough memory to store the message. δ is the *switching delay*, or the

time required by the router to establish communications (connect an input port to an output port) through a node. τ is the *propagation time*, or the time required to send a single unit of information the length of the circuit. In practice, the *propagation time*, τ, is proportional to the length of the message [6]. The length of the circuit (i.e. path) is considered to have little effect on the propagation time, thus we ignore the circuit length and consider the propagation time proportional only to the length of the message. In our model we assume uniform message lengths where each message contains L units of information. In order to provide a uniform propagation time of τ we assume a uniform bandwidth of $\frac{1}{\tau}$ in the network. The *bandwidth* is the amount of information that can be transmitted on a link during a unit of time. Fraigniaud and Peters in [6] show that this is a reasonable model of real machines (i.e. *i*PSC/860).

When q virtual channels are used to multiplex q messages across a physical link, the propagation cost is multiplied by a factor of q since virtual channels do not increase the bandwidth of the links.

The cost of multiplexing the virtual channels is shown by Fraigniaud and Peters in [6] to be very minor in comparison to the start-up and propagation costs. As well the costs of sending the header packet and receiving the acknowledgment are also stated as being minor costs. In a related paper, Seidel [18] presents a larger list of additional cost factors which can affect real message passing networks. From his simulation results on the Intel Delta mesh he showed that these factors attribute very little to the overall cost and therefore are ignored in his analysis. Considering the α, δ, and τ terms as the major factors affecting the time of our algorithms we ignore the minor costs associated with the other factors outlined in the papers cited above.

It should also be pointed out that in most current machines, message transmissions are initiated in software and switching is done in hardware, so δ is usually much smaller than α. In general it should be kept in mind throughout this thesis that, $\delta << \alpha + L\tau$.

In presenting our algorithms it is often the case that tradeoffs exist between the different factors. For example, the number of rounds can often be decreased by an increase in the propagation costs. The algorithms we present attempt to approach their lower bounds with the emphasis being on minimizing the number of rounds without causing more than a minor increase in the propagation cost.

Chapter 3

Information Disseminations

In this chapter we present the algorithms for the four basic information dissemination problems on circuit-switched toroidal meshes using the linear cost model for analysis. The communications problems of this designation are: Broadcasting (one-to-all communications), Scattering (one-to-all personalized communications), Gossiping (all-to-all communications), and Multi-Scattering (all-to-all personalized communications). Each of these can be used as stated or adapted for use on some subset of the entire network, making them one-to-many or many-to-many problems. A fifth operation we study in this chapter is the Gathering operation (all-to-one personalized communications). Gathering is the inverse operation of Scattering and is covered in the same section (Section 3.3).

In Section 3.1 we present the lower bounds of the all-ports model in comparison to those known for the one-port model for circuit-switched toroidal meshes. In Sections 3.2 to 3.5 we present the algorithms for each dissemination problem. In the last section we present a summary of our results and compare these results with those already known for the one-port model.

3.1 Lower bounds

Fraigniaud and Peters [6] present algorithms and lower bounds for broadcasting, scattering, gossiping and multi-scattering on a cycle using the same model we use, with

the restriction of sending and receiving on one-port instead of all-ports. Table 3.1 gives a summary of the one-port lower bounds from [6] in comparison to our all-ports lower bounds. The all-ports broadcasting lower bound is taken from [14] by Peters and Syska.

Table 3.1: Lower Bounds on Communication Times on a d-dimensional Torus

Broadcasting	one-port	$\max\{d\log_{q+1}(n)\alpha,\ d\frac{n}{2}\delta, L\tau\}$
	all-ports	$\max\{d\log_{2dq+1}(n)\alpha, d\frac{n}{2}\delta, \frac{L\tau}{2d}\}$
Scattering	one-port	$\max\{d\log_{q+1}(n)\alpha,\ d\frac{n}{2}\delta, (N-1)L\tau\}$
	all-ports	$\max\{d\log_{2dq+1}(n)\alpha, d\frac{n}{2}\delta, \frac{(N-1)}{2d}L\tau\}$
Gossiping	one-port	$\max\{d\log_{q+1}(n)\alpha,\ d\frac{n}{2}\delta, (N-1)L\tau\}$
	all-ports	$\max\{d\log_{2dq+1}(n)\alpha, d\frac{n}{2}\delta, \frac{(N-1)}{2d}L\tau\}$
Multi-Scattering	one-port	$\max\{d\log_{q+1}(n)\alpha,\ d\frac{n}{2}\delta, \frac{nN}{8}L\tau\}$
	all-ports	$\max\{d\log_{2dq+1}(n)\alpha, d\frac{n}{2}\delta, \frac{nN}{16d}L\tau\}$

The lower bounds are presented as the maximum of three independent lower bounds since it very difficult to provide a cumulative lower bound. The three independent lower bounds are based on the α, δ, and τ terms.

From the table we observe that the lower bound on the number of rounds (α term) is the same for all four operations using the one-port model. Similarly, this is true for the all-ports model. In all four cases, using either model, each source node must send a message (unique or not) to all other nodes. If we consider the one-port model, in the first round a single node can only inform one other. In each subsequent round each informed node can only inform one other and thus number of informed nodes at most doubles. In order for all nodes to be informed $\lceil \log_2 N \rceil$ rounds are required, where N is the total number of nodes. When we consider the use of q virtual channels we find that each node can inform q other nodes and that $\lceil \log_{q+1} N \rceil$ rounds are required to inform all nodes.

In the all-ports model during the first round the originating node can inform one node through each of its $2d$ ports. After the first round, $2d+1$ nodes have been

informed and in each subsequent round each informed node can also inform $2d$ other nodes. Thus in order for all nodes to be informed using this model $\lceil \log_{2d+1} N \rceil$ rounds are required. Again if we introduce q virtual channels each informed node can inform $2dq$ other nodes and the total number of rounds required becomes $\lceil \log_{2dq+1} N \rceil$.

In Table 3.1 we use the substitution of $N = n^d$ in order to make the lower bounds comparable to the results listed in Table 3.2 at the end of this chapter. In addition to simplify the presentation of equations throughout this thesis we will ignore the ceiling and floor functions.

For the lower bound on the switching delay (δ term), we note that since a single node must communicate with all other nodes, the lower bound can be calculated as the maximum distance between any two points in the torus (i.e. the diameter). The diameter of a d-dimensional torus is $d\frac{n}{2}$. Using virtual channels does not affect this bound nor does it affect the bound on the propagation cost since virtual channels do not shorten the distances between nodes nor do they increase the number of available ports.

The lower bound which varies for these operations is the propagation time (τ term). In the one-port model, under broadcasting the source node must send at least one message of length L into a network of bandwidth $\frac{1}{\tau}$, resulting in a lower bound of $L\tau$. Using all-ports, the message may be broken up and sent by a node through all of its ports. Peters and Syska [14] give this broadcasting lower bound as $\frac{L}{2d}\tau$.

A similar result holds for scattering. In scattering the source node must send $N-1$ messages, resulting in a lower bound of $(N-1)L\tau$ for the one-port model. Assuming an all-ports model, the source node can divide the $N-1$ messages evenly and scatter them through all of its ports resulting in a lower bound of $\frac{(N-1)}{2d}L\tau$. The argument for gossiping is simply the scattering argument turned around. Each node must receive $N-1$ messages from a network with a bandwidth of $\frac{1}{\tau}$. When only one port is in operation at any given time the lower bound is $(N-1)L\tau$. With all-ports in operation the destination node can receive the $N-1$ messages through its $2d$ ports resulting in a lower bound of $\frac{(N-1)}{2d}L\tau$.

The arguments presented by Fraigniaud and Peters [6] for the multi-scattering propagation time apply directly to both the one-port and all-ports models. The term

$\frac{nN}{8p}L\tau$ is derived from the communication capacity required (circuit lengths times expected traffic) divided by the total bandwidth available. The communication capacity required to perform a scatter from a node is $(\frac{1}{p}\sum_{i=1}^{D} i\Gamma(i)L = \frac{ndN}{p4}L)$, where D is the diameter, $\Gamma(i)$ is the number of vertices at distance i from a given vertex and p is the number of ports (1 or $2d$). Since the nodes within a torus are vertex transitive we know that the same cost applies to all nodes making the communication capacity required for the multi-scatter operation: $N\frac{ndN}{p4}L$. The total bandwidth available is $2dN\frac{1}{\tau}$ (the total number of links, both directions).

3.2 Broadcast, one-to-all

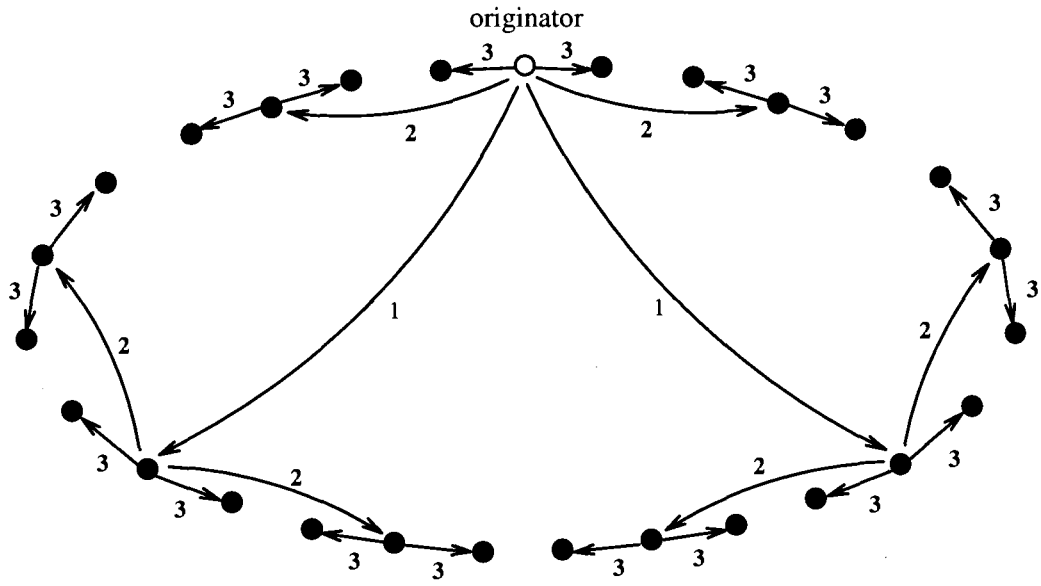


Figure 3.1: Broadcasting on a cycle

Broadcasting is the operation where a single node (the originator) has a single message to send to all other nodes within the network, in our case within the torus. Broadcasting within the cycle using all ports is illustrated in Figure 3.1. In the first round the cycle is divided into three equal sections and the originator sends a message (distance $\frac{n}{3}$) to the two sections it is not contained in. In each succeeding round, the sections are again divided into three and the informed node in the middle section sends

its message (distance $\frac{n}{3^r}$, where r is the round number) to the other two sections. This pattern of communication continues until each section only contains a single node. The paths used in each round are numbered on the diagram. The number of rounds required to inform the nodes of the cycle is computed as $\log_3(n)$. The switching time, (δ term), is the sum of the distances traveled in each round ($\sum_{i=1}^{\log_3(n)} \frac{n}{3^i} \approx \frac{n}{2}$). Finally, the propagation cost, $\log_3(n)L\tau$, is computed using the sum of the message lengths (L) transmitted in each round.

If we allow the use of q virtual channels, it is possible that a node could inform $2q$ other nodes within the cycle rather than just two. This would cause the number of rounds required to be $\log_{2q+1}(n)$. The switching time is calculated using the same method as above and results in the same time. The propagation cost is q times the number of rounds since there are q messages in each round being multiplexed. This gives a total propagation cost of $q\log_{2q+1}(n)$. Formula 3.1 gives the cost of broadcasting in a cycle with q virtual channels. If we substitute $q = 1$ we have the cost of broadcasting where each physical link is used by only one channel (i.e. the virtual channel capability is not used). From the equation below we see a tradeoff in the use of virtual channels. Virtual channels reduce the number of rounds by increasing the propagation time.

$$\log_{2q+1}(n)\alpha + \frac{n}{2}\delta + q\log_{2q+1}(n)L\tau \tag{3.1}$$

One method of broadcasting within a d dimensional torus is to perform a cycle broadcast in each of the dimensions sequentially, requiring d steps. Since we have use of all ports it is possible for us to broadcast in all dimensions using cycle broadcasts in each of the d steps required above. Using this algorithm we are able to establish d disjoint paths between the originator and any one destination. Thus we can reduce the propagation cost by a factor of d by dividing each of the messages into d parts and sending one part along each path. Fraigniaud [3] in his thesis used this same method of simultaneous broadcasts to perform broadcasting on a torus using an all-ports store-and-forward model. An example illustrates how this takes place. Let the node in the bottom left corner of a 2-dimensional toroidal mesh be the originator (this in fact could be any node since the torus is vertex transitive). The broadcast

will require, in this case, two sets of cycle broadcasts. In the first set the originator informs the row it is in of one half of the message and the column it is in of the other half of the message. In the second set of cycle broadcasts, each node in the bottom row broadcasts the half message it learned to its respective column, and each nodes in the left column broadcasts the half message it learned to its respective row.

In d dimensions, we can use the tuple $(m_1,m_2,\ldots,m_d)$ to represent which part of the message is being broadcast in each dimension. Since we want each part of the message broadcast in each dimension we can use the following scheme to achieve the broadcast: $(1,2,\ldots,d)$, $(2,3,\ldots,d,1)$, $\ldots$, $(d,1,\ldots,d-1)$. Using this scheme, the number of rounds and switching cost are both d times larger than in a cycle since there are d sets of cycle broadcasts. The propagation cost is the same as in a cycle since the number of ports available also increases by a factor of d. Therefore broadcasting on a d-dimensional torus by this method results in a cost of:

$$d\log_{2q+1}(n)\alpha + d\frac{n}{2}\delta + q\log_{2q+1}(n)L\tau \tag{3.2}$$

3.3 Scatter (Gather), one-to-all (all-to-one) personalized

The scatter operation involves a single originator sending individualized messages to each other node. In order to perform this operation on a cycle we use the same paths (see Figure 3.1) that were used in broadcasting but send bundles of messages along each path rather than just a single message. The bundles consists of all the messages that the receiving nodes will have to forward to other nodes (including itself). Thus to scatter, we send packets of decreasing size until in the final round we send packets of size L (i.e. $\frac{n}{3}L, \frac{n}{9}L, \frac{n}{27}L, \ldots, L$). The number of rounds and switching cost are the same as those in the broadcast operation since the same pattern is followed. The propagation cost can be computed using the sum of the size of packets sent in each round multiplied by τ $(n(\sum_{i=0}^{\log_3(n)} \frac{1}{3}^i)L\tau = \frac{n-1}{2}L\tau)$. When virtual channels are used, the cost can still be computed in the same manner and results in the same cost since the decrease of q in the message size is offset by the added cost of multiplexing the q

virtual channels. The total cost for scattering in a cycle is:

$$\log_{2q+1}(n)\alpha + \frac{n}{2}\delta + \frac{n-1}{2}L\tau \tag{3.3}$$

In order to perform the scatter operation on a d-dimensional torus (with N nodes where $N = n^d$) we apply the same method that was used for broadcasting with the exception that the packets used in the scattering operations are of different lengths. In the first set of cycle scatterings we scatter bundles of messages of size $(n^{d-1})\frac{L}{d}$ since messages are of size $\frac{L}{d}$ and each node informed during the first round will need to inform a torus with one less dimension in the following rounds. This results in a propagation cost of $(\frac{n-1}{2})(n^{d-1})\frac{L}{d}\tau$ for the first set of cycle scatterings. In each succeeding set of cycle scatterings the bundle sizes are reduced by a factor of n since each informed node is required to scatter only in the remaining sets of cycle scatterings. Using this pattern we can calculate the total propagation cost using the following equation $(\frac{n-1}{2}\sum_{i=0}^{d-1} n^i)\frac{L}{d}\tau = \frac{N-1}{2d}L\tau$. The number of round s and the switching cost is the same as described for broadcasting in Formula 3.2. Therefore, the total cost for scattering in d dimensions is:

$$d\log_{2q+1}(n)\alpha + d\frac{n}{2}\delta + \frac{N-1}{2d}L\tau \tag{3.4}$$

The reverse operation, gather or all-to-one personalized, requires the same time as the scatter operation and is performed by simply reversing the steps described above.

3.4 Gossip, all-to-all

Gossiping is the operation where each node has a message it broadcasts to all other nodes in the network. Our approach to the problem follows the 2-phase gossiping scheme introduced by Fraigniaud and Peters [6]. Our network model differs from theirs only in the number of ports available, all ports instead of one port. Gossiping using the 2-phase scheme is only useful in the presence of virtual channels ($q \geq 2$).

The general concept of the 2-phase gossip scheme is illustrated in Figures 3.2 and 3.3. In the first phase, we divide the cycle into k virtual cycles (with $\frac{n}{k}$ nodes in

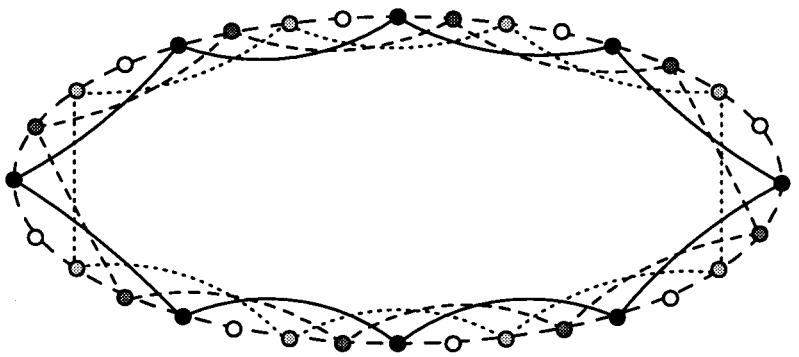


Figure 3.2: Gossip - Phase 1 - Four virtual cycles of eight nodes each

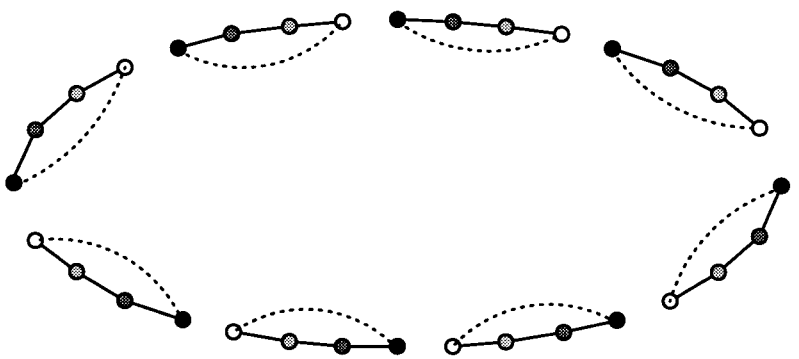


Figure 3.3: Gossip - Phase 2 - Eight sub-cycles with constant contention (q=2)

each cycle) and gossip along the virtual cycle. Figure 3.2 illustrates how these virtual cycles are set up. After phase 1, each node contains $\frac{n}{k}$ messages.

The cycles used in the second phase are created by combining k adjacent nodes, one from each of the k virtual cycles. In order to complete the cycle, we add a virtual path of length $k-1$ to the cycle. This is illustrated in Figure 3.3. Once we have gossiped on the phase 2 cycles, each node has received the messages from all other nodes in the whole cycle ($k\frac{n}{k} = n$).

How we gossip on these two sets of cycles determines our cost. The method used in the one-port model by Fraigniaud and Peters [6] is to exchange messages between pairs of nodes. On a cycle with an even number of nodes, the edges can be 2-coloured such that each node has a Blue and Red edge entering it (Blue and Red edges alternate

through the cycle). In the first round of gossiping each node exchanges a single message with its neighbour along the Blue edge. In the second round each node has two messages it can exchange with its Red neighbour. In each successive round we alternate between exchanging two messages along the Blue and Red edges. In total, $\frac{p}{2}$ rounds are required to complete gossiping in a cycle with p nodes, and there is no contention for links.

An all-ports version of gossiping on the cycle allows each node to exchange a single message with both its neighbours in each round. In the first round a node exchanges its own message with each neighbour. In each successive round it passes the message it received from its left neighbour to its right neighbour and the message from its right neighbour to its left. Again a total of $\frac{p}{2}$ rounds are required to complete gossiping in a cycle with p nodes, and there is no contention for links. The all-ports version has the advantage of sending only one message through a port in any given round reducing the propagation cost.

When all-ports gossiping is used to implement phase 1 of our scheme the maximum number of virtual cycles we can use is q (i.e. $k \leq q$), where q is the number of virtual channels available. Each cycle in phase 1 has $\frac{n}{k}$ nodes, thus $\frac{n}{2k}$ rounds are required. The switching cost is $\frac{n}{2}$ which is the maximum distance a message is switched. The propagation cost of this first phase, $\frac{n}{2}L\tau$ is calculated as the number of rounds ($\frac{n}{2k}$), multiplied by the contention (k), the message size[1] (L), and τ. The cost of phase 2 is calculated in the same manner, the number of rounds being $\frac{k}{2}$. The switching term is $\frac{k}{2}(k-1)$ since in each round a message must travel along the virtual path which has length $k-1$. The propagation term is calculated using the same method as above resulting in a cost of $\frac{k}{2}2\frac{n}{k}L = nL$. Using $q = k$, these total for a result of:

$$(\frac{n}{2q} + \frac{q}{2})\alpha + (\frac{n}{2} + \frac{q^2}{2} - \frac{q}{2})\delta + \frac{3n}{2}L\tau \tag{3.5}$$

The number of rounds in the above equation is significantly higher than the number of rounds required when the one-port model is used (see Table 3.2). The increase is attributed to the first phase of gossiping where using the all-ports model we can

[1]The message size is the size of the message being gossiped. In phase 1 each node is gossiping its own message. In phase 2 each node is gossiping the $\frac{n}{k}$ messages of size L it learned in phase 1.

have at most q virtual cycles (every link is used in every round). Using the one-port model which uses alternating links in different rounds we can use approximately $2q$ virtual cycles. With more virtual cycles in the one-port algorithm we are able gossip in phase 1 using approximately half the number of rounds but with a propagation cost in each round of approximately double that of the all-ports algorithm (resulting in approximately the same total propagation cost). In total we find that a tradeoff exists between the two algorithms. The one-port algorithm for phase 1 requires approximately half the number of rounds and the all-ports algorithm has a slightly lower propagation cost. Phase 2, however, is free from these problems since the contention is constant and the tradeoff between the α and τ terms does not exist. Considering the reduction in the number of rounds greater than the reduction in the propagation cost in phase 1, we find that our best solution is a hybrid algorithm which uses the one-port algorithm in the first phase and the all-ports algorithm in the second phase.

The cost of one-port gossiping in phase 1 is given as $\frac{n}{2k}\alpha + \frac{n}{2}\delta + (\frac{n}{2} + \frac{n}{k} - \frac{k}{2} - 1)L\tau$ in [6]. Combining this with the cost of all-ports gossiping in phase 2 gives us a new total cost for gossiping in the cycle. Formula 3.6 represents this cost where $q \approx \frac{k}{2}$. Thus on the cycle we are able to maintain the number of rounds required by the one-port algorithm and reduce the propagation term from the one-port algorithm by approximately one quarter.

$$(\frac{n}{4q} + q)\alpha + (\frac{n}{2} + 2q^2 - q)\delta + (\frac{3n}{2} + \frac{n}{2q} - q - 1)L\tau \tag{3.6}$$

In order to gossip on a d-dimensional torus, we use the same approach that was used to broadcast on a d-dimensional torus. Each message is again divided into d parts and we perform d sets of d simultaneous gossiping operations, one in each dimension. Once we have completed the first set of gossips, each node knows $d\frac{n}{d}$ messages. After the second set each node knows $d\frac{n^2}{d}$ messages. Once the d sets are complete each node knows all $d\frac{n^d}{d} = N$ messages. Since we gossip using the cycle gossiping strategy d times, we simply multiply the number of rounds and the switching cost by d. The propagation cost is multiplied by a factor of $\frac{1}{d}\sum_{i=0}^{d} n^i = \frac{N-1}{d(n-1)}$, since in each set of cycle gossips the number of messages which each node must gossip increases by n. The $\frac{1}{d}$ factor represents that we originally divided the messages by d. Applying these

factors to Formula 3.6 we present below the cost of gossiping on a d-dimensional torus.

$$d(\frac{n}{4q}+q)\alpha + d(\frac{n}{2}+2q^2-q)\delta + (\frac{3n}{2}+\frac{n}{2q}-q-1)\frac{N-1}{d(n-1)}L\tau \qquad (3.7)$$

3.5 Multi-scatter, all-to-all personalized

The multi-scatter problem is that of sending personalized messages from every node to every other node. By analogy to the broadcasting and scattering problems, we can apply the same method employed in gossiping to multi-scattering. The only term which is then affected is the propagation time due to the varying size of bundles.

How we multi-scatter on the cycles in each of the phases determines our cost. Multi-scattering by the one-port algorithm [6] requires that each node exchanges half ($\frac{p}{2}$, where p is the number of nodes and p is even) of its messages with one of its neighbours. In the second round, each node exchanges the other half ($\frac{p}{2}-1$) of its messages plus the messages it learned in the first round (removing the message for itself) in the other direction. In each succeeding round the exchange edges alternate and each node forwards on the number of messages it receives minus the two for itself. The propagation cost, $\frac{p^2}{4}L\tau$, is determined as the sum of the messages moved in each round times τ ($\frac{p}{2}L + 2\sum_{i=1}^{\frac{p}{2}}(\frac{p}{2}-i)L = \frac{p^2}{4}L$).

Multi-scattering by the all-ports method requires that in the first round each node sends half ($\lfloor\frac{p}{2}\rfloor$, where p is odd) of its information out each port. In the second and each succeeding round each node forwards all the messages it received minus the one for itself in the same direction the messages were initially traveling. The propagation cost, $(\frac{p^2}{8}-\frac{p}{4})L\tau$, is again determined as the sum of the messages moved in each round times τ ($\lfloor\frac{p}{2}\rfloor L + \sum_{i=1}^{\lfloor\frac{p}{2}\rfloor}(\lfloor\frac{p}{2}\rfloor - i)L \approx (\frac{p^2}{8}+\frac{p}{4})L$).

Applying these simple multi-scattering techniques in our 2-phase circuit switched algorithm we can determine the cost of the overall algorithm. For each phase all we have to calculate is the new propagation term since the number of rounds and switching time is the same as it was for gossiping. In Phase 1, the number of nodes,p, in the cycle is $\frac{n}{k}$. In order to multi-scatter each node sends its messages in bundles of k so that in the second phase these messages can be multi-scattered to their correct

destinations. The total propagation cost using the one-port model, $(\frac{n^2}{8}+\frac{n^2}{4k})L\tau$, can be determined as the sum of the message sizes moved in each round, $(\frac{\frac{n}{k}^2}{4}L)$, multiplied by the size of the message bundles and the contention, $\frac{k}{2}-1$. The total propagation cost using the all-ports model, $(\frac{n^2}{8}+\frac{nk}{4})L\tau$, can be determined as the sum of the message sizes moved in each round, $((\frac{\frac{n}{k}^2}{8}+\frac{\frac{n}{k}}{4})L)$, multiplied by the size of the message bundles and the contention, k.

In Phase 2 the number of nodes, p, in the cycle is k. From the multi-scatter in the first phase each node has a total of $\frac{n}{k}$ messages which it must multi-scatter to every other node in its Phase 2 cycle. The total propagation cost using the all-ports model, $(\frac{nk}{4}+\frac{n}{2})L\tau$, can be determined as the sum of the message sizes moved in each round, $((\frac{k^2}{8}+\frac{k}{4})L)$, multiplied by the size of the message bundles, $\frac{n}{k}$, and the contention, 2.

In Formula 3.5 we presented the cost of gossiping using the all-ports algorithm in both phases ($q=k$). In Formula 3.8 we present the cost of multi-scattering using the same model.

$$(\frac{n}{2q}+\frac{q}{2})\alpha+(\frac{n}{2}+\frac{q^2}{2}-\frac{q}{2})\delta+(\frac{n^2}{8}+\frac{(q+1)n}{2})L\tau \tag{3.8}$$

In order to reduce the number of rounds to the same level as was reported for one-port multi-scattering [6], we apply the same hybrid method we did for gossiping. Using the one-port algorithm in the first phase followed by using the all-ports algorithm in the second phase, we are able to multi-scatter in the same number of rounds. The cost for this hybrid multi-scattering is shown below, where $q \approx \frac{k}{2}$.

$$(\frac{n}{4q}+q)\alpha+(\frac{n}{2}+2q^2-q)\delta+(\frac{n^2}{8}+\frac{n^2}{8q}+\frac{(q+1)n}{2})L\tau \tag{3.9}$$

When we apply multi-scattering to the d-dimensional torus we multiply each term by the same factors we did for gossiping. For the number of rounds and switching term this is d. In the case of the propagation term, the size of the message bundles decreases by the same amount it increased for gossiping at each stage. This results in the same factor for multi-scattering. The results are shown in Table 3.2.

3.6 Summary

Table 3.2 summarizes the results[2] from this chapter. From an analysis of Table 3.1 we see that in general terms the addition of multiple ports reduces the lower bounds on the number of rounds by a factor of $\frac{d\log_{q+1}(n)}{d\log_{2dq+1}(n)} = \log_{q+1}(2dq+1)$. and on the propagation cost by a factor of $2d$. With these reductions on the lower bounds we would expect similar reductions in the cost of our algorithms.

Table 3.2: Dissemination Communication Times on a d-dimensional Torus

Broadcast	one-port	$d\log_{q+1}(n)\alpha$	$+d\frac{n}{2}\delta$	$+dq\log_{q+1}(n)L\tau$
	all-ports	$d\log_{2q+1}(n)\alpha$	$+d\frac{n}{2}\delta$	$+q\log_{2q+1}(n)L\tau$
Scatter	one-port	$d\log_{q+1}(n)\alpha$	$+d\frac{n}{2}\delta$	$+(N-1)L\tau$
	all-ports	$d\log_{2q+1}(n)\alpha$	$+d\frac{n}{2}\delta$	$+\frac{N-1}{2d}L\tau$
Gossip	one-port	$d(\frac{n}{4q}+q)\alpha$	$+d(\frac{n}{2}+q^2)\delta$	$+(2n-\frac{n}{2q}-q-1)\frac{N-1}{n-1}L\tau$
	hybrid	$d(\frac{n}{4q}+q)\alpha$	$+d(\frac{n}{2}+2q^2-q)\delta$	$+(\frac{3n}{2}+\frac{n}{2q}-q-1)\frac{N-1}{d(n-1)}L\tau$
	all-ports	$d(\frac{n}{2q}+\frac{q}{2})\alpha$	$+d(\frac{n}{2}+\frac{q^2}{2}-\frac{q}{2})\delta$	$+(\frac{3n}{2})\frac{N-1}{d(n-1)}L\tau$
Multi-Scatter	one-port	$d(\frac{n}{4q}+q)\alpha$	$+d(\frac{n}{2}+q^2)\delta$	$+((\frac{q+1}{q})\frac{n^2}{8}+\frac{3qn}{4})\frac{N-1}{n-1}L\tau$
	hybrid	$d(\frac{n}{4q}+q)\alpha$	$+d(\frac{n}{2}+2q^2-q)\delta$	$+((\frac{q+1}{q})\frac{n^2}{8}+\frac{(q+1)n}{2})\frac{N-1}{d(n-1)}L\tau$
	all-ports	$d(\frac{n}{2q}+\frac{q}{2})\alpha$	$+d(\frac{n}{2}+\frac{q^2}{2}-\frac{q}{2})\delta$	$+(\frac{n^2}{8}+\frac{(q+1)n}{2})\frac{N-1}{d(n-1)}L\tau$

Comparing the costs between the one-port and all-ports models we find that for both broadcasting and scattering the number of rounds is reduced by a factor of $\log_{q+1}(2q+1)$, where q is the number of virtual channels. The propagation term is also reduced by factors of $1.58d$ and $2d$, for broadcasting and scattering respectively.

Using the all-ports model it is possible to obtain better bounds for the broadcasting and scattering algorithms on a 2-dimensional torus. Peters and Syska [14], and Park, Lee, and Choi [13] present better 2-dimensional algorithms which take advantage of tiling patterns on the torus. In order to obtain these results the authors approach the torus as a 2-dimensional network and not as a product of cycles. We hypothesize

[2]To simplify the presentation and analysis the ceiling and floor functions have been dropped.

based on these results that in order to obtain the optimal number of rounds on a d-dimensional torus, broadcasting and scattering algorithms must be based upon a d-dimensional not 1-dimensional pattern.

For the gossiping and multi-scattering algorithms the difference between the one-port and multi-port model is a tradeoff rather than a strict improvement. The tradeoff is in favour of the propagation term over the switching term, where the additive factor to the switching term[3] is small (it depends only on d and q). In both operations the switching cost increases by an additive factor of $(dq^2 - dq)\delta$ and the propagation cost decreases by a multiplicative factor of approximately $1.33d$ and $1d$ for gossiping and multi-scattering respectively.

Several methods were attempted to use an all-ports algorithm in phase 1 of the 2-phase scheme for the 'all-to-all' operations. Of these, none outperformed the one-port algorithm already provided in [6]. The problem that was found was that all-port algorithms created at least as much contention (if not double) as the one-port algorithm. By creating extra contention it was not possible to reduce the overall cost. The reason we were able to reduce the cost of phase 2 was because the contention was constant and the all-ports algorithm had a lower propagation cost. This translated into larger cost savings for gossiping since its propagation cost is higher in phase 2, and into a smaller cost savings for multi-scattering which has its higher propagation cost in phase 1 (which was not changed).

From our study of these problems we hypothesize that the additional capabilities of multiple ports are best applied to problems that have bandwidth open for use as was the case in the 'one-to-all' problems.

[3]The switching term in our communications model is also the smallest of the three cost terms.

Chapter 4

Information Permutations

In this chapter we present the algorithms for operations in the information permutation classification on 1-,2- and 3-dimensional toroidal meshes. These operations can be divided into two types of permutations. The first type are called global permutations and are uniform permutations (e.g. rotations, transpositions, and reflections) on the cartesian axes. The second type are translation based permutations in which nodes vary the amount of translation they are required to perform based upon their position in the mesh or according to the dimensions of the mesh. Each permutation of either type can be performed as a combination of 1-dimensional transpositions or 1-dimensional translations.

Prior to the presentation of the algorithms we present the approach and common procedures followed in developing these algorithms. The basic 1-dimensional transposition and the very closely related 2-dimensional transposition are presented in Section 4.2. These operations are used extensively in Section 4.4 where we present the algorithms for the global permutations. The basic 1-dimensional translation is presented in Section 4.3. This operation has application to both types of permutations (Sections 4.4 and 4.5). In Section 4.6 we present the lower bounds on d-dimensional permutations to act as a guide in determining the efficiency of our algorithms. Finally, we conclude this chapter with a summary of our results. Once again in this chapter we simplify the presentation and analysis of our results by ignoring the ceiling and floor functions.

4.1 Common procedures

4.1.1 Labeling

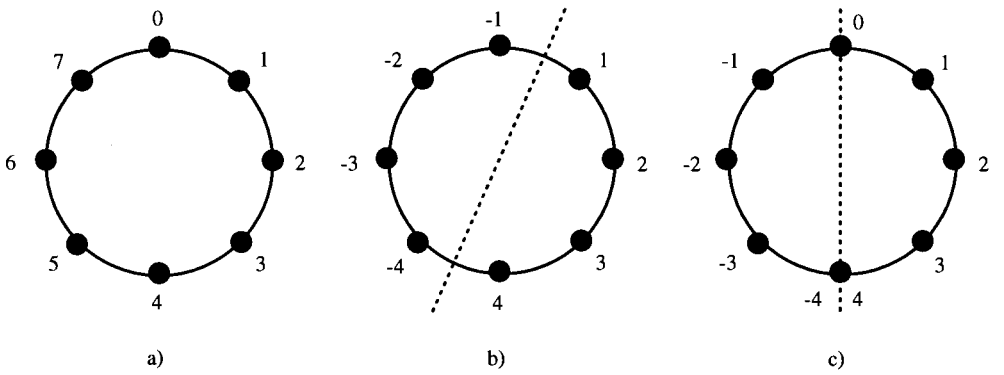


Figure 4.1: a) Natural and b,c) Symmetric labelings of C_8

The labeling of nodes is an important issue in one-to-one operations. We will be using two types of labeling, natural and symmetric. Natural labeling of a cycle is done with labels starting with zero and increasing by one as they go around the cycle (i.e. from 0 to n-1, where n is the number of nodes). Symmetric labeling has several forms. We can perform the labeling by placing a mirror on the cycle. Those on one side of the mirror are labeled with increasing numbers starting at 1. Those in the reflection have negative labeling. Figure 4.1 depicts these two types of labeling. It is possible in symmetric labeling to have zero, one or two nodes on the line of reflection. If they are present, nodes on the line of reflection are labeled 0 and $\pm\frac{N}{2}$.

Transpositions can also be thought of as reflections and are best described under symmetric labeling, that is 3 is mapped onto -3, and -3 is mapped onto 3 and so forth. Translations on the other hand are best described under natural labeling. A translation might be of the form $newx = (x + 3) \bmod n$. In this way each node communicates with the node ahead of it by three in the positive direction.

Symmetric labelings on multi-dimensional toroidal meshes require that we fix the axes such that they divide each dimension in half $(-\frac{n}{2}, -\frac{n}{2}, \ldots) \rightarrow (\frac{n}{2}, \frac{n}{2}, \ldots)$. Natural labelings require the toroidal mesh to be contained within the positive quadrant of the axes $(0, 0, \ldots) \rightarrow (n-1, n-1, \ldots)$.

4.1.2 Problem reduction - Collection

In dealing with permutation problems, a major obstacle that must be faced is the problem of contention for links in order to make circuits. Given that we are working with a model with only a constant number of virtual channels, we need to make the best use of those channels. One method is to use each of the virtual channels in each round to perform a task, say transpose a node in C_n (a cycle with n nodes). In Section 4.2 we will show that $\frac{1}{4}$ of the nodes in a cycle want to transpose through the same point on the line of reflection in the same direction. Our first approach allows q nodes of the $\frac{n}{4}$ to transpose in each round, resulting in a total of $\frac{n}{4q}$ rounds to complete the operation. This first approach is used by Fraigniaud and Peters in [6] to implement transpositions and translations with the one-port model. The approach we take in this thesis uses a collection algorithm to reduce the size of the problem until the permutation can be performed trivially on the nodes with collected information. Once these are permuted we reverse the collection (distribution) such that all collected messages are also permuted properly. The difference between using a one-port and all-ports model is discussed at the end of this section.

Figure 4.2: Collection on $\frac{1}{4}$ of C_{200} (2 virtual channels)

In order to reduce the problem by collection in this manner, we rely heavily on the ability to combine messages. Figure 4.2 illustrates how our reduction (collection) takes place on a section of 50 adjacent nodes. When two virtual channels are in use we can collect the information from the two left and right neighbours into the center nodes. In the second round of collection the collector nodes from the first round are collected into second round collector nodes. This procedure continues until the information from the section of the cycle being collected is reduced into at most q nodes. The number of rounds required to collect our example is calculated as $\log_{2q+1}(\frac{n}{4q})$, where $n = 200$. Once collection is complete we are able to perform the desired operation (i.e.

transposition, translation etc.). Finally, we distribute[1] messages to their respective destinations. This is made possible only on permutations that are neighbourhood preserving. We define *neighbourhood preserving* as those operations that map each source node and its neighbours onto a destination node and its neighbours. As we will prove later, each of our operations which use collection are neighbourhood preserving and thus our distribution operation is simply the reverse of the collection operation and takes the same time.

In our example above we calculated the number of rounds based on sections of size $\frac{n}{4}$. In general if we collect M adjacent nodes the number of rounds required is $\log_{2q+1} \frac{M}{q}$. The switching time for each round can be calculated as $q(2q+1)^{r-1}$, where r is the round number (1,2,...). Summing these distances and multiplying by two for the distribution phase results in a total switching cost of $(\sum_{r=1}^{\log_{2q+1}(\frac{M}{q})} q(2q+1)^{r-1}\delta = (\frac{M}{q} - 1)\delta$. The propagation cost is calculated in much the same manner. In each round of collection the packet size is $(2q+1)^{r-1}L$. Summing these and multiplying by the contention q and by two for the distribution phase results in a total propagation cost of $(\frac{M}{q} - 1)L\tau$. Therefore the total cost to collect and distribute messages from M adjacent nodes into at most q nodes results in a cost of:

$$2\log_{2q+1}(\frac{M}{q})\alpha + (\frac{M}{q} - 1)\delta + (\frac{M}{q} - 1)L\tau \tag{4.1}$$

Most of the operations we will be dealing with will require collection on 2- and 3-dimensional toroidal meshes. Collection on a 2-dimensional toroidal mesh, which we picture as a square mesh with wrap-around edges, is done such that we reduce both dimensions at the same time, resulting in the same number of collector nodes being present in each dimension. Collecting into diagonals[2] is one way we can accomplish this purpose. Figure 4.3 illustrates this type of collection with no virtual channels. In the first step each diagonal collects the values of the diagonal on either side of it. In the second step these collected diagonals are again collected, reducing the number of nodes in any row or column to be two. Figure 4.4 shows how using two virtual

[1] We use the terms collect and distribute to avoid confusion with other terms such as gather and scatter which we relate to the general all-to-one and one-to-all operations (Section 3.3).

[2] Collection can be done in either diagonal, only one is shown.

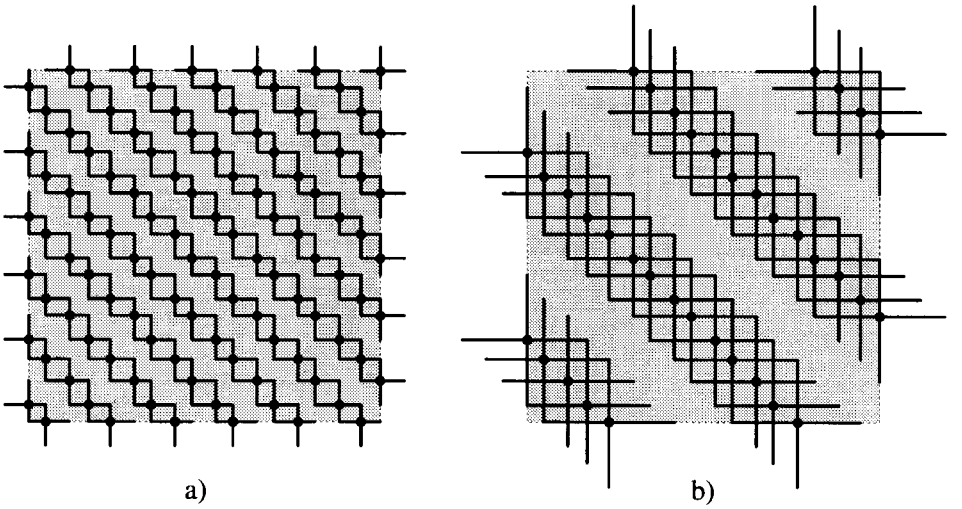


Figure 4.3: Collection pattern on a 2-dimensional torus, with no virtual channels

channels, two diagonals on either side of the collector diagonal are collected. The thick lines collect information from the diagonals next to the collector diagonals, the thin lines collect information from the diagonals a distance of two away from the collector diagonals.

Figure 4.5 shows the collection pattern on a 3-dimensional toroidal mesh, which we illustrate as a cube (again the wrap-around edges are not shown, simplifying the picture). In 3 dimensions we collect into planes perpendicular to the axis[3] between opposite corners. The nodes shown in Figure 4.5 are those which would be considered the collector nodes. The other nodes in the mesh are not shown.

The collection and distribution costs in d dimensions are very similar to those for the 1-dimensional case (Formula 4.1). In 2 dimensions we are collecting M adjacent diagonals into q diagonals and in 3 dimensions we collect M adjacent planes into q planes. Since each dimension is n nodes wide and there are n diagonals in the 2-dimensional case, the number of rounds required to collect using 1- or 2-dimensional collection is the same. This holds true in the d-dimensional case as well. In addition, since we collect adjacent diagonals (or planes) the switching distance in d-dimensions does not change from the distance we calculated earlier for collection within a cycle.

[3]In 3 dimensions we have four different axes along which we can collect.

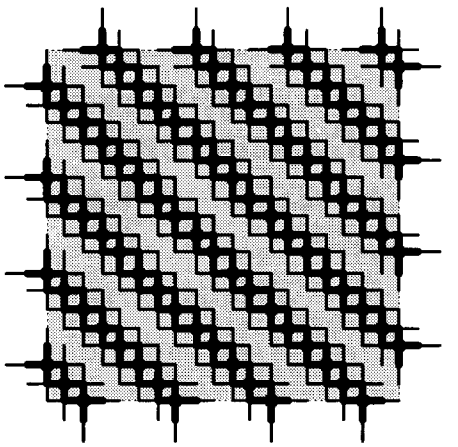


Figure 4.4: Collection pattern on a 2-dimensional torus, with 2 virtual channels

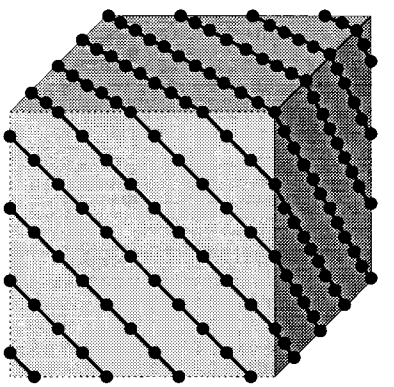


Figure 4.5: Collection pattern on a 3-dimensional torus

Each node on a diagonal (or plane) being collected is connected to d nodes on the collecting diagonal (or plane). Thus it sends only $\frac{1}{d}$ its information along each path reducing the propagation cost by a factor of d. In total the cost for collecting a d-dimensional toroidal mesh is:

$$2\log_{2q+1}(\frac{M}{q})\alpha + (\frac{M}{q} - 1)\delta + (\frac{M}{dq} - \frac{1}{d})L\tau \tag{4.2}$$

In the sections which follow, we show how these collection patterns are used to reduce the size of 2- and 3-dimensional permutation problems down to the point where these operations can be performed in one or two steps.

We note that the difference between the multi-port model and the one-port model with regard to collection is that the one port model will require $2\log_{q+1}(\frac{M}{q})$ rounds and the propagation time will not be divided by the number of dimensions, since each node can only send and receive on a single port. With these changes, each of the algorithms presented in this chapter can be applied to the one-port model as well as to the all-ports model.

4.2 Transposition

4.2.1 1-dimensional Transposition

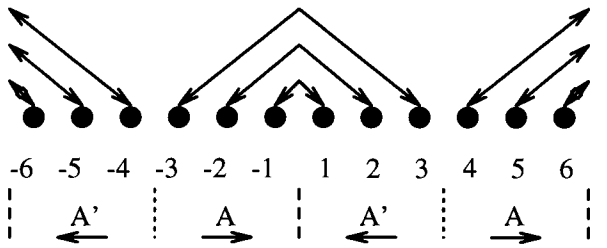


Figure 4.6: 1-dimensional Transposition, $(x) \rightarrow (-x)$

Transposition on a cycle, $(x) \rightarrow (-x)$, is illustrated in Figure 4.6. The maximum distance any message must travel is $\frac{n}{2}$. From the diagram we note that the cycle can be divided into four quadrants (A, A', B, B') where each node in the quadrant sends its message in the same direction.

Performing the transposition in this manner results in a contention of $\frac{n}{4}$. With only q virtual channels, we must reduce the contention to q. Using our standard collection technique, we can collect $2q+1$ nodes into one node in each round. Since our goal is to reduce the contention from $\frac{n}{4}$ to q, we can replace M in Formula 4.1 with $\frac{n}{4}$ to give us the cost of collecting and distributing messages in order to transpose a cycle (Formula 4.3).

$$2\log_{2q+1}(\frac{n}{4q})\alpha + (\frac{n}{4q} - 1)\delta + (\frac{n}{4q} - 1)L\tau \tag{4.3}$$

With only q nodes in each quarter of the cycle, it is then possible to perform the transposition in a single round. Since the same amount of information must pass through the bottleneck, regardless of the collection, the propagation cost is $\frac{n}{4}L\tau$. An upper bound on the switching cost can be determined as the maximum distance, $\frac{n}{2}$, which may exist between the collector nodes. Using these bounds, Formula 4.4 shows the cost of the transposition step and Formula 4.5 shows the total cost of transposing a cycle. This bound on the 1-dimensional transposition will be used in Section 4.4 and its sub-sections, where we combine 1-dimensional transpositions in order to provide solutions for multi-dimensional permutations.

$$\alpha + (\frac{n}{2})\delta + (\frac{n}{4})L\tau \tag{4.4}$$

$$(2\log_{2q+1}(\frac{n}{4q}) + 1)\alpha + (\frac{n}{2} + (\frac{n}{4q} - 1))\delta + (\frac{n}{4} + \frac{n}{4q} - 1)L\tau \tag{4.5}$$

A better bound is possible on the switching term in the transposition step if we can make some further assumptions on what has happens in the collection rounds. During collection, the nodes have been collected a maximum distance of $\frac{\frac{n}{4q}-1}{2}$ (see Formula 4.3). If we assume that the node furthest from the bottleneck (a distance of $\frac{n}{4}$) has been moved the maximum distance, we can say that the furthest collector node is now a distance of $\frac{n}{4} - \frac{\frac{n}{4q}-1}{2}$ away from the bottleneck. During the transposition step then, the furthest distance a node has to travel is now twice this distance or $\frac{n}{2} - (\frac{n}{4q} - 1)$. Due to our assumption earlier, this analysis is only useful when we do not combine transpositions and are able to ensure that the furthest node is collected as stated above. Using the newly calculated cost of the switching term in the transposition step we calculate the total cost of the 1-dimensional transposition to be:

$$(2\log_{2q+1}(\frac{n}{4q}) + 1)\alpha + (\frac{n}{2})\delta + (\frac{n}{4} + \frac{n}{4q} - 1)L\tau \tag{4.6}$$

As was stated earlier, in order for the distribution step to work correctly we need to show that our operations are neighbourhood preserving. From the symmetric labeling associated with transpositions, it can be shown that any vertex i will be transposed onto the vertex $-i$ and that the neighbours of i, ($i+1$ and $i-1$), will be transposed onto the two neighbours of $-i$, ($-i-1$ and $-1+1$). Since this holds for

any i, $(-\frac{n}{2} \leq i \leq \frac{n}{2})$, we know that 1-dimensional transpositions are neighbourhood preserving.

Finally, we present one other algorithm for performing a transposition on a cycle. This algorithm provides a better result than the algorithms already presented but lacks the flexibility which will be required of the 1-dimensional transposition algorithm in the following sections. The basic concept of the algorithm is that it combines the collection/distribution and transposition phases. Our previous algorithms only sent information through the bottleneck during the transposition phase. In this algorithm at the same time we are collecting and distributing information, we will transpose information as well. Due to the overlapping of phases in this algorithm, it is only useful for doing a single 1-dimensional transposition. Since a main focus of this thesis is to show how 1-dimensional transpositions can be combined we refer the reader to Appendix A for a complete discussion of the algorithm. The cost of performing the 1-dimensional transposition by this algorithm is:

$$(2\log_{2q+1}(\frac{n+4}{4q+4})+1)\alpha + (c(\frac{q+4}{2q+2})(n+4))\delta + (\frac{n}{4})L\tau \tag{4.7}$$

4.2.2 2-dimensional Transposition

There are two 2-dimensional transpositions, one in both of the main diagonals resulting in permutations of $(x,y) \rightarrow (-y,-x)$ and $(x,y) \rightarrow (y,x)$. Each node exchanges its message with its reflection. The line of reflection for the $(x,y) \rightarrow (-y,-x)$ permutations is shown in Figure 4.7 a) as the dashed line. Due to the wrap-around nature of the torus the line of reflection appears along the main diagonal and along the outer diagonals. Using the cycles depicted in Figure 4.7 b) we are able to connect each node to its reflection with a minimal distance path. The dotted line in a) is a line which is not crossed by any minimal distance paths and which divides the nodes into four sections A, A', B, and B'. Each node in each section sends its message in the same direction.

The 2-dimensional transposition can be performed using n 1-dimensional transpositions in parallel on the n cycles shown in Figure 4.7 b). In order to reduce the cost of this operation we associate half the nodes (nodes of the same colour according to the

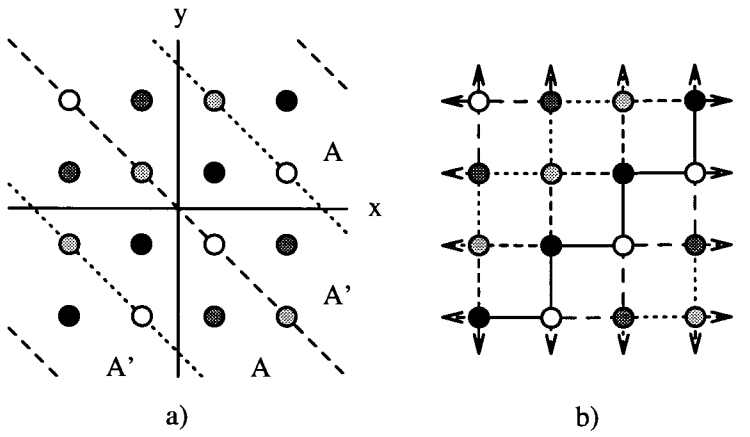


Figure 4.7: 2-dimensional Transposition, $(x, y) \rightarrow (-y, -x)$

diagram) on each cycle to only one of the cycles. This reduces the number of nodes on each cycle from $2n$ to n but does not reduce the length of each cycle. Collection can occur along these cycles but collection using the standard 2-dimensional collection scheme (Formula 4.2, collection into diagonals) has a lower propagation cost since messages can be divided in two and sent. The amount of collection required using either method is $\frac{n}{4}$ nodes or diagonals collected into q nodes or diagonals. Using $M = \frac{n}{4}$ we find the cost to collect and distribute using the 2-dimensional scheme to be:

$$2\log_{2q+1}(\frac{n}{4q})\alpha + (\frac{n}{4q} - 1)\delta + (\frac{n}{8q} - \frac{1}{2})L\tau \tag{4.8}$$

Once the information has been collected into q nodes within each section the transposition step can take place along the cycles shown in Figure 4.7 b). Since the distance between nodes is double that of the 1-dimensional case the cost is essentially the same as in the 1-dimensional case (Formula 4.4), with the difference that the switching cost has increased to n. The other 2-dimensional transposition, $(x, y) \rightarrow (y, x)$, can be performed by collecting and transposing with respect to the diagonal, $y = x$. The total cost for either 2-dimensional transposition is:

$$(2\log_{2q+1}(\frac{n}{4q}) + 1)\alpha + (n + (\frac{n}{4q}) - 1))\delta + (\frac{n}{4} + \frac{n}{8q} - \frac{1}{2})L\tau \tag{4.9}$$

Again we need to prove that this operation is neighbourhood preserving for our distribution to work. Since the two 2-dimensional transpositions simply exchanges

cartesian axes or exchanges and negates the axes, it is easily shown that the two operations are neighbourhood preserving by choosing any point and showing its neighbours are preserved through the transposition. Knowing that the 1- and 2-dimensional transpositions are neighbourhood preserving allows us to make the statement that all the global permutations (Section 4.4) are neighbourhood preserving since each can be constructed as a combination of these two operations.

In the same way we modified the 1-dimensional transposition in Formula 4.6, we can arrange collection on the 2-dimensional toroidal mesh in order to reduce the switching term to the diameter without affecting the other factors (α and τ). An even better solution in terms of the number of rounds and propagation cost uses the 1-dimensional transposition algorithm described in Appendix A. Using this algorithm, the cost of the 2-dimensional transposition is calculated using n parallel 1-dimensional transpositions and is shown in Formula 4.10. The difference in cost between the 1- and 2-dimensional transpositions using the algorithm from Appendix A is simply a doubling in the switching factor since the distance between nodes is now doubled.

$$(2\log_{2q+1}(\frac{n+4}{4q+4})+1)\alpha+(2c(\frac{q+4}{2q+2})(n+4))\delta+(\frac{n}{4})L\tau \tag{4.10}$$

4.3 Translation

Translations are the second major method we use in this thesis to perform permutations. Unlike transpositions for which there is only one form, $(x) \rightarrow (-x)$, translation distances vary and have the form, $(x) \rightarrow (x+i)$. A translation on a cycle is the same as a rotation on a cycle where the angle of rotation is expressed as a distance.

In this section the d-dimensional translations we study are of the form, $(x, y, \ldots) \rightarrow (x+i, y+i, \ldots)$. A translation of different distances in different dimensions can be performed as a combination of translations (see Section 4.5.1).

We use the same general algorithm for translations as was described for transpositions. First, we collect messages using our standard collection scheme until the contention is low enough such that we can perform a simple translation with contention q. After the translation we can then distribute the messages to their respective locations.

In order for this operation to work, translations must be neighbourhood preserving operations. This can be trivially shown, since performing a global translation adds the same value to all variables thus maintaining their relative positions.

We analyze translations in two categories in the following subsections; $i = \frac{n}{2}$ and $i < \frac{n}{2}$.

4.3.1 Translations of $i = \frac{n}{2}$

The easier of the two cases to discuss is the $\frac{n}{2}$ case, since each node has exactly two paths of the same distance in either direction along which it can send its message. When neighbouring nodes send in opposite directions, we reduce the contention to $\frac{n}{4}$ in both direction (as shown in Figure 4.8) for a propagation cost of $\frac{n}{4}L\tau$. Trying to send half the message out both ports results in the same propagation delay and in the worst case (all virtual channels in use) doubles the number of rounds. For this reason sending single messages out one port is preferred. The cost of this operation is again based on collection and distribution plus the actual translation time itself. Since we have only q virtual channels the amount of collection required reduces messages from $\frac{n}{4}$ nodes into q nodes (see Formula 4.1). Adding the cost of the translation step $(1\alpha + \frac{n}{2}\delta + \frac{n}{4}L\tau)$, we find the total cost for a translation of distance $\frac{n}{2}$ on a cycle is:

$$(2\log_{2q+1}(\frac{n}{4q}) + 1)\alpha + (\frac{n}{2} + \frac{n}{4q} - 1)\delta + (\frac{n}{4} + \frac{n}{4q} - 1)L\tau \qquad (4.11)$$

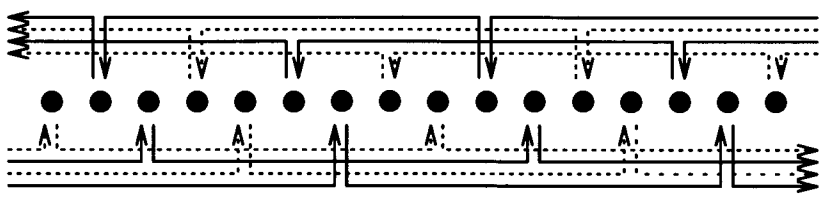


Figure 4.8: Translation on C_{16} ($i = \frac{n}{2}$), uses edges in both directions

The complexity of translating on the d-dimensional toroidal mesh is not much different than on the cycle since we implement the translation on cycles of length dn.

We construct our cycles in d dimensions such that two nodes are connected in a cycle if and only if they are separated by a value of plus one in all dimensions. On the 3-dimensional toroidal mesh each cycle would be of the form: $\{(x,y,z),(x+1,y+1,z+1),(x+2,y+2,z+2),\ldots,(x+(n-1),y+(n-1),z+(n-1))\}$. Between any two nodes on one of these cycles there exists d ($d=3$ in this example) disjoint paths connecting them. An example of such a path which connects (x,y,z) to $(x+1,y+1,z+1)$ is $(x,y,z) \rightarrow (x+1,y,z) \rightarrow (x+1,y+1,z) \rightarrow (x+1,y+1,z+1)$. If we choose to use all d connecting paths we can establish d disjoint ways of connecting the same cycle. This at first appears good, as it would allow us to divide our messages and send them along disjoint cycles. However, each of the edges on the chosen paths are also used by $d-1$ other cycles in the mesh, resulting in a contention of d if we were to use all connecting paths. It turns out to be less expensive if we choose only a single path to connect each cycle. In this manner every edge in the mesh is used in only one cycle resulting in no contention between translations on different cycles. Figure 4.7 b) shows an example of these cycles on a 2-dimensional toroidal mesh. Choosing each cycle to only work with the four nodes of the same colour (see figure) results in the best solution we found.

Using our standard multi-dimensional collection procedure (Formula 4.2) and our simple 1-dimensional translation (with the modification of a distance d between nodes), we present the cost of d-dimensional global translations where i=$\frac{n}{2}$ as:

$$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha+(\frac{dn}{2}+\frac{n}{4q}-1)\delta+(\frac{n}{4}+\frac{n}{d4q}-\frac{1}{d})L\tau \qquad (4.12)$$

4.3.2 Translations of i < $\frac{n}{2}$

A 1-dimensional translation of this variety can be thought of as any rotation around the cycle other than 180°. Translations of distances greater than $\frac{n}{2}$ are simply translations of distances less than $\frac{n}{2}$ in the other direction.

Multi-dimensional translations of this variety are not rotations. The relationship between 1-dimensional and d-dimensional translations which held for translations of $\frac{n}{2}$ hold as well for these translations. Thus, The switching term of the translation step increases by a factor of d and the propagation cost in the collection and distribution

phases is reduced by a factor of d.

Translations of $x \rightarrow x + 1$ have no contention and are the simplest to perform. Translations where the number of virtual channels (q) is greater than or equal to the translation distance (i) are also straightforward to perform. The cost of this type of operation is shown below:

$$\alpha + di\delta + iL\tau \tag{4.13}$$

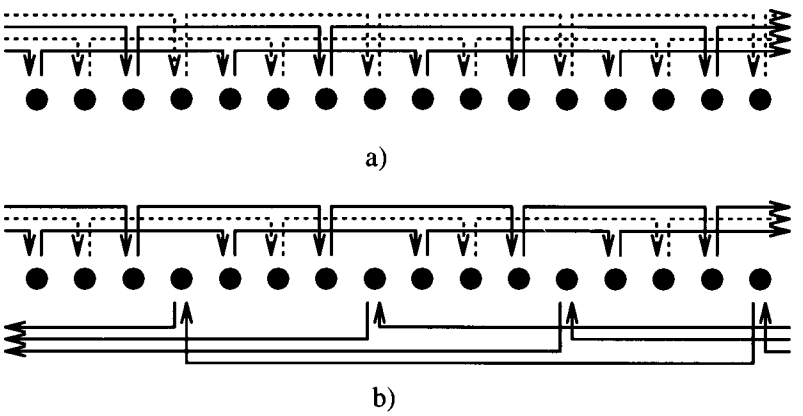


Figure 4.9: Translation on C_{16} ($i = 4$), a) one direction, b) both directions

If the number of virtual channels (q) is less than the translation distance (i), collection and distribution phases are added to the translation. Collection takes place on sections of the cycles of size i and continues until the i nodes are reduced into at most q collector nodes. The cost of doing this can be calculated using Formula 4.1. Once the collection is complete, we can translate the collector nodes according to Figure 4.9. Figure 4.9 a) depicts this type of translation using edges in only one direction. Formula 4.14 represents the cost of this operation where $i = \frac{n}{a}$.

$$(2\log_{2q+1}(\frac{n}{aq}) + 1)\alpha + (d\frac{n}{a} + \frac{n}{aq} - 1)\delta + (\frac{n}{a} + \frac{n}{daq} - \frac{1}{d})L\tau \tag{4.14}$$

In order to make better use of edges without increasing the number of virtual channels we select a fraction of the nodes to perform their translations backwards. This method increases the switching term while decreasing the other two cost terms. If

we represent the translation distance i as $\frac{n}{a}$, then $\frac{(a-1)n}{a}$ nodes will continue to translate their messages forward while $\frac{n}{a}$ nodes will translate their messages backwards. The reduction in the number of rounds and propagation cost is indicated in Formula 4.15 by the factor $\frac{a-1}{a}$. The increase in the propagation cost is represented by the factor (a-1). Figure 4.9 b) depicts a translation of $\frac{n}{4}$ where $\frac{3}{4}$ of the information continues forward while $\frac{1}{4}$ can be translated on the backward edges. The following formula gives the cost for this type of movement.

$$(2\log_{2q+1}(\frac{(a-1)n}{a^2q})+1)\alpha+(\frac{d(a-1)n}{a}+\frac{(a-1)n}{a^2q}-1)\delta+(\frac{(a-1)n}{a^2}+\frac{(a-1)n}{da^2q}-\frac{1}{d})L\tau \tag{4.15}$$

4.4 Global Permutations

With the algorithms and cost equations for the 1- and 2-dimensional transpositions and the 1-dimensional translations, we are able to present the algorithms and cost equations for all global permutations. Each global permutation can be represented as a permutation of the cartesian axes.

Table 4.1: Global Permutations, 1-dimensional

Operation	Permutations
Identity	(x)
1D Transposition	$(-x)$

Table 4.2: Global Permutations, 2-dimensional

Operation	Permutations
Identity	(x,y)
1D Transposition	$(-x,y),(x,-y)$
2D Transposition	$(y,x),(-y,-x)$
2D 90° Rotation	$(y,-x),(-y,x)$
2D 180° Rotation	$(-x,-y)$

For the cycle there exist only two operations, the identity and the transposition (or reflection). These are listed in Table 4.1. A 2-dimensional toroidal mesh has eight

permutations (four choices for the first axis ($\pm x and \pm y$), and two for the second). Table 4.2 shows the permutations on a 2-dimensional torus and the operation associated with each. In some cases there are more ways than one to perform the operation. We explore some alternatives and present the best algorithm found. An example of this is the 180° 2-dimensional rotation which can be performed as either a rotation (Section 4.4.2) or as a reflection through the mid-point of the toroidal mesh (Section 4.4.1).

Table 4.3: Global Permutations, 3-dimensional

Operation	Permutations
Identity	(x, y, z)
1D Transposition	$(-x, y, z), (x, -y, z), (x, y, -z)$
2D Transposition	$(x, z, y), (y, x, z), (z, y, x),$ $(x, -z, -y), (-y, -x, z), (-z, y, -x)$
2D 90° Rotation*	$(x, -z, y), (-y, x, z), (-z, y, x),$ $(x, z, -y), (y, -x, z), (z, y, -x)$
2D 180° Rotation	$(-x, -y, z), (x, -y, -z), (-x, y, -z)$
3D Mid-point Reflection	$(-x, -y, -z)$
3D 120° Rotation*	$(y, z, x), (z, x, y), (-y, z, -x), (-z, -x, y),$ $(y, -z, -x), (-z, x, -y), (-y, -z, x), (z, -x, -y)$
3D 180° Rotation	$(-x, z, y), (y, x, -z), (z, -y, x),$ $(-z, -y, -x), (-y, -x, -z), (-x, -z, -y)$
3D Transposition (type 1)	$(-z, -x, -y), (-y, -z, -x), (z, x, -y), (y, -z, x),$ $(z, -x, y), (-y, z, x), (-z, x, y), (y, z, -x)$
3D Transposition (type 2)	$(-x, -z, y), (-x, z, -y), (-z, -y, x),$ $(z, -y, -x), (-y, x, -z), (y, -x, -z)$

Following the pattern above we present the 48 (6*4*2) permutations on the 3-dimensional toroidal mesh in Table 4.3. Again the permutations are broken down into respective operations by which they can be performed.

Each operation listed in the three permutation tables can be performed using a combination of 1 and 2-dimensional transpositions. In some cases we find the cost of combining these operations expensive and turn to using translations as an alternative method which provides better results. A summary of the results of this section is given in Section 4.7. The two * operations in the table above are those which can be

performed more efficiently using translational rather than transpositional methods.

In each of the following subsections, we describe the algorithm by which the transformation can be performed and the cost of performing it, the results of these are tabulated in Table 4.4.

4.4.1 Reflections through the mid-point

A reflection through the mid-point results in only one transformation. That transformation being $(x, y, \ldots) \rightarrow (-x, -y, \ldots)$. In one dimension this transformation is the same as the 1-dimensional transposition. In 2 dimensions it is the same as the 180° rotation discussed in the next section.

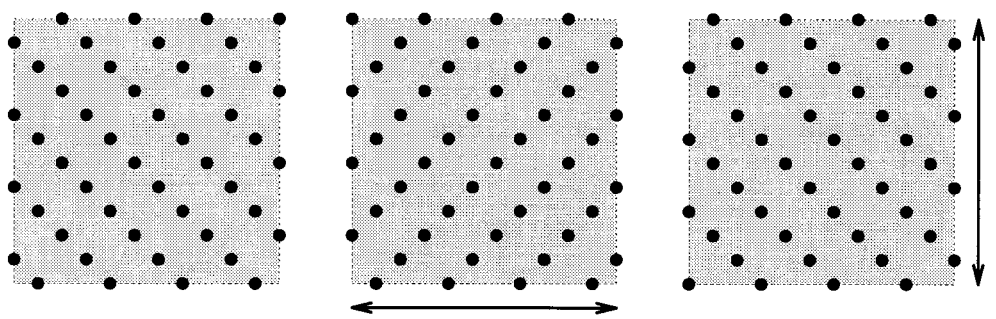


Figure 4.10: 2-dimensional Reflection through the mid-point

This transformation can be created by doing a 1-dimensional transposition in each dimension of a d-dimensional torus. In the case of 2 dimensions (as in Figure 4.10) we can perform it by transposing dimension x followed by transposing dimension y, at double the cost of a single transposition.

If we perform the standard d-dimensional collection we find that there is no need to distribute and re-collect messages in between transposition steps since the initial collection reduced each dimension the same amount such that further reductions are unnecessary. Therefore we perform a standard d-dimensional collection at the beginning followed by the d transposition steps and then a d-dimensional distribution phase.

An improvement on this algorithm notices that the transposition steps are all one-to-one operations and can be combined into one large one-to-one step. This holds since each transposition is in a different dimension and thus the edges used are all disjoint. By transitivity then we can join the one-to-one functions creating one large one-to-one function.

Combining d transposition steps into one requires us to sum the switching costs but not the propagation costs since the bandwidth demands are the same in each but the distances messages travel is cumulative. With this information and the cost of the single transposition in Formula 4.5 we find the cost of the combined transposition step to be:

$$\alpha + (d\frac{n}{2})\delta + (\frac{n}{4})L\tau \tag{4.16}$$

Adding to this the cost of the d-dimensional collection and distribution phases (Formula 4.2) gives us a total cost for this operation in d-dimensions. Reflection through the mid-point is the only permutation we analyze on d-dimensions. The cost of this operation is given below:

$$(2\log_{2q+1}(\frac{n}{4q}) + 1)\alpha + (d\frac{n}{2} + \frac{n}{4q} - 1)\delta + (\frac{n}{4} + \frac{n}{d4q} - \frac{1}{d})L\tau \tag{4.17}$$

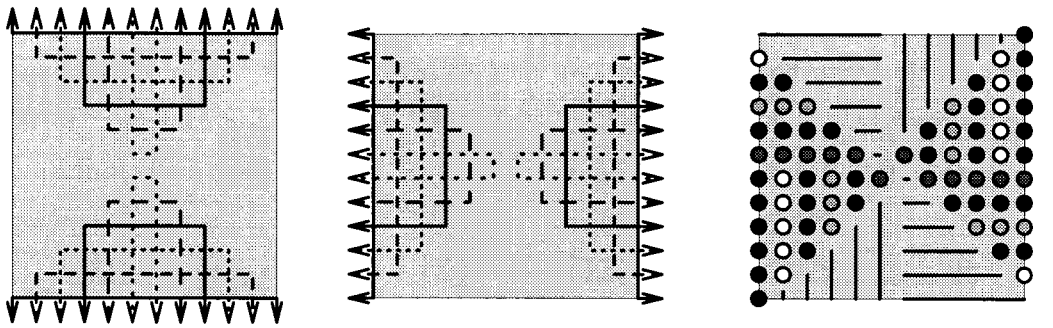
4.4.2 2-dimensional Rotations

180° Rotation

The 2-dimensional 180° rotation as noted in the previous section is the same as a reflection through the mid-point $(x, y) \rightarrow (-x, -y)$. However, it can also be performed using a translation technique at a small increase in the cost. This translational algorithm is presented to provide a comparison between the two techniques and shows that the transpositional method is better due to its more regular pattern of collection.

Figure 4.11 a) depicts a set of twelve cycles, all of length $2n$, which can allow us to perform this transformation by translation. The cycles are divided between two pictures to make it easier to see the pattern.

In order to perform a translation on these cycles we cannot use the standard collection pattern on the 2-dimensional torus, rather we collect messages along the



a) b)

Figure 4.11: a) Cycles used in the 2D 180° Rotation, b) partitioning

cycles indicated using a 1-dimensional collection pattern on each cycle. Since each node in the 2-dimensional torus is on two of these edge disjoint cycles, we partition the nodes such that they will participate on only one of the cycles they appear on. In this manner we can reduce the number of active nodes on each cycle from $2n$ to n. Figure 4.11 b) illustrates how the nodes on the cycles can be partitioned. The lines indicate the general pattern that is followed to divide the nodes while the coloured points (one colour per cycle) show specifically which nodes are active on each cycle.

At this point we can perform a translation of distance $\frac{N}{2}$, where N is $2n$ (the length of each cycle). Since the number of active nodes on each cycle is only n, we obtain the same translation cost as is given in Formula 4.11, except that the translation distance is n and the switching cost is increased for collection and distribution since nodes are not adjacent. We approximate the switching cost for collection and distribution to be twice the standard cost since the average distance between nodes is doubled. Compared to the result given in the previous section, this result (Formula 4.18) is more expensive due to a less regular collection pattern being employed. For this reason we use the cost of the transpositional algorithm (2-dimensional reflection through the mid-point) for this operation in the summary table (Table 4.4).

$$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha+(n+\frac{n}{2q}-2)\delta+(\frac{n}{4}+\frac{n}{4q}-1)L\tau \qquad (4.18)$$

90° Rotation

Like the 180° rotation, the 90° rotation can also be performed using either transpositional or translational methods. It differs in that it cannot be performed by two 1-dimensional transpositions but by one 2-dimensional and one 1-dimensional transposition.

$$+90° = (x,y) \rightarrow (y,x) \rightarrow (y,-x) \text{ or } (x,y) \rightarrow (-y,-x) \rightarrow (y,-x)$$
$$-90° = (x,y) \rightarrow (y,x) \rightarrow (-y,x) \text{ or } (x,y) \rightarrow (-y,-x) \rightarrow (-y,x)$$

Since one of the dimensions (x or y) is used in both transpositions, the two transpositions cannot be combined as they were in Section 4.4.1. Rather they must be performed one after the other. It is possible to perform the operation using a 1-dimensional transposition followed by a 2-dimensional transposition. In the examples above we have shown the 2-dimensional transposition being done first.

Since both operations require the same amount of collection no additional distribution or collection steps are required between the two transposition steps. The total cost of this operation is easily calculated as the sum of Formulas 4.4 and 4.9 resulting in:

$$(2\log_{2q+1}(\frac{n}{4q}) + 2)\alpha + (\frac{3n}{2})\delta + (\frac{n}{2} + \frac{n}{8q} - \frac{1}{2})L\tau \tag{4.19}$$

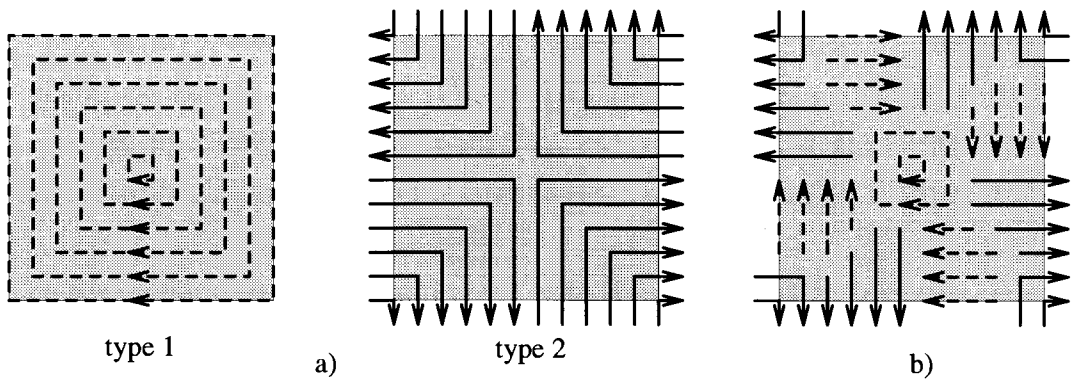


Figure 4.12: a) Cycles used in the 2D 90° Rotation b) partitioning

We can obtain better results by using a translational method which requires only a single step in between collection and distribution. Figure 4.12 a) shows two sets

of edge disjoint cycles which we use to perform this operation. Unlike the 180° case where all cycles were of the same length, here we deal with the problem of cycles of different lengths ($4 \leq N \leq 4n$).

Since each node appears on two cycles we can again divide the nodes such that only half the nodes on each cycle are active. The 90° rotation is accomplished by performing a translation of $\frac{N}{4}$ on each of the cycles, where N is the length of each cycle. The cost of this operation when only forward edges are used in the translation is:

$$(2\log_{2q+1}(\frac{n}{2q}) + 1)\alpha + (n - 1 + \frac{n}{2q} - 1)\delta + (\frac{n}{2} + \frac{n}{2q} - 1)L\tau \tag{4.20}$$

When we include the backward edges we can obtain a cost of:

$$(2\log_{2q+1}(\frac{3n}{8q}) + 1)\alpha + (3(n - 1) + \frac{3n}{8q} - 1)\delta + (\frac{3n}{8} + \frac{3n}{8q} - 1)L\tau \tag{4.21}$$

Again, we can do better than this. The methods described above divide the number of nodes evenly on all cycles. If we divide them unevenly we are able to reduce contention and reduce the cost of the operation. When translations are considered in the forward direction only, the contention in a translation of $\frac{N}{4}$ is the same as the number of active nodes on any quarter of the cycle. The cycles in the methods above have contentions ranging from 0 to $\frac{n}{2}$. Balancing the contention results in increasing the number of nodes which are active on the short cycles and reducing the number of nodes active on the longer cycles. It is not possible to evenly balance the contention since some cycles are too short. Figure 4.12 b) illustrates a possible partitioning of the nodes which takes into account this balancing. Using these short cycles it can be proved by induction that the maximum contention can be reduced from $\frac{n}{2}$ to $(\lfloor\frac{n}{2}\rfloor - \lfloor\frac{n+2}{6}\rfloor) \approx \frac{n-1}{3} \approx \frac{n}{3}$.

The new maximum contention, $\frac{n}{3}$, applies to translations using edges in the forward direction only. If we use backward edges as well we are able to reduce the contention to $\frac{n}{4}$ (since $\frac{3}{4} * \frac{n}{3} = \frac{n}{4}$). Since our cycles do not fit the standard translation algorithms we adapt the algorithms to come up with a cost equation. From the pattern shown in Figure 4.12 b) we see that the nodes to be collected are all adjacent and so we can use the standard collection formula (Formula eq:collectM) to determine the cost

where $M = \frac{n}{4}$. The cost of the translation step is also straightforward. A single round is required to perform the translation and the switching cost is $3(n-1)\delta$ (the length of the longest edge used). The propagation cost is determined using the maximum contention and is found to be $\frac{n}{4}L\tau$. In total the cost of this operation using our balancing technique is:

$$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha + (3n + \frac{n}{4q} - 4)\delta + (\frac{n}{4} + \frac{n}{4q} - 1)L\tau \qquad (4.22)$$

4.4.3 3-dimensional Rotations

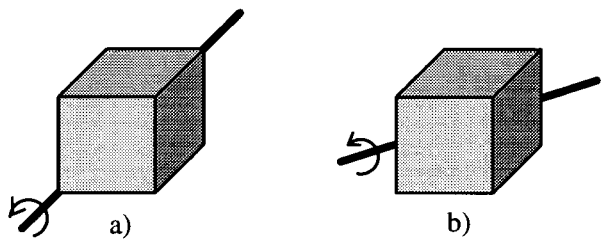


Figure 4.13: 3-dimensional Rotations, a) 120°, b) 180°

120° Rotation

The axis of rotation of a 120° 3-dimensional rotation is shown is Figure 4.13 a). There are four such axes between opposite corners and two rotations $\pm 120°$ leading to the eight permutations listed in Table 4.3.

As with the 90° 2-dimensional rotation discussed earlier, the 120° rotation can also be performed using two transpositions but a better method uses translations. It appears from these problems that the transpositional method works best if no dimension is used more than once. When one dimension is required to be used more than once it becomes a 2-phase transposition which nearly doubles the propagation cost.

The 120° rotation can be performed as a combination of two 2-dimensional transpositions, where one of the dimensions is used in both. In order to calculate the total cost (Formula 4.23) we sum the cost of collecting and distributing on a 3-dimensional

toroidal mesh and the cost of performing two 2-dimensional transposition steps.

$$(2\log_{2q+1}(\frac{n}{4q}) + 2)\alpha + (2n)\delta + (\frac{n}{2} + \frac{n}{12q} - \frac{1}{3})L\tau \tag{4.23}$$

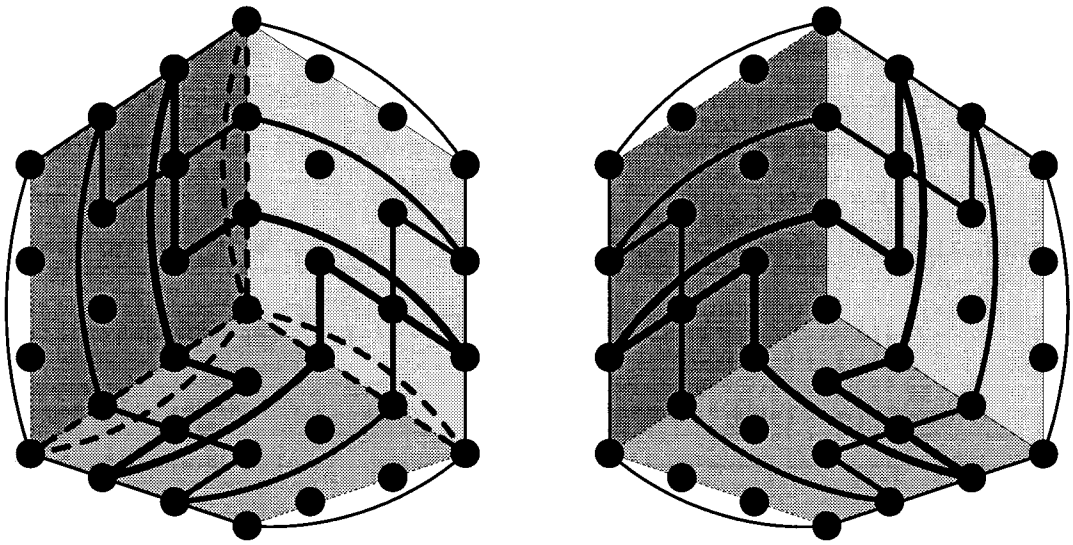


Figure 4.14: Cycles used in a 3-dimensional 120° Rotation

Figure 4.14 depicts the pattern of cycles used in order to perform the rotation by translation. The cycles are broken into two pictures so that their pattern is more easily recognizable. The cycles can be used either clockwise for a 120° rotation or counter-clockwise for a $-120°$ rotation. The edges in the diagrams which are curved are those which require the use of the wrap-around links in order to make the connection. The rotation is performed simultaneously on the front and back three faces of the cubic mesh around the axis of rotation.

As shown in the diagram these cycles connect nodes appearing on the surface of the cubic mesh. Similar cycles are established on the surface of each cube recursively contained within the outer cube. The cycles on the surface of the outermost cube have a length of $3n$, since the three groups of (n-1) links on the surface of the cube are connected by three wrap-around edges of length one. On the largest cube within the outermost cube the cycle lengths are still $3n$, since the three groups of (n-3) links on the surface of that cube are connected with three wrap-around edges of length three.

The wrap-around edges of length three pass through two nodes on the outermost surface which simply route the messages through. In fact, for each cube recursively held within the outermost cube the cycle lengths are $3n$. In addition, on all cycles, the distance between source and destination nodes is n.

From the pattern in Figure 4.14 we also find that each node is a member of two cycles. Translating each node on only one of the cycles allows us to reduce the contention in the translation from n to $\frac{n}{2}$. Using backward edges in the translation allows us to reduce the translation contention to $\frac{n}{3}$, since two-thirds of the nodes will continue to route their messages forwards, $\frac{2}{3}\frac{n}{2} = \frac{n}{3}$. Using Formula 4.15 where $\frac{n}{a} = \frac{n}{3}$ and $3n$ as the length of the cycles, we compute the cost of this operation to be:

$$(2\log_{2q+1}(\frac{n}{3q}) + 1)\alpha + 2(n + \frac{n}{3q} - 1)\delta + (\frac{n}{3} + \frac{n}{3q} - 1)L\tau \qquad (4.24)$$

Again this result is better than the one produced by the transpositional method. We note that the switching cost is multiplied by two since the nodes on the cycles being collected are not necessarily adjacent.

180° Rotation

The axis of rotation for this operation is shown in Figure 4.13 b). The axis runs between opposite edge mid-points. There are six such axes leading to the six different permutations listed in Table 4.3.

This transformation can be performed as a combination of a 2-dimensional and a 1-dimensional transposition (e.g. $(x, y, z) \rightarrow (-y, -x, z) \rightarrow (-y, -x, -z)$). Since the transpositions can be chosen such that different axes are used in both we can combine the two transposition steps into one step. In addition, since the collection pattern is the same for the 1- and 2-dimensional transpositions we can perform this operation with a single collection and distribution phase. The cost of this transformation is shown below.

$$(2\log_{2q+1}(\frac{n}{4q}) + 1)\alpha + (\frac{3n}{2} + \frac{n}{4q} - 1)\delta + (\frac{n}{4} + \frac{n}{12q} - \frac{1}{3})L\tau \qquad (4.25)$$

4.4.4 3-dimensional Transpositions

3-dimensional Transposition (type 1)

The 3-dimensional transposition is best understood as a transformation of the form $(x, y, z) \rightarrow (-y, -z, -x)$ or $(x, y, z) \rightarrow (-z, -x, -y)$. Figure 4.15 a) and b) graphically depict these two transformations on one face of the cube. Table 4.3 lists the eight different permutations which fall into this category.

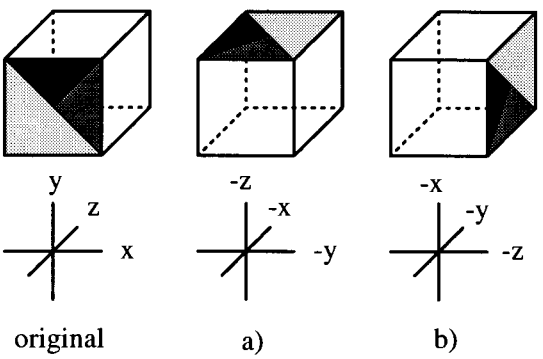


Figure 4.15: 3-dimensional Transposition type 1, (one face shown)

This operation can be performed by a 2-dimensional 90° rotation followed by a 2-dimensional transposition. This provides us with the correct result but has two draw-backs. The first is that it requires a single dimension to be used twice (i.e. the middle steps must be performed sequentially). The second is that the best solution for the 90° rotation required a special collection procedure which does not match that required by the 2-dimensional transposition and thus a second collection step would be required as well.

Using other combination choices we can eliminate the second problem but not the first. For example we could use a combination of a 3-dimensional 180° rotation[4] followed by a 2-dimensional transposition. Here, two dimensions are used twice but since we have to use at least one twice there is no additional cost for using two. Combining these two operations one after the other using our standard collection

[4]Which is itself a 2-dimensional transposition followed by a 1-dimensional transposition.

procedures results in the following cost:

$$(2\log_{2q+1}(\frac{n}{4q})+2)\alpha+(\frac{5n}{2}+\frac{n}{4q}-1)\delta+(\frac{n}{2}+\frac{n}{12q}-\frac{1}{3})L\tau \qquad (4.26)$$

In an attempt to come up with a better solution using translational methods, we identified a set of cycles which appeared[5] to perform the transformation. The number of rounds required was comparable to that given above while the propagation cost was approximately three times that above. Due to the added cost of performing this operation by this method and the complexity of the cycles we have not included diagrams or descriptions of these cycles. Another possible solution we looked at involved modifying the 90° rotation such that it could be used in conjunction with the 2-dimensional transposition in this special case. Again this solution required approximately the same number of rounds and increased the propagation cost by a factor of $\frac{3}{2}$.

In the end we were unable to identify a better solution than the one presented in Formula 4.26. In order to be able to perform global permutations efficiently using translations it appears that the length of the cycles found must be relatively short. In the case of 3-dimensional rotations the cycles can be organized to circle the axis of rotation (maximum length $3n$). In this operation the cycles found connected nodes on each face of the cube resulting in cycles of length at least $6n$.

3-dimensional Transposition (type 2)

The final global operation appears to be very similar to the 3-dimensional transposition (type 1) and for lack of a better name it is referred to here as the 3-dimensional transposition (type 2). Figure 4.16 shows the effect of this operation on a single face of a 3-dimensional toroidal mesh, under two different permutations a) $(x,y,z) \rightarrow (z,-y,-x)$ and b) $(x,y,z) \rightarrow (-z,-y,x)$. It can be constructed by a 3-dimensional 180° rotation followed by a 1-dimensional transposition (rather than a 2-dimensional transposition in type 1). Table 4.3 lists the six permutations which fall into this category.

[5]The cycles were only tested on a subset of 3-dimensional toroidal meshes.

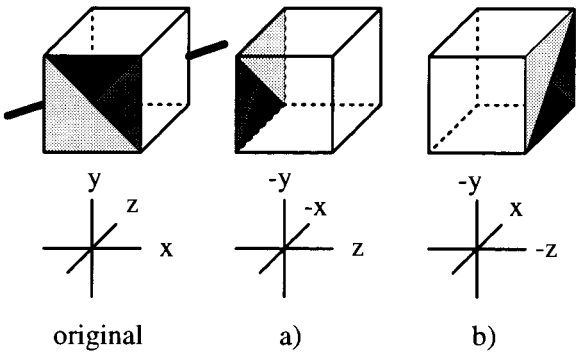


Figure 4.16: 3-dimensional Transposition type 2, (one face shown)

Like the 3-dimensional transposition (type 1), this operation requires two steps since it requires one dimension to be used more than once. Again standard collection patterns can be used since we are only dealing with 1 and 2-dimensional transpositions. The cost of this operation is:

$$(2\log_{2q+1}(\frac{n}{4q}) + 2)\alpha + (2n)\delta + (\frac{n}{2} + \frac{n}{12q} - \frac{1}{3})L\tau \tag{4.27}$$

It would be ideal for this operation to use a 2-dimensional 90° rotation followed by a 1-dimensional transposition since this would only require a single middle step. As was pointed out earlier however, the solution for the 2-dimensional 90° rotation requires non-standard collection, and cannot be combined with other operations which require use of the standard collection scheme.

A translational algorithm was not attempted for this solution due to its similarities with the previous operation where we were unable to find a better solution than the transpositional solution presented.

A summary all the global permutations and their costs is provided in Table 4.4 (Section 4.7).

4.5 Other Translation-based Permutations

There exist many other permutations which do not involve a permutation on the cartesian axes. The most common of these are discussed in this section. Each of

these operations is performed using translations only.

4.5.1 Combination of Translations

The most common application of combining translations involves performing a translation of different distances in different dimensions. Translations of this nature can be performed in at most d steps, where d is the number of dimensions. We can perform a translation on each dimension sequentially or we can perform multi-dimensional translations on a decreasing number of dimensions. An example of both methods is shown below:

$$\text{Example: } (x, y, z) \rightarrow (x + 3, y + 9, z + 7)$$

$$\text{Method 1: dimension by dimension}$$

$$(x, y, z) \rightarrow (x + 3, y, z) \rightarrow (x + 3, y + 9, z) \rightarrow (x + 3, y + 9, z + 7)$$

$$\text{Method 2: multi-dimensional translations}$$

$$(x, y, z) \rightarrow (x + 3, y + 3, z + 3) \rightarrow (x + 3, y + 7, z + 7) \rightarrow (x + 3, y + 9, z + 7)$$

The cost by either method involves summing the cost of the three individual translations. In Section 4.3 it was shown that the cost of performing multi-dimensional translations in terms of the α and τ terms is essentially the same as for a translation in a single dimension. With this knowledge it is clear that combining translations of different sizes is most efficiently handled by the second method. In the best case the two methods result in the same cost, in the worst case Method 1 costs d times that of Method 2 (in terms of α and τ).

4.5.2 2-dimensional Shear

Shear is an example in which translations of different sizes can be used together to produce a wanted effect. Figure 4.17 shows the effect of one type of shear, namely $(x, y) \rightarrow (x + y, y)$. We restrict our attention in this section to 2-dimensional shears. It is possible however to extend this discussion to shears of higher dimension.

More general shears can also be performed, for example $(x, y) \rightarrow (x + cy, y)$, where c is some constant. To measure the cost of these operations we consider which

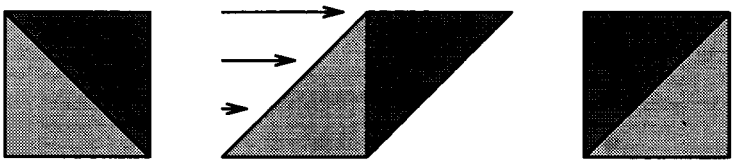


Figure 4.17: 2-dimensional Shear

translation has the highest cost for each of the three factors. Each row of the mesh is translated a different distance from 0 to $\frac{n}{2}$ (translations of over $\frac{n}{2}$ are the same as a translation of less than $\frac{n}{2}$). Using Formula 4.15 where $(1 \leq \frac{n}{a} \leq \frac{n}{2})$ we find that the maximum for each factor results in the following equation.

$$(2\log_{2q+1}(\frac{n}{4q}) + 1)\alpha + (n + \frac{n}{4q} - 2)\delta + (\frac{n}{4} + \frac{n}{4q} - 1)L\tau \tag{4.28}$$

4.5.3 Other 2-dimensional Rotations

There are many other 2-dimensional rotations, in addition to 90°, 180° and 270°. The problem with these (e.g. 30°, 45°) rotations is that the 'one-to-one' correspondence does not necessarily hold. For example, the rotation θ could be modeled as $(x, y) \rightarrow (x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta)$. Either we find that two nodes map to the same destination or if we start from the destinations we can find a source node which does not map to a destination node. In both cases we fail in our attempts to make the problem 'one-to-one'.

Another form of rotation, which is not perfect but is 'one-to-one', specifies the paths of rotation and then causes the rotation to occur within those specific paths. For example, we can assign each node to be a member of a cycle which has a specific radius from the center. Figure 4.18 illustrates one set of cycles which allow this type of rotation. The dotted lines in the figure show how the diagonal connections can be made. Using translations of θr (where r is radius and cycle number) on cycles 0–5, we are able to perform a basic rotation. Depending on the angle of rotation cycles 6–7 (corner cycles) could be ignored or rotated as well. The algorithm for translating cycles 6–7 is more complicated since the nodes on the cycles are partially reversed.

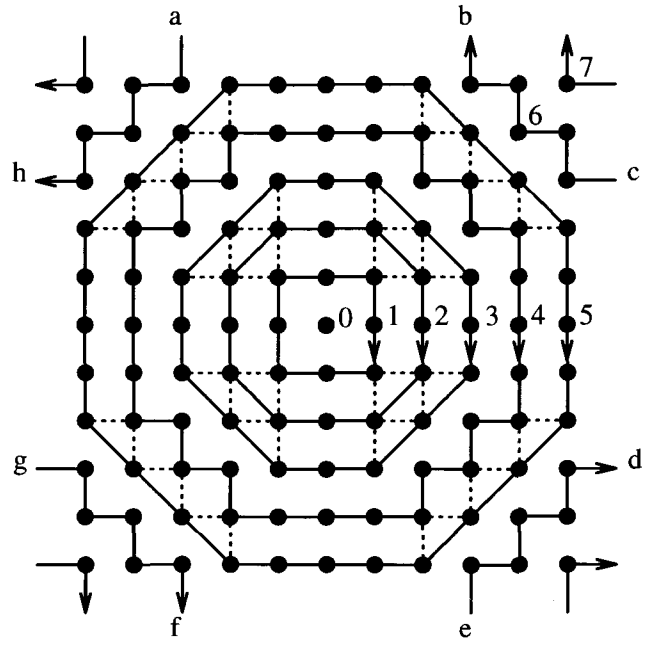


Figure 4.18: 2-dimensional Rotation with cycles based on radius

For the nodes to be in 'normal' order we would expect the cycle to be connected in the following manner: a to b, b to c, ..., h to a. Instead they are connected as shown in the diagram since there are no remaining edges in the expected direction on which to run the cycle.

The cost of this rotation (excluding the corner cycles) can be calculated as the cost of performing the translation on the longest cycle using either Formula 4.14 (forward edges only) or Formula 4.15 (edges in both directions).

4.6 Lower bounds

In order to show the efficiency of our permutation algorithms, in this section we present lower bounds on 1-dimensional transpositions, d-dimensional translations of distance $\frac{n}{2}$ and d-dimensional global permutations where pairs of nodes exchange messages[6]. As was the case with the dissemination algorithms we derive the lower bounds as the

[6]Mid-point Reflections and 180° Rotations are two permutations which fit into this last category.

maximum of the lower bounds based on three independent terms (α, δ, and τ).

For each of the operations mentioned above the maximum distance between an originating node and its destination node is the diameter of the network. Therefore, a lower bound on the switching cost for each operations is $d\frac{n}{2}$. The lower bound on the number of rounds follows from the work done with dissemination problems in the previous chapter. In those problems we looked at the total number of nodes that needed to receive a message from the originator and then determined the number of rounds based on the number of ports available. Since we have one-to-one pairing, a lower bound on the number of rounds is 1.

In the 1-dimensional transposition and d-dimensional global permutation described above each node exchanges messages with its reflection. For the transposition, the line of reflection cuts the cycle at two points dividing the cycle into two equal parts. In order to transpose the cycle, all the information from one half must pass through one of the two dividing points. This gives us a lower bound on the propagation cost of: $\frac{\frac{n}{2}}{2}L\tau = \frac{n}{4}L\tau$. In the d-dimensional global permutation we again have a reflection where the network is divided into two equal parts and each node exchanges messages with its reflection in the other part. In order to disconnect the network into two equal parts $2n^{d-1}$ links must be removed (2 in 1 dimension, $2n$ in 2 dimensions etc.). Since the parts are equal we find that half the messages in the network must pass through the removed $2n^{d-1}$ links resulting in a lower bound on the propagation cost of $\frac{n^d}{2}(\frac{1}{2n^{d-1}})L\tau = \frac{n}{4}L\tau$.

In order to calculate the translation propagation cost for distances of $\frac{n}{2}$ we refer to the method we applied to multi-scattering in Section 3.1. There we derived the lower bound on the propagation cost as the communication capacity required divided by the bandwidth available. Since the diameter,$d\frac{n}{2}$, is the distance each message (n^d in total) is translated, the communication capacity required is $n^d d\frac{n}{2}L$. The total bandwidth available is $2dn^d\frac{1}{\tau}$ (the total number of links, both directions). Dividing the communication capacity required by the total bandwidth gives us a lower bound on the propagation cost of $\frac{n}{4}L\tau$ for all d-dimensional translations of distance $\frac{n}{2}$.

For each of the three types of operations described in this section the lower bound is the same and is given as: $\max\{\alpha, d\frac{n}{2}\delta, \frac{n}{4}L\tau\}$.

Using our model the lower bound of one on the number of rounds is not achievable with a constant number of virtual channels. Since we are limited in the number of circuits that can be established we hypothesize that a better lower bound on the number of rounds when combined with the propagation cost would be logarithmic. This because in a single round we can at most cause a propagation cost of q (where for all but trivial problems $q < \frac{n}{4}$) and the lower bound on the propagation cost is $\frac{n}{4}$.

4.7 Summary

In this chapter we have presented algorithms and cost equations for all global 1, 2, and 3-dimensional permutations and a collection of other translation-based permutations. A list of the cost equations for each of these algorithms is given in Table 4.4[7].

Table 4.4: Permutation Communications Times on Toroidal meshes

1D Transpose	$(2\log_{2q+1}(\frac{n+4}{4q+4})+1)\alpha + (c(\frac{q+4}{2q+2})(n+4))\delta + (\frac{n}{4})L\tau$
2D Transpose	$(2\log_{2q+1}(\frac{n+4}{4q+4})+1)\alpha + (2c(\frac{q+4}{2q+2})(n+4))\delta + (\frac{n}{4})L\tau$
2D 90° Rotation*	$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha + (3n+\frac{n}{4q}-4)\delta + (\frac{n}{4}+\frac{n}{4q}-1)L\tau$
2D 180° Rotation	$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha + (n+\frac{n}{4q}-1)\delta + (\frac{n}{4}+\frac{n}{8q}-\frac{1}{2})L\tau$
3D Mid-point Reflection	$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha + (\frac{3n}{2}+\frac{n}{4q}-1)\delta + (\frac{n}{4}+\frac{n}{12q}-\frac{1}{3})L\tau$
3D 120° Rotation*	$(2\log_{2q+1}(\frac{n}{3q})+1)\alpha + (2n+\frac{2n}{3q}-2)\delta + (\frac{n}{3}+\frac{n}{3q}-1)L\tau$
3D 180° Rotation	$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha + (\frac{3n}{2}+\frac{n}{4q}-1)\delta + (\frac{n}{4}+\frac{n}{12q}-\frac{1}{3})L\tau$
3D Transpose (type 1)	$(2\log_{2q+1}(\frac{n}{4q})+2)\alpha + (\frac{5n}{2}+\frac{n}{4q}-1)\delta + (\frac{n}{2}+\frac{n}{12q}-\frac{1}{3})L\tau$
3D Transpose (type 2)	$(2\log_{2q+1}(\frac{n}{4q})+2)\alpha + (2n+\frac{n}{4q}-1)\delta + (\frac{n}{2}+\frac{n}{12q}-\frac{1}{3})L\tau$
dD Translation ($i=\frac{n}{2}$)	$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha + (\frac{dn}{2}+\frac{n}{4q}-1)\delta + (\frac{n}{4}+\frac{n}{d4q}-\frac{1}{d})L\tau$
2D Shear	$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha + (n+\frac{n}{4q}-2)\delta + (\frac{n}{4}+\frac{n}{4q}-1)L\tau$

Since each operation in the table is a product of 1-dimensional transpositions (excluding translation and shear) we would expect the resulting costs to appear close

[7]To simplify the presentation and analysis the ceiling and floor functions have been dropped.

to that of the d-dimensional mid-point reflection.

$$(2\log_{2q+1}(\frac{n}{4q})+1)\alpha+(d\frac{n}{2}-(\frac{n}{4q}-1))\delta+(\frac{n}{4}+\frac{n}{4dq}-\frac{1}{d})L\tau \qquad (4.29)$$

In most cases this is correct. Two exceptions to the rule are the 1-dimensional and 2-dimensional transpositions which can be performed by an algorithm which is more efficient than the general algorithm (see Appendix A). In these operations both the number of rounds and propagation costs are reduced at the expense of the switching term. Since the switching term is the least expensive term, an added cost there is generally acceptable.

The two * operations are those for which a translational algorithm was found which is better than our standard transpositional algorithms. In these two operations and in the two 3-dimensional transpositions we found that the operations were sufficiently complex (each operation required using edges in some dimension more than once) such that the 1-dimensional and 2-dimensional transpositions could not be combined. In the case of the two * operations we were able to overcome this by finding paths on which a translation could be performed to accomplish the rotation in a single step. In the case of the 3-dimensional transpositions no such translation paths were found with better results, resulting in a sequential ordering of transpositions and therefore a doubling in the propagation costs.

The costs of the d-dimensional translations of distance $\frac{n}{2}$ and the 2-dimensional shear are given in the table as representatives of those operations using translations. Many permutations are based on translations which exhibit the same cost patterns as transpositions. Translations are useful for modeling many permutations which are not possible to model using transpositions.

In our presentation we have tried to minimize the number of rounds while maintaining the propagation cost within a small factor, $\frac{q+1}{q}$, of the lower bound. In most of our algorithms we have succeed in this goal provided that a cumulative lower bound using both the α and τ results in the number of rounds being logarithmic instead of 1. In addition for most algorithms the cost of the switching term is close to the lower bound, $d\frac{n}{2}$.

Chapter 5

Conclusions

In this thesis we have been able to provide algorithms for performing each of the basic problems in the information dissemination classification and in the information permutation classification. These algorithms have been modeled on 1-, 2- and 3-dimensional toroidal meshes using circuit-switched routing.

In the dissemination classification we were able to show the benefits of the multiple port model over the single port model. The benefit of the multi-port model was most pronounced in the broadcasting and scattering algorithms where the number of rounds was decreased by a factor of $\log_{q+1}(2q+1)$ and the propagation time was decreased by a factor of $2d$ and $1.58d$ for scattering and broadcasting respectively.

Gossiping and multi-scattering presented different problems and we hypothesized that the reductions were not as good as the 'one-to-all' operations due to the increased competition and usage of links. The algorithms for the 'all-to-all' operations provided no improvement in the number of rounds and decreased the propagation cost by factors smaller than their 'one-to-all' counterparts. From our study of these problems we hypothesize that the additional capabilities of multiple ports are best applied to problems which have bandwidth open for use as was the case in the 'one-to-all' problems.

The operations in the information permutation classification presented several interesting results. First, it was noted that the lower bounds on several single and multi-dimensional permutations worked out to the same values. In addition each of

the global permutations were easily identifiable as a permutation on the axes and as such could be performed as a product of 1- and 2-dimensional transpositions. In order to provide efficient solutions we presented general algorithms for these two transpositions which allowed the transpositions to be combined provided the permutations occurred on different axes. In cases where transpositions could not be combined efficiently, it was shown in some instances that the global permutations could be performed efficiently using parallel translations instead. Other permutations such as the 2-dimensional shear and 2-dimensional rotations (of degrees other than $\pm 90°$ and $180°$) were also shown to be possible and efficient using translations.

A more efficient algorithm (in the case where transpositions are not combined) for the 1- and 2-dimensional transpositions was also presented and analyzed. This algorithm was able to improve upon the number of rounds and reduced the propagation cost to the lower bound.

In general we were able to provide permutation algorithms at a cost near the lower bound (within a factor of $\frac{q+1}{q}$) with respect to the switching and propagation terms. We hypothesize that a logarithmic number of rounds is also good but this was not shown since our lower bounds treated each cost factor independently.

Appendix A

1-dimensional Transposition

A better approach to the transposition problem allows messages in each round to pass through the bottleneck rather than collecting all the messages and sending them through at the same time. We show that by using this approach the propagation cost is reduced to the lower bound.

The method of collection on the cycle is much the same as described for the standard case with the exception that at each step there are message bundles which are sent through the bottleneck rather than being collected. Figure A.1 depicts the movement of data in each round for one half of a cycle in one direction using two virtual channels (transposes quadrant I of the cycle into quadrant II).

In the first round since we have 2 (q in the general case) virtual channels, the two nodes closest to the bottleneck can be moved across the bottle neck and the remainder of the nodes in each quadrant can be collected into groups of 5 ($2q+1$ in the general case). In the second round the two groups of 5 closest to the bottleneck can be moved across the bottleneck and the remainder can be collected into groups of 25. In the third round since there are no more nodes to collect we are able to move the groups of 25 over the bottleneck. In the fourth round we can start distributing the groups of 25 (as groups of 5) and move the remaining 2 groups of 5 over the bottleneck. In the fifth and final round we unpack all the groups of 5 and move the final 2 messages over the bottleneck.

In total the algorithm takes $2\log_{2q+1}(\frac{n+4}{4q+4}) + 1$ rounds. We derive this number

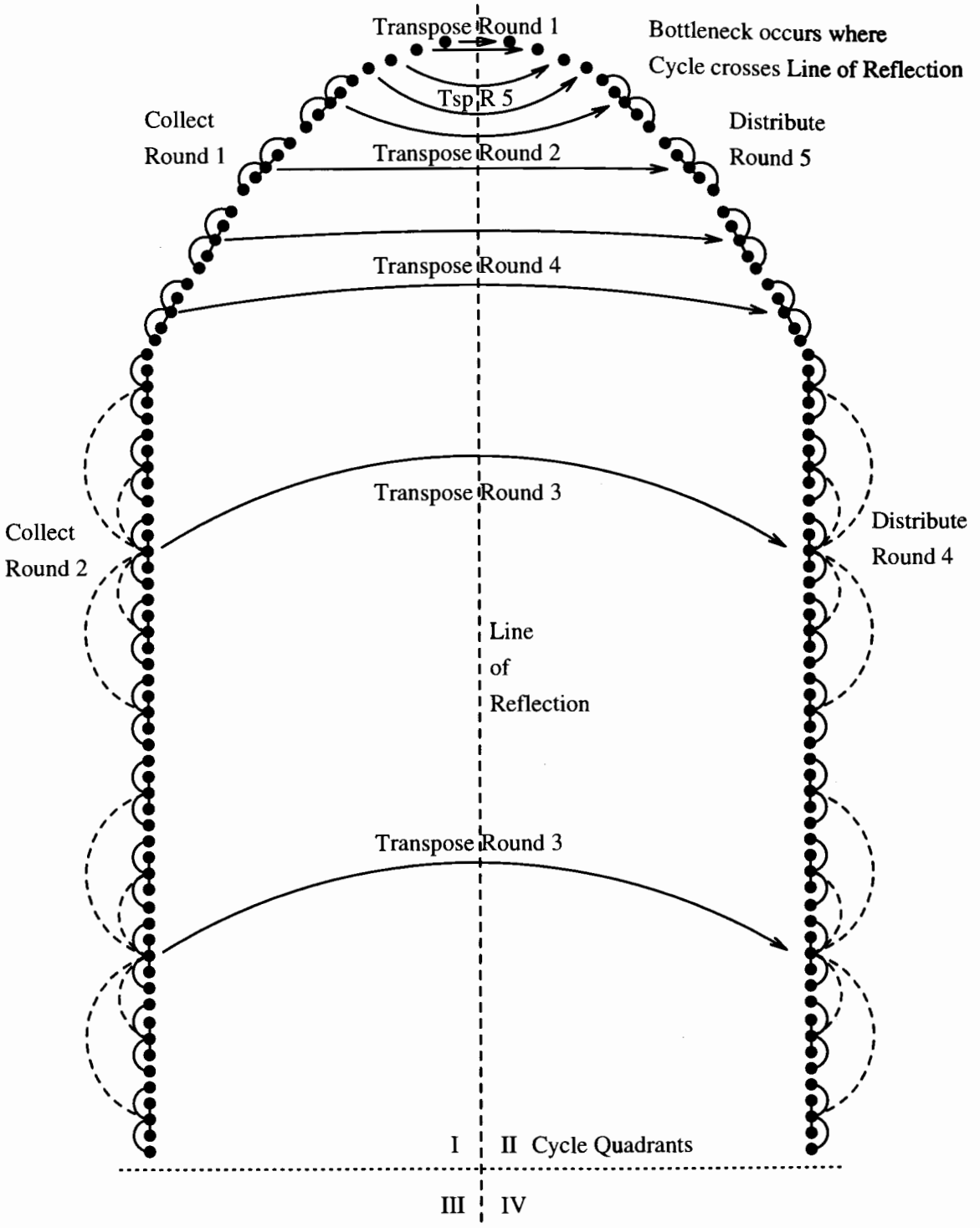


Figure A.1: 1-dimensional Transposition, (combined phases algorithm)

from the following expression, where r is the number of rounds required to collect the information and $2r + 1$ is the total number of rounds required to perform the transposition.

$$\frac{n}{4} = 2q\sum_{i=0}^{r-1}(2q+1)^i + q(2q+1)^r \tag{A.1}$$

The propagation cost, p, is also fairly easy to compute and turns out to be $\frac{n}{4}$. It is computed from the following expression:

$$p = 2q\sum_{i=0}^{r-1}(2q+1)^i + q(2q+1)^r \tag{A.2}$$

The switching cost, s, does not simplify as nicely. It is the sum of the maximum distances messages are switched in each round. Since the transposition distances are always greater than the collection or distribution distances we sum the transposition distances. The following equation is used to calculate the switching time.

$$s = (2q-1)(2q+1)^r + (10q-2)\sum_{i=0}^{r-1}(2q+1)^i + 8q\sum_{j=0}^{r-2}\sum_{i=0}^{j}(2q+1)^i \tag{A.3}$$

The equation above can be reduced to a closed form (below), which is still far too complicated. We approximate the switching time to be $c(\frac{q+4}{2q+2})(n+4)$, where $c \geq 1$. When q is relatively large we obtain results close to the lower bound $\frac{n}{2}$. When q is small we obtain results closer to n or a multiple of n.

$$s = (\frac{q+4}{2q+2})(n+4) + (\frac{n+4}{2q^2+2q})(\frac{3}{2} + \log_{2q+1}(\frac{n+4}{4q+4})) - \frac{2(2q+1)^2}{q} - 5 + \frac{1}{q} \tag{A.4}$$

In total, the cost of transposing a cycle by this method is:

$$(2\log_{2q+1}(\frac{n+4}{4q+4}) + 1)\alpha + (c(\frac{q+4}{2q+2})(n+4))\delta + (\frac{n}{4})L\tau \tag{A.5}$$

Bibliography

[1] J-C. Bermond, P. Michallon and D. Trystram, *Broadcasting in wraparound meshes with parallel monodirectional links*, Rapport de recherche n° 91-30, LMC-IMAG, Grenoble, France, 1991

[2] C. Calvin and D. Trystram, *Matrix transposition on usual networks*, manuscript, June 1993

[3] P. Fraigniaud, *Communications intensives dans les architectures à mémoire distribuée et algorithmes parallèles pour recherche de racines de polynômes*, Ecole Normale Supérieure de Lyon, 1990

[4] P. Fraigniaud, *Complexity Analysis of Broadcasting in Hypercubes with Restricted Communication Capabilities*, Journal of Parallel and Distributed Computing, 16, pp. 15–26, 1992

[5] P. Fraigniaud and E. Lazard, *Methods and problems of communication in usual networks*, Research Report 91-33, Lab. de l'Informatique du Parallelisme, Ecole Normale Superieure de lyon, France, 1991, to appear in Discrete Applied Mathematics, 1994

[6] P. Fraigniaud and J. Peters, *Structured communication in torus networks*, manuscript, 1993

[7] S.M. Hedetniemi, S.T. Hedetniemi, A.L. Liestman, *A survey of gossiping and broadcasting in communication networks*, Networks, vol. 18, pp. 319–349, 1986

[8] C-T. Ho, *Matrix transpose on meshes with wormhole and XY routing*, IBM Research Division, Technical Report RJ 9385 (82637), June 1993

[9] J. Hromkovič, R. Klasing, B. Monien and R. Peine, *Dissemination of information in interconnection networks (broadcasting and gossiping)*, to appear in: F. Hsu, D.-Z. Du (eds.), Combinational Network Theory, Science Press & AMS

[10] S.L. Johnsson, *Communication efficient basic linear algebra computations on hypercube architectures*, Journal of Parallel and Distributed Computing, 4:133–172, 1987

[11] D.W. Krumme, G. Cybenko, and K.N. Venkataraman, *Gossiping in minimal time*, SIAM Journal of Computing, Vol 21 No. 1, pp. 111–139, Feb. 1992

[12] Z. Li, F. Tong and R. Laughlin, *Parallel algorithms for line detections on a 1xN array processor*, CSS-IS TR 90-06, Centre for Systems Science, Simon Fraser University, July 1990

[13] J-Y.L. Park, S-K. Lee, and H-A. Choi, *New algorithms for broadcasting in meshes*, Technical Report GWU-IIST-93-03, George Washington University, 1993

[14] J. Peters and M. Syska, *Circuit-switched broadcasting in torus networks*, Technical Report CMPT TR 93-04, School of Computing Science, Simon Fraser University, May 1993

[15] B. Plateau and D. Trystram, *Optimal total exchange for a 3-D torus of processors*, Information Processing Letters, 42 (1992), pp. 95–102

[16] Y. Saad, M.H. Schultz, *Data communication in parallel architectures*, Parallel Computing, vol. 11, pp. 131–150, 1989

[17] D. Scott, *Efficient all-to-all communication patterns in hypercube and mesh topologies*, 6th Distributed Memory Computing Conference, IEEE Computer Society Press, pp. 398–403, 1991

[18] S.R. Seidel, *Broadcasting on linear arrays and meshes*, Report ORNL/TM-12356, Oak Ridge National Laboratory, March 1993