All-port Total Exchange in Cartesian Product Networks

Vassilios V. Dimakopoulos

Technical Report 28-2002

Depatrment of Computer Science, University of Ioannina
P.O. Box 1186, Ioannina, Greece GR-45110.
Tel: +30-26510-98809, Fax: +30-26510-98890,
E-mail: dimako@cs.uoi.gr

December 2002

Abstract

We present a general solution to the total exchange communication problem for any homogeneous multidimensional network under the *all-port* assumption. More specifically, we consider cartesian product networks where every dimension is the same graph (e.g. hypercubes, square meshes, n-ary d-cubes) and where each node is able to communicate simultaneously with all its neighbors. We show that if we are given an algorithm for a single n-node dimension which requires T steps, we can construct an algorithm for d-dimensions and running time of $n^{d-1}T$ steps, which is provably optimal for many popular topologies. Our scheme, in effect, generalizes the total exchange algorithm given by Bertsekas et al [1] for the hypercubes and complements our theory [6] for the single-port model.

1. Introduction

Distributed-memory multiprocessors (as for example the NCube, Cray T3D, Intel Paragon, MIT Alewife among many) have a node interconnection structure that is mostly based on cartesian product (or multidimensional) networks such as hypercubes, meshes and tori. Multidimensional networks have been used and studied extensively because of their rich graph-theoretic properties which are directly derived from the properties of their constituent dimensions. Because of their symmetric nature *homogeneous* multidimensional networks traditionally receive most of the attention. Such a network has all of its dimensions identical, yielding thus a high degree of regularity. Hypercubes, square meshes, n-ary d-cubes are a few examples of homogeneous multidimensional networks.

As long as nodes communicate through an interconnection network and no common memory is physically available, information dissemination is a prevailing issue. Apart from the need for a pair of nodes to exchange information (usually termed 'unicast' communication), information dissemination includes collective communications where a multitude of nodes are involved. The frequent appearance of such communication patterns especially in parallel numerical algebra [9, 2] demands efficient communication algorithms and has spawned an increasing amount of research.

A general survey on collective communication problems was given in [8], including single/multinode broadcasting, scattering, gathering and total exchange. *Total exchange* is the subject of this work and is known to be the most demanding of all the aforementioned problems. In total exchange, which is also known as *multiscattering* or *all-to-all personalized communication*, each node in a network has a distinct message to send to every other node. Thus, in an n-node network every node scatters $N-1$ messages destined to each of the other nodes. Various data permutations occurring e.g. in parallel FFT and basic linear algebra algorithms can be viewed as instances of the total exchange problem [2].

Solutions to the total exchange problem have been mainly ad hoc, meaning that they apply only to particular topologies, usually hypercubes or two-dimensional tori/meshes, and even not for arbitrary network sizes (see e.g. [12, 10, 1, 17] for packet-switched networks, [4, 16, 11, 14, 13, 15] for circuit switched/wormhole-routed

networks).

In a previous work [6] we provided a structured solution applicable to *any* multidimensional network which is packet-switched and adheres to the single-port model. However, under the *all-port model* where simultaneous communication will all neighbors is allowed, an algorithm was only given for two-dimensional homogeneous networks. Here we complete the theory by presenting a general algorithm that works for *any number of dimensions*. In particular, we show that if there exists a total exchange algorithm for an n-node graph H that takes T_H time units, then an algorithm for graph H^d can be constructed that takes time $n^{d-1}T_H$. Hypercubes and n-ary d-cubes are just two well known examples of homogeneous networks to which our method applies.

Under the model we follow we only know of few optimal algorithms, the ones developed in [1, 7] for hypercubes and the optimal or near-optimal ones for n-ary d-cubes in [17]. In fact, our work can be viewed as a generalization of the work of Bertsekas et al [1] to arbitrary homogeneous multidimensional networks. Although the algorithm is not tied to any particular switching scheme, as in [1, 6] we will deal with packet-switched networks under the constant model [8] where:

- communication links are bidirectional and fully duplex
- a message requires one time unit (or *step*) to be transferred between two nodes
- only adjacent nodes can exchange messages.

The *all-port* capability will also be assumed where a node may send/receive messages to/from all its neighbors in a single step. Notice that the packet-switched model is only assumed for the purposes of the analysis. The derived algorithm works without modifications for any switching scheme, e.g. circuit switching or wormhole routing. However, in such cases it may not achieve strict optimality.

The rest of the paper is organized as follows. After presenting multidimensional networks and some of their properties in Section 1.1, we review the algorithm given by Bertsekas et al [1] for the hypercubes in Section 2. In the same section we give the basic idea behind our generalization. The algorithm, along with its correctness proof, is described in detail in Section 3. Section 4 gives the conditions under which the proposed algorithm behaves optimally. Finally, Section 5 concludes the paper.

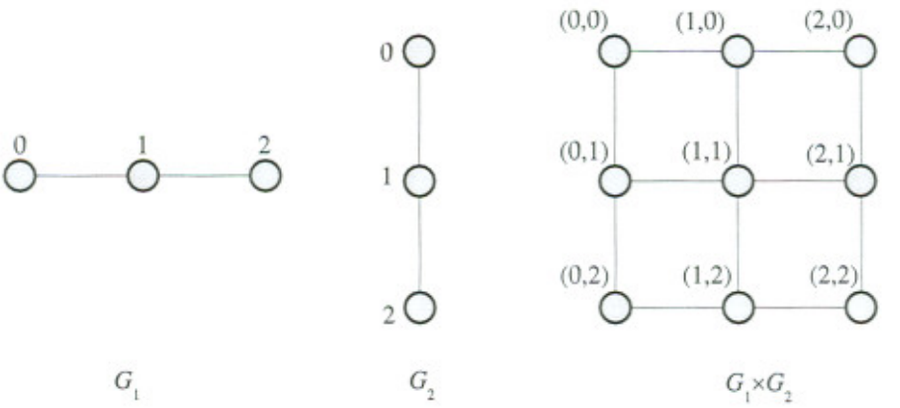

Figure 1. *A homogeneous two-dimensional mesh*

1.1. Multidimensional networks

Let $G = (V, E)$ be an undirected graph[1] with node (or vertex) set V and edge (or link) set E. This is the usual model of representing a multiprocessor interconnection network: processors correspond to nodes and communication links correspond to edges in the graph. The nodes in G are $|V|$ in number, labeled as $0, 1, \ldots, |V| - 1$. An edge in E between nodes i and j is written as the unordered pair ij and i and j are said to be *adjacent* to each other, or just *neighbors*.

Given d graphs $G_k = (V_k, E_k)$, $k = 1, 2, \ldots, d$, their (cartesian) product is defined as the graph $G = G_1 \times \cdots \times G_d = (V, E)$ whose vertices are labeled by a d-tuple $(i_1, \ldots, i_d)$ and

$$V = \Big\{(i_1, \ldots, i_d) \mid 0 \leq i_k \leq |V_k| - 1, k = 1, \ldots, d\Big\}.$$

Nodes $(i_1, \ldots, i_d)$ and $(j_1, \ldots, j_d))$ are adjacent if and only if

$$\exists k \text{ s.t. } i_k j_k \in E_k \text{ and } i_l = j_l \text{ for all } l \neq k.$$

We will call such products of graphs *multidimensional* graphs and G_k will be called the kth *dimension* of the product. The kth component of the address tuple of a node will be called the kth *address digit* or the kth *coordinate*. The definition of E above in simple words states that two nodes are adjacent if they differ in exactly one

[1]The terms 'graph' and 'network' are considered synonymous here.

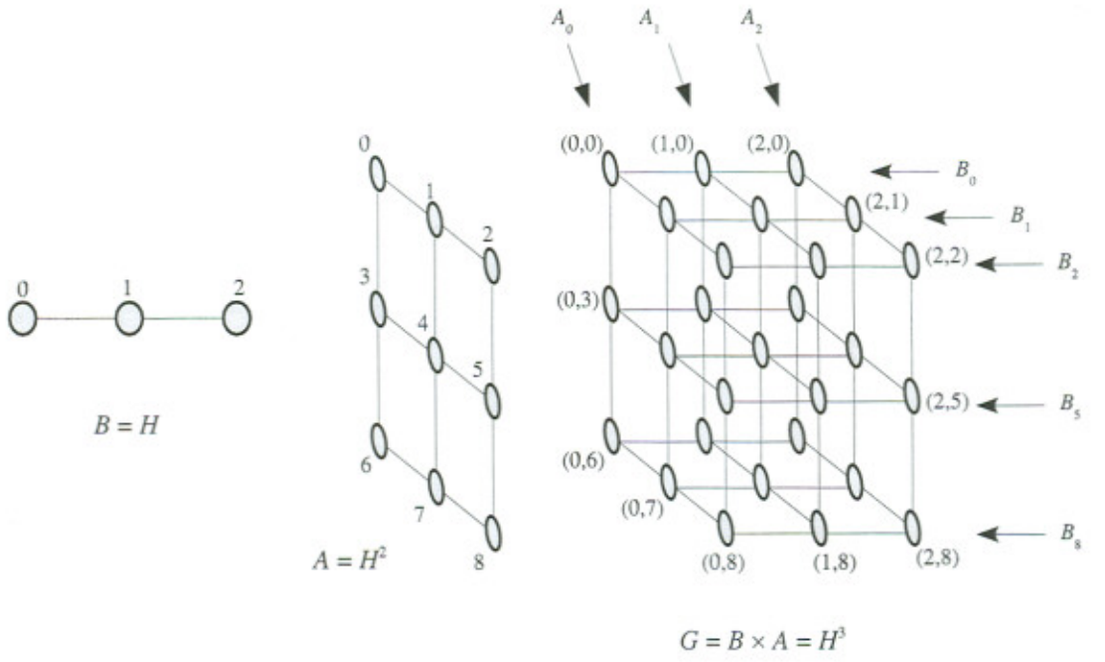

Figure 2. *A homogeneous three-dimensional mesh*

address digit. Their differing coordinates should be adjacent in the corresponding dimension. An example is given in Fig. 1.

If all dimensions are identical, that is, $G_k = H$, $k = 1, 2, \ldots, d$, then the network is characterized as *homogeneous*, and we will denote it at $G = H^d$. Hypercubes are homogeneous products of two-node linear arrays (or rings). n-ary d-cubes are homogeneous d-dimensional products of n-node rings. Homogeneous generalized hypercubes are products of K_n, the n-node complete graph, [3]. The network in Fig. 1 is a homogeneous two-dimensional mesh.

Multidimensional graphs have $|V_1||V_2|\cdots|V_d|$ nodes, where $|V_k|$ is the number of nodes in G_k, $k = 1, 2, \ldots, d$. We will assume that graph H has n nodes; consequently the homogeneous network $G = H^d$ will have n^d nodes.

Graph $G = H^d$ can be defined equivalently as the product of two graphs $G = B \times A$, where $A = H^{d-1}$ and $B = H$. Thus, in this case, which will serve best our purposes here, a node is labeled as (i, j) where for the first coordinate $0 \leq i \leq n-1$ and for the second one $0 \leq j \leq n^{d-1} - 1$. This is illustrated in Fig. 2.

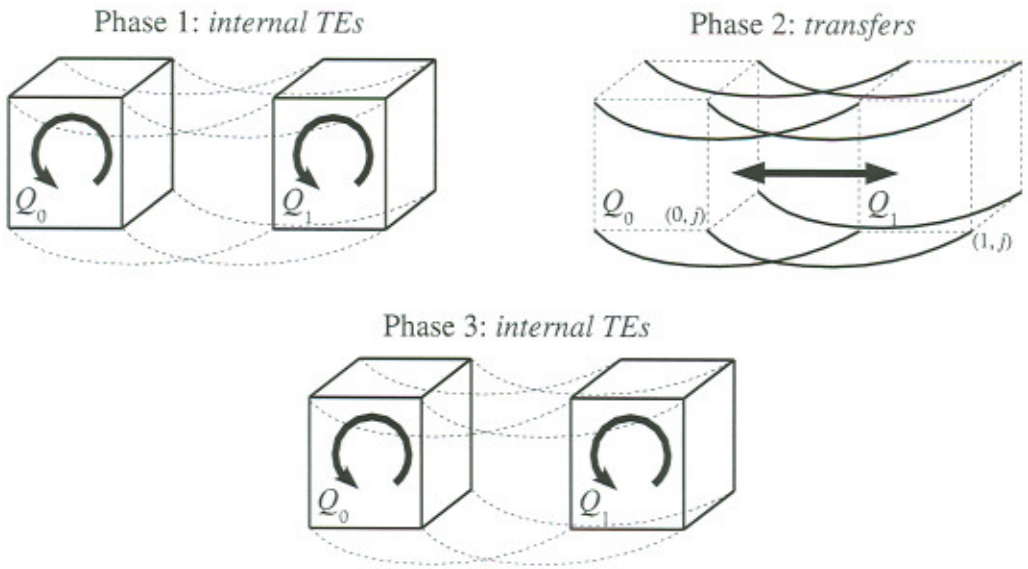

Figure 3. *Total exchange in hypercubes*

The product $B \times A$ can be viewed as n interconnected copies of A. Copy A_i, $i = 0, 1, \ldots, n-1$, contains all nodes whose first coordinate is equal to i, i.e. nodes $(i, \cdot)$. Corresponding nodes in the copies of A are interconnected according to B. Similarly, the network can be viewed as n^{d-1} interconnected copies of B, where copy B_j contains nodes $(\cdot, j)$ (see Fig. 2).

2. Preliminaries

Bertsekas et al [1] presented an optimal total exchange (TE) algorithm for hypercubes whose operation is shown in Fig. 3. The d-cube is partitioned in two subcubes (Q_0 and Q_1) of $d-1$ dimensions each. The algorithm consists of three phases. In the first phase, TEs within the two subcubes are performed. In the second phase every node j of Q_0 (Q_1) transfers all its messages for nodes of Q_1 (Q_0) to the corresponding node j of Q_1 (Q_0). In the final phase another internal TE withing the two subcubes is performed, concluding the operation. By appropriately scheduling the transfers, the second phase can overlap completely in time with the other two

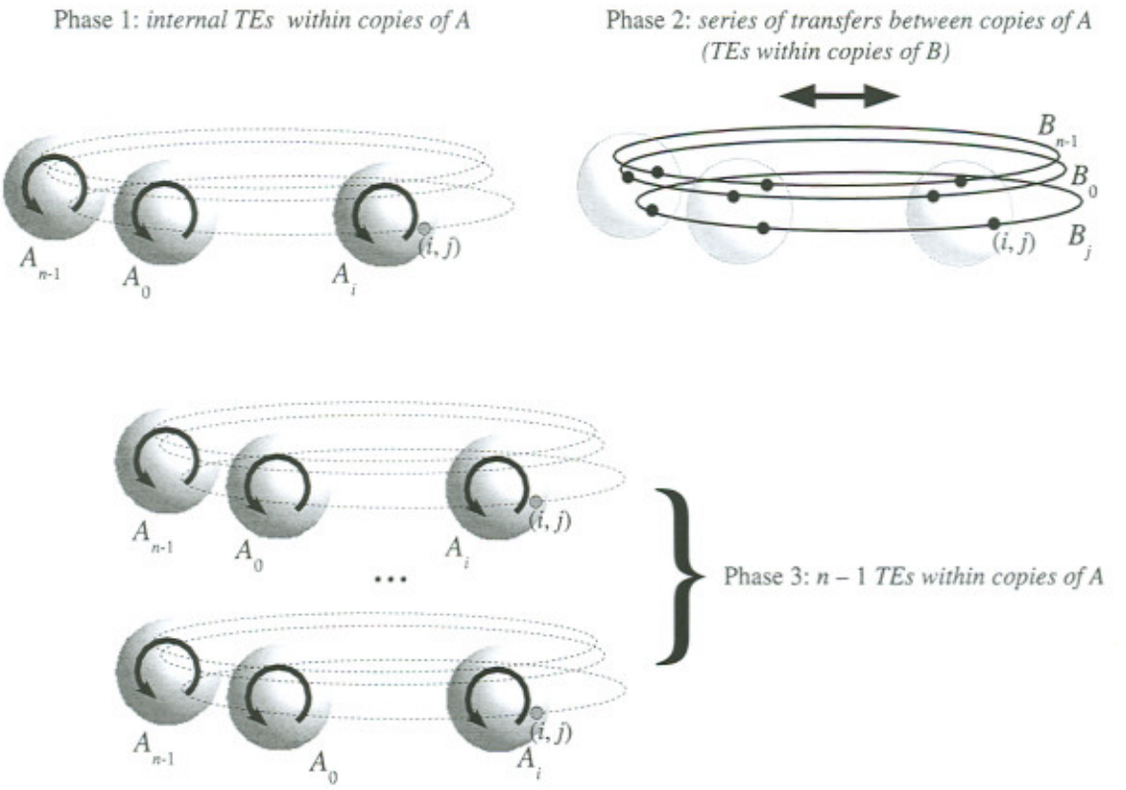

Figure 4. *Total exchange in a general multidimensional network*

phases and thus conclude the operation in the minimum possible number of steps.

The algorithm of [1] suggests a generalization for any kind of multidimensional network. In particular, if the network is $G = B \times A$, where $A = H^{d-1}$ and $B = H$, we may consider an analogous three-phase algorithm as shown in Fig. 4. However, the second and third phases are no longer simple; for example, the third phase will have to consist of a series of $n-1$ TEs within the n copies of A, each TE distributing messages originating from the $n-1$ other copies. What complicates things even more is the second phase. In the hypercube, a node simply transfers its messages to the unique corresponding node of the opposite subcube over their unique incident link. In G, however, the corresponding nodes of each copy of A may no longer be adjacent; they are interconnected according to B, and the transfers will have to utilize more than one link of B. Given that H can be any graph, the only systematic way for such transfers is to perform *total exchanges* within B. Consequently, the second

```
        MPTE(H^d) {
1         A = H^{d-1}, B = H;
2         Do in parallel
3           TE_A(), TE_B();
        }

        TE_B() {
4         For t = 1, 2, ..., n^{d-1}    // Phase 2
5           Do in parallel for all B_j, j = 0, 1, ..., n^{d-1} - 1
6             TotalExchange in B_j
        }

        TE_A() {
9         Do in parallel for all A_i, i = 0, 1, ..., n - 1    // Phase 1
10          TotalExchange in A_i: internal messages
11        For r = 1, ..., n - 1    // Phase 3
12          Do in parallel for all A_i, i = 0, 1, ..., n - 1
13            TotalExchange in A_i: external messages
        }
```

Figure 5. *Outline of the proposed total exchange algorithm*

phase should contain a series of TEs within B in order to distribute the messages needed for the third phase.

In what follows, we prove that the above generalization is indeed possible by carefully organizing the TEs of the second and the third phase. Moreover, it is possible to schedule the message transfers in such a way that phase two can completely overlap in time with phases one and three, which will be shown to be the best we can do. The outline of the algorithm is shown in Fig. 5 and, as will be seen, the only assumption is that we already know of a total exchange algorithm for graph H.

3. The algorithm

The algorithm outline in Fig. 5 consists of TEs within the copies of A and B; since A and B use different links, TE_A() and TE_B() can indeed be performed physically in parallel (there will be no link conflicts). Also, the copies of A have no links in common and thus any TE within a particular copy of A can be performed in parallel with a TE in any other copy of A. Whenever we say that a TE within A is performed, we will mean that all copies A_i, $i = 0, 1, \ldots, n-1$, of A perform internal TEs concurrently. The same goes for the copies of B.

Phase 1 of the algorithm is given in lines 9–10 and involves one TE within A ("internal" messages). Phase 2 is given in lines 4–6 and consists of a series of TEs in B ("transfers" between the copies of A). Finally, phase 3 is given by a series of $n-1$ TEs in A (lines 11–13), where the exchanged messages in each copy have originated at different copies of A ("external" messages).

During the TE in G, a node must scatter a total of $n^d - 1$ messages to all the other nodes in the network. Exactly $n^{d-1} - 1$ messages will be scattered in phase 1 of the algorithm, within a copy of A. The remaining $n^d - n^{d-1}$ messages will be transfered through the TEs in B, i.e. through TE_B(), before being further forwarded during phase 3. Since $B = H$, in every TE within B a node scatters $n-1$ of its messages. To send all its $n^d - n^{d-1}$ remaining messages, a total of $(n^d - n^{d-1})/(n-1) = n^{d-1}$ exchanges in B are required, hence line 4 in Fig. 5.

What remains is to schedule what messages are exchanged in phases 2 and 3 and show that phase 2 can indeed overlap in time with the other two phases. Having shown that, the algorithm MPTE(H^d) in Fig. 5 will be a correct total exchange algorithm for any multidimensional network $G = H^d$.

Let us assume that a total exchange algorithm for H requires T_H steps, equal to one time *slot*. Phase 2 occupies, then, n^{d-1} slots. In [6] we had derived an algorithm for two-dimensional networks, following the outline in Fig. 5 and we showed that it requires $n \times T_H$ steps, i.e. n slots in total. What we are going to show here is that Fig. 5 works correctly for d dimensions and it requires a total of n^{d-1} slots.

In order to prove what we need, we will use a recursive argument. In particular, we will assume that the algorithm works for $d-1$ dimensions as advertised, and we

will show that it works for d dimensions, too. Thus, phases 1 and 3, which exchange messages within $A = H^{d-1}$, are performed utilizing the algorithm as MPTE(H^{d-1}). Furthermore, algorithm MPTE(H^{d-1}) takes exactly n^{d-2} time slots.

In summary, we assume that:

- A TE algorithm for H requires 1 time slot ($= T_H$ steps).
- The algorithm in Fig. 5 works correctly for $d-1$ dimensions, and requires exactly n^{d-2} slots.

3.1. Scheduling the TEs in B

Since $G = B \times A$, where $A = H^{d-1}$ and $B = H$, a node's address is written as a tuple (i, j), where $0 \leq i \leq n-1$ and $0 \leq j \leq n^{d-1} - 1$. This node belongs to B_j and A_i, i.e. jth copy of B and the ith copy of A.

Let us focus in a TE within a particular copy of A. It consists of n^{d-1} nodes and thus any node will have to scatter $n^{d-1} - 1$ different messages. Isolate one particular node j and let

$$D_1(j), D_2(j), \ldots, D_{n^{d-1}-1}(j)$$

be the destinations of the scattered messages in the order they depart from node j during a TE (ties are broken arbitrarily if more than one messages must leave in one step).

During phase 2 of the algorithm messages are exchanged in the first dimension (B) in order to be further forwarded within the second dimension (A) through the TEs of phase 3. During one TE in phase 3, a node must scatter $n^{d-1} - 1$ messages, which will have been received from phase 2. However, during one TE in B a node scatters and gathers just $n-1$ messages. Hence, in order for a node to collect a set of $n^{d-1} - 1$ messages (so as to participate in one TE of phase 3),

$$R = \frac{n^{d-1} - 1}{n - 1} \tag{1}$$

TEs in B are required.

In order for a TE in B to be performed, each node (i, j) must scatter $n-1$ messages, each one to be delivered to one of the other nodes in B_j. Thus, the destinations of the messages (i, j) is going to scatter must be of the form form $(i', \cdot)$,

$t=1$	$t=2$	$\cdots$	$t=R$
$(i\oplus 1, D_1(j))$	$(i\oplus 1, D_n(j))$		$(i\oplus 1, D_{n^{d-1}-n+1}(j))$
$(i\oplus 2, D_2(j))$	$(i\oplus 2, D_{n+1}(j))$		$(i\oplus 2, D_{n^{d-1}-n+2}(j))$
$\vdots$	$\vdots$		$\vdots$
$(i\oplus (n-1), D_{n-1}(j))$	$(i\oplus (n-1), D_{2n-1}(j))$		$(i\oplus (n-1), D_{n^{d-1}-1}(j))$

Table 1. *Destinations of node (i,j)'s messages during the first $R=(n^{d-1}-1)/(n-1)$ TEs of phase 2*

where $i'=0,1,\ldots,n-1$ and $i'\neq i$. Letting $\oplus$ denote addition modulo n, in any TE in B_j the destinations of the messages node (i,j) scatters must be:

$$\{(i\oplus k,\cdot)\,|\,k=1,2,\ldots,n-1\}.$$

Since the purpose of the TEs in B is to provide messages for TEs in A (for phase 3), the second coordinate of the message destinations must be chosen so as to cover all nodes in A. In particular, the second coordinate will be chosen according to sequence $D(j)$ described above.

In Table 1 we give the destinations of the messages that node (i,j) should scatter during the first R TEs of phase 2. The tth column of the table shows the destinations of messages scattered during the tth slot. What is important to notice is that

1. in each column, the messages sent by node (i,j) constitute a perfect set of messages for a total exchange in B_j
2. after the R slots, every node in B_j will have scattered a total of $n^{d-1}-1$ messages; it will also have gathered $n^{d-1}-1$ messages which are destined to all other nodes of A.

In conclusion, Table 1 schedules the total exchanges in B so that they provide each node with a complete set of $n^{d-1}-1$ messages, ready to be distributed in A during the first TE of phase 3.

A general schedule for any time slot can be derived as follows. Let $t-1=qR+p$, where $0\leq p\leq R-1$. The set of messages node (i,j) scatters during the tth,

$t = 1, 2, \ldots, n^{d-1} - 1$, TE in B_j is:

$$\begin{aligned} S_{(i,j)}(t) \;=\; \{\; & m_{(i,j)}(i \oplus (q+1),\, D_{pn+1}(j)), \\ & m_{(i,j)}(i \oplus (q+2),\, D_{pn+2}(j)), \\ & \vdots \\ & m_{(i,j)}(i \oplus (n-1),\, D_{pn+(n-q-1)}(j)), \\ & m_{(i,j)}(i \oplus 1,\, D_{pn+(n-q)}(j)), \\ & \vdots \\ & m_{(i,j)}(i \oplus q,\, D_{(p+1)n-1}(j)) \;\} \end{aligned}$$

where the notation $m_{(i,j)}(k,l)$ means the message of node (i,j) destined for node (k,l). Since the first coordinate of the message destinations are $i' = 0, 1, \ldots, n-1$ except $i' = i$, $S_{(i,j)}(t)$ is a legal set of messages for a TE in B_j. Every R such TEs, the node also gathers $n^{d-1} - 1$ messages which are ready for distribution in a subsequent TE within A_i (during phase 3). What is more important is that it gathers $n-1$ messages per TE in B_j, in the order of $D(j)$, i.e. in the order they are are going to leave during the TE in A_i.

The TEs in B conclude in the final, n^{d-1}th slot; during this TE nodes in B_j exchange internal messages destined to each other. Node (i,j) send messages $m_{(i,j)}(i', j)$, for all $i' \neq i$.

Summarizing the above discussion, we present the final form of the algorithm in Fig. 6. Phase 2 starts concurrently with phase 1 and exchanges messages to be distributed through phase 3, which starts immediately after phase 1. The only thing remaining to be proved is that phase 2 is able to provide the messages needed by phase 3 *on time*, so that TE_A() and TE_B() can indeed be performed in parallel.

3.2. Correctness

Consider a node j in A and denote by

$$g_{d-1}(t), \;\; t = 1, 2, \ldots, n^{d-2}$$

the number of j's messages that leave j during slot t (recall that, by assumption, a TE in A occupies n^{d-2} slots). Based on the algorithm in Fig. 6 we have the following lemma.

```
        MPTE(H^d) {
1         A = H^{d-1}, B = H;
2         Do in parallel
3           TE_A(), TE_B();
        }

        TE_B() {
4         For t = 1, 2, ..., n^{d-1} - 1    // Phase 2
5           Do in parallel for all B_j, j = 0, 1, ..., n^{d-1} - 1
6             TotalExchange in B_j: node (i, j) scatters S_{(i,j)}(t)
7         Do in parallel for all B_j, j = 0, 1, ..., n^{d-1} - 1
8           TotalExchange in B_j: internal messages
        }

        TE_A() {
9         Do in parallel for all A_i, i = 0, 1, ..., n - 1    // Phase 1
10          TotalExchange in A_i: internal messages
11        For r = 1, ..., n - 1    // Phase 3
12          Do in parallel for all A_i, i = 0, 1, ..., n - 1
13            TotalExchange in A_i: node (i, j) scatters messages arriving from B_j
        }
```

Figure 6. *The proposed total exchange algorithm in its final form*

Lemma 1 *If $g_d(t)$ is the number of node (i,j)'s messages leaving during slot t, then:*

$$g_d(t) = \begin{cases} d(n-1), & t = 1 \\ (d-1)(n-1), & t = 2, 3, \ldots, n \\ \vdots & \\ 2(n-1), & t = n^{d-3}+1, n^{d-3}+2, \ldots, n^{d-2} \\ (n-1), & t = n^{d-2}+1, n^{d-2}+2, \ldots, n^{d-1}. \end{cases}$$

Proof. In each TE in B, a node scatters $n-1$ of its own messages. Consequently, for the n^{d-1} slots of phase 2, the number of own messages leaving a node through

dimension B is:

$$g_d^B(t) = n-1,\ t = 1, 2, \ldots, n^{d-1}.$$

As seen from Fig. 6, in dimension A, the own messages all leave during phase 1 (phase 3 only distributes foreign messages). Hence, the number of own messages leaving a node through dimension A is:

$$g_d^A(t) = \begin{cases} g_{d-1}(t), & t = 1, 2, \ldots, n^{d-2} \\ 0, & t = n^{d-2}+1, n^{d-2}+2, \ldots, n^{d-1}. \end{cases}$$

Thus node (i, j)'s own messages leave it as in:

$$g_d(t) = g_d^A(t) + g_d^B(t) = \begin{cases} g_{d-1}(t) + n - 1, & t = 1, 2, \ldots, n^{d-2} \\ n-1, & t = n^{d-2}+1, n^{d-2}+2, \ldots, n^{d-1}, \end{cases}$$

which easily leads to the required result, since in graph H, $g_1(1) = n-1$. ■

Fig. 7 shows graphically the situation for a network with $d = 4$ dimensions, and $n = 3$ nodes in each dimension. At the top of the figure, $g_{d-1}(t)$ is illustrated for a TE in A, which occupies $n^{d-2} = 9$ slots. During slot 1 a node scatters 6 of its own messages, in slots 2 and 3 it scatters 4 and in the next 6 slots it scatters 2 $(= n-1)$ messages according to Lemma 1.

The figure also unfolds the TEs in the two dimensions of $G = B \times A$ according to the proposed algorithm (Fig. 6). Phase 2 contains TEs in B and starts simultaneously with phase 1. During the first $R = (n^{d-1}-1)/(n-1) = 13$ slots, messages are distributed so that they can be further forwarded by the first $(r = 1)$ TE of phase 3. In the next 13 slots dimension B provides messages needed for the second $(r = 2)$ TE of phase 3. The final TE in B exchanges internal messages.

What Lemma 1 establishes, if applied to $A = H^{d-1}$, is two things: first, at every slot *at least* $n-1$ messages depart from each node; second, in order for a TE in A to be performed, not all messages are needed at the beginning. Since messages leave a node in batches according to $g_{d-1}(t)$, it is seen that in G a TE in phase 3 may commence before all required messages have been received from the first dimension (B). In fact, we prove next that B is able to provide all the messages required by A before they are actually needed.

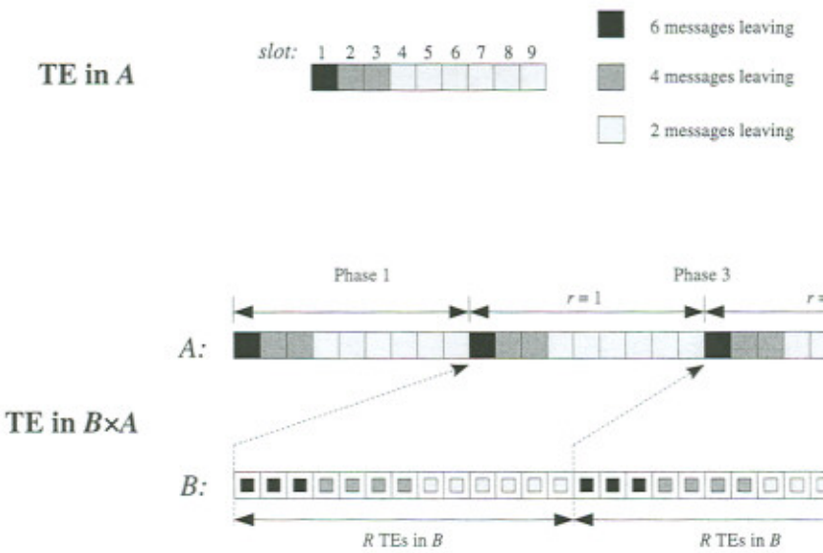

Figure 7. *Timing diagram for TE in $A = H^{d-1}$ and in $G = B \times A$, where $B = H$, $d = 4$ and $n = 3$*

Theorem 1 *All messages needed by the rth TE of phase 3, $r = 1, 2, \ldots, n-1$, will have arrived on time through the TEs in B.*

Proof. Since the first n^{d-2} slots are occupied by phase 1, the rth TE in phase 3 occurs in the slot interval

$$(rn^{d-2}, (r+1)n^{d-2}]$$

where the notation $(x, y]$ means all slots from x to y not including x, that is slots $x+1$, $x+2$, ..., y. Let $E_A = (r+1)n^{d-2}$ denote the right end of the interval, i.e. the time the rth TE in A finishes.

On the other hand, we know that R TEs in B are needed to provide messages for one TE in A (Eq. (1)). Thus, the messages needed for the rth TE of phase 3 arrive from B in the interval:

$$((r-1)R, rR],$$

Let $E_B = rR$ be the time the last message from B arrives.

First, notice that $E_B < E_A$. Indeed,

$$E_B < E_A \quad \Leftrightarrow \quad r\frac{n^{d-1}-1}{n-1} < (r+1)n^{d-2}$$

$$\Leftrightarrow \quad n^{d-1} - (r+1)n^{d-2} + r > 0. \tag{2}$$

It can be easily seen that because $n, d \geq 2$ the expression $n^{d-1} - (r+1)n^{d-2} + r$ is a decreasing function of r. Consequently, the minimum occurs for $r = n-1$, giving the value $n-1$ which is obviously > 0. As a result (2) is always true and hence $E_B < E_A$.

We now claim that $E_B < E_A$ guarantees that all messages needed by the rth TE in A will have arrived on time. We know that TEs in B provide every node with $n-1$ messages at each slot, continuously, to be used by a subsequent total exchange in A. On the other hand a TE in A consumes, during phase 3, at least $n-1$ message per time slot, continuously, till the end of the TE in G (see Lemma 1).

We will prove our claim by contradiction, that is, we will assume that $E_B < E_A$ but at some slot x, A will not have received all the messages it needs. Since, after slot x, A needs continuously *at least* $n-1$ messages, and B can provide *at most* $n-1$ messages per slot, B cannot keep up. In other words, *there is no slot after* x in which A will have all the messages it needs. But this is a contradiction; due to the fact that $E_B < E_A$, the entirety of messages A needs will have been received from B before the last slot. Hence the claim. ■

We have shown that the algorithm in Fig. 6 is a correct TE algorithm for any d-dimensional network $G = H^d$. The algorithm, which is recursive in nature, consists of series of TEs in each dimension. In effect, based on a TE algorithm for network H we have synthesized a general algorithm for any multidimensional network H^d.

4. Optimality

The time requirements of our algorithm can be calculated in a straightforward manner. Since a TE in A was assumed to occupy n^{d-2} slots, it is seen that the n TEs of phase 1 and phase 3, i.e. TE_A(), require a total of n^{d-1} slots. Similarly, TE_B() consists of n^{d-1} TEs in B, each one lasting for 1 slot, thus TE_B() also needs n^{d-1} slots. Since TE_A() and TE_B() occur simultaneously, we have the following theorem.

Theorem 2 *The proposed algorithm takes $n^{d-1}T_H$ steps, where T_H is the time*

needed for a TE in H.

A lower bound for the total exchange problem can be found as follows. Consider any network $G = (V, E)$ with N nodes. Every node has to send $N - 1$ distinct messages, one for each of the other nodes. Partition the vertex set V arbitrarily in two disjoint sets V_1 and V_2 such that $V_1 \cup V_2 = V$. Let $C_{V_1V_2}$ be the number of edges in E joining the two parts, i.e. edges ij such that $i \in V_1$ and $j \in V_2$. Messages from nodes in V_1 destined for nodes in V_2 must cross those $C_{V_1V_2}$ edges. The total number of such messages is $|V_1||V_2|$. Since only $C_{V_1V_2}$ messages are able to pass from V_1 to V_2 at a time, we obtain the following lower bound for total exchange time:

$$T_{TE} \geq \frac{|V_1||V_2|}{C_{V_1V_2}}. \tag{3}$$

Theorem 3 *Let $G = H^d$, where $d \geq 2$. If total exchange in H can be performed in time equal to the lower bound of (3) then the same holds for G.*

Proof. From Theorem 2, total exchange in G requires $T = n^{d-1}T_H$ time units, where $n = |V_H|$. If T_H achieves the lower bound in (3) then there exists a partition V_{H_1}, V_{H_2} of the node set of H such that $T_H = \frac{|V_{H_1}||V_{H_2}|}{C_{V_{H_1}V_{H_2}}}$, where $C_{V_{H_1}V_{H_2}}$ is the number of links separating the two parts.

Consider the following partition of V, the node set of $G = H \times H^{d-1}$:

$$\begin{aligned} V_1 &= \bigcup_{i \in V_{H_1}} (i, *) \\ V_2 &= \bigcup_{i \in V_{H_2}} (i, *), \end{aligned}$$

where $*$ is the don't-care symbol, taking any value between 0 and $n^{d-1} - 1$. Then clearly, $|V_1| = |V_{H_1}|n^{d-1}$ and $|V_2| = |V_{H_2}|n^{d-1}$. Notice that G contains n^{d-1} copies of H and that in order to separate the two parts we only need to disconnect each copy of H by removing only links in the first dimension. Since $C_{V_{H_1}V_{H_2}}$ links are needed to disconnect each copy of H, we obtain

$$C_{V_1V_2} = n^{d-1}C_{V_{H_1}V_{H_2}}.$$

Thus, V_1 and V_2 is a partition of G such that

$$\frac{|V_1||V_2|}{C_{V_1V_2}} = \frac{n^{d-1}|V_{H_1}|n^{d-1}|V_{H_2}|}{n^{d-1}C_{V_{H_1}V_{H_2}}} = n^{d-1}\frac{|V_{H_1}||V_{H_2}|}{C_{V_{H_1}V_{H_2}}} = n^{d-1}T_H,$$

which is equal to T, the time needed for total exchange in G. Thus the bound in (3) is tight for G, too. ■

Summarizing, the algorithm in Fig. 6 is an all-port total exchange algorithm for homogeneous networks of dimensionality $d \geq 2$. If TE in H can be performed in time equal to the lower bound of (3) then the proposed algorithm optimally solves the TE problem in $G = H^d$. For example, in [5] we gave algorithms that achieve the lower bound in rings. Consequently, Fig. 6 leads to an optimal total exchange algorithm for homogeneous tori (k-ary d-cubes).

5. Summary

In this paper we studied the total exchange problem in the context of homogeneous multidimensional networks, under the all-port model. The algorithm we derived solves the problem in any such network and can be seen as a generalization of the algorithm presented in [1] for hypercubes.

In particular, we proved that a general solution for the problem can be synthesized by utilizing only a total exchange algorithm designed for one dimension. Moreover, this approach can yield optimal algorithms for packet-switched networks, given that the algorithm for the single dimension achieves the lower bound of (3).

The proposed algorithm works for *any* multidimensional network, including hypercubes, k-ary n-cubes, generalized hypercubes, etc. For many of these networks our algorithm behaves optimally since optimal algorithms for simple dimensions are already known in the literature (e.g. [5] for rings).

The algorithm, as it is, can be applied to circuit switched and wormhole routed networks as well. However, it is questionable whether it performs optimally or not in such a context. Wormhole-routed TE in multidimensional networks is an interesting area of future research since the known algorithms for this model are restricted to hypercube and mesh topologies.

References

[1] D. P. Bertsekas, C. Ozveren, G. D. Stamoulis, P. Tseng and J. N. Tsitsiklis, "Optimal communication algorithms for hypercubes," *J. Parallel Distrib. Comput.*, Vol. 11, pp. 263–275, 1991.

[2] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Englewoods Cliffs, N.J.: Prentice - Hall, 1989.

[3] L. N. Bhuyan and D. P. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," *IEEE Trans. Comput.*, Vol. C-33, No. 4, pp. 323–333, Apr. 1984.

[4] S. H. Bokhari, "Multiphase complete exchange on a circuit switched hypercube," in *Proc. 1991 Int'l Conf. Parall. Proc.*, Aug. 1991, pp. I-525 – I-529.

[5] V. V. Dimakopoulos and N. J. Dimopoulos, "Optimal total exchange in linear arrays and rings," in *Proc. ISPAN'94, Int'l Symp. Parall. Arch., Algor. and Networks*, Kanazawa, Japan, Dec. 1994, pp. 230–237.

[6] V. V. Dimakopoulos and N. J. Dimopoulos, "A theory for total exchange in multidimensional interconnection networks," *IEEE Trans. Parall. Distrib. Syst.*, Vol. 9, No. 7, pp. 639–649, July 1998.

[7] A. Edelman, "Optimal matrix transposition and bit reversal on hypercubes: all-to-all personalized communication," *J. Parallel Distrib. Comput.*, Vol. 11, No. 4, pp. 328–331, 1991.

[8] P. Fraigniaud and E. Lazard, "Methods and problems of communication in usual networks," *Discrete Appl. Math.*, Vol. 53, pp. 79–133, 1994.

[9] D. B. Gannon and J. van Rosendale, "On the impact of communication complexity on the design of parallel numerical algorithms," *IEEE Trans. Comput.*, Vol. C-33, No. 12, pp. 1180–1194, Dec. 1984.

[10] S. L. Johnsson and C. - T. Ho, "Optimum broadcasting and personalized communication in hypercubes," *IEEE Trans. Comput.*, Vol. 38, No. 9, pp. 1249–1268, 1989.

[11] C. C. Lam, C. - H. Huang and P. Sadayappan, "Optimal algorithms for all-to-all personalized communication on rings and two dimensional tori," *J. Parallel Distrib. Comput.*, Vol. 43, pp. 3–13, 1997.

[12] Y. Saad and M. H. Schultz, "Data communications in hypercubes," *J. Parallel Distrib. Comput.*, Vol. 6, pp. 115–135, 1989.

[13] Y. - J. Suh and K. G. Shin, "All-to-all personalized communication in multidimensional torus and mesh networks," *IEEE Trans. Parall. Distrib. Syst.*, Vol. 12, No. 1, pp. 38–59, Jan. 2001.

[14] Y. - J. Suh and S. Yalamanchili, "All-to-all communication with minimum start-

up costs in 2D/3D tori and meshes," *IEEE Trans. Parall. Distrib. Syst.*, Vol. 9, No. 5, pp. 442–458, May 1998.

[15] N. S. Sundar, D. N. Jayashima, D. K. Panda and S. Sadayappan, "Hybrid algorithms for complete exchange in 2D meshes," *IEEE Trans. Parall. Distrib. Syst.*, Vol. 12, No. 12, pp. 1201–1218, Dec. 2001.

[16] R. Thakur and A. Choudhary, "All-to-all communication on meshes with wormhole routing," in *Proc. 8th Int'l Parall. Proc. Symp.*, Cancun, Mexico, Apr. 1994, pp. 561–565.

[17] E. A. Varvarigos and D. P. Bertsekas, "Communication algorithms for isotropic tasks in hypercubes and wraparound meshes," *Parallel Comput.*, Vol. 18, pp. 1233–1257, 1992.