

Dynamic Broadcasting in Parallel Computing¹

by

Emmanouel A. Varvarigos and Dimitri P. Bertsekas²

Abstract

We consider the problem where broadcast requests are generated at random time instants at each node of a multiprocessor network. In particular, in our model packets arrive at each node of a network according to a Poisson process, and each packet has to be broadcast to all the other nodes. We propose an on-line decentralized routing scheme to execute the broadcasts in this dynamic environment. A related, although static, communication task is the partial multinode broadcast task, where $M \leq N$ arbitrary nodes of an N -processor network broadcast a packet to all the other nodes. The results that we obtain for the dynamic broadcasting scheme apply to any topology, regular or not, for which partial multinode broadcast algorithms with certain properties can be found. For the dynamic scheme we find an upper bound on the average delay required to serve a broadcast request, and we evaluate its stability region. As an application we give a near-optimal partial multinode broadcast algorithm for the hypercube network. The stability region of the corresponding hypercube dynamic scheme tends to the maximum possible as the number of nodes of the hypercube tends to infinity. Furthermore, for any fixed load in the stability region, the average delay is of the order of the diameter of the hypercube.

¹ Research supported by NSF under Grant NSF-DDM-8903385 and by the ARO under Grant DAAL03-86-K-0171.

² Laboratory for Information and Decision Systems, M.I.T, Cambridge, Mass. 02139.



1. INTRODUCTION

Broadcasting is the operation where a packet is copied from a node to all the other nodes of a network. Because of the variety of applications which involve broadcasts, such operations are often implemented as communication primitives in parallel computers. One of the most frequent broadcasting tasks is the *multinode broadcast* (abbreviated MNB), where every node of a network broadcasts a packet to all the other nodes. The MNB arises, for example, in iterations of the form

$$x = f(x), \quad (1)$$

where each processor i computes a component (or a block of components) x_i of the vector x . If iteration (1) takes place synchronously, and all the components of x change during each iteration, it is necessary that at the end of an iteration every processor i broadcasts the updated value of x_i to all the other processors for use at the next iteration; this is a MNB.

In iterations of the form given above it is very probable that only few of the components of the vector x will change appreciably during an iteration. If the new value of a component is close to its previous value, there is no reason to waste communication bandwidth in order to broadcast it. This gives rise to the task, where only few, but arbitrary, processors broadcast a packet. We call this generalization of the MNB task a *partial multinode broadcast* (or PMNB). The PMNB arises often in applications, and we believe that it deserves a position among the prototype tasks of a communication library.

The MNB and the PMNB communication tasks are *static* broadcasting tasks, that is, they assume that at time $t = 0$ some nodes have a packet to broadcast, and this takes place once and for all. Static broadcasting tasks in multiprocessor networks have been studied extensively in the literature ([BeT89], [BOS91], [Ho90], [JoH89], [LEN90], [VaB90]). In this paper we consider the *dynamic version of the broadcasting problem*. We assume that packets are generated at each node according to a Poisson process with rate λ independently of the other nodes, and each packet has to be broadcast to all the other nodes. We propose a dynamic scheme to execute the broadcasts in this dynamic environment, and we evaluate its performance. The assumption of Poisson arrivals is made only because the mathematics of the analysis require it and is inessential for the schemes that we present. We are interested in two performance criteria. The first is the average delay, that is, the average time between the arrival of a packet at a node, and the completion of its broadcast. The second criterion is the stability region of the scheme, that is, the maximum load that it can sustain with the average delay being finite. We set two objectives for a dynamic broadcasting scheme: stability for as big a load as possible, and average delay which is of the order of the diameter for

any fixed load in the stability region.

The dynamic broadcasting problem is important for a variety of reasons. First, consider the case where iteration (1) takes place asynchronously. Each processor i computes at its own speed without waiting for the others, and broadcasts the updated value of x_i whenever it is available. Asynchronous parallel computation naturally results in a dynamic communication environment like the one we are considering. Asynchronous computation algorithms are increasing in importance (see e.g. [BeT89]) as a way to circumvent the synchronization penalty. The latter is a major cause of inefficiency in parallel computers, especially when the processors are not equally powerful, or when the load distribution is not balanced. In such a case, static algorithms, for example the MNB, become inefficient, since a fast processor would have to wait for all the other processors before starting a MNB. A second reason that makes algorithms for static tasks difficult to use is that the task must be detected and identified by the compiler, or the corresponding communication subroutine must be called explicitly by the programmer. It is plausible that the programmer and the compiler may fail to identify such communication tasks. Even more importantly, broadcasts may be generated in *run-time*, during the execution of a program. This poses a problem because in order to use precomputed static communication algorithms we must know the communication pattern in advance. Multitasking and time-sharing make the communications even less predictable, and the use of static communication algorithms more difficult. The preceding reasons motivate dynamic broadcasting schemes that will run continuously, and execute on-line the broadcast requests.

The only previous work on dynamic broadcasting we know of is that of Stamoulis [Sta91] for the hypercube network. There are two algorithms of Stamoulis that are most interesting from a theoretical point of view: the direct scheme, and the indirect scheme. In the direct scheme, d spanning trees, where d is the dimension of the hypercube, are defined for each node, having the node as a root. A packet that arrives at a node selects at random one of the d trees of the node and is broadcast on it. The direct scheme meets the stability objective described above, but its average delay analysis is approximate. In the indirect scheme, d spanning trees are defined in the hypercube. A packet that arrives at some node selects at random one of these trees. It is then sent to the root of that tree, and from there it is broadcast to all the other nodes using links of the tree. The indirect scheme meets the delay objective, but its stability region is not the maximum possible. Therefore, the two hypercube schemes of [Sta91] do not provably satisfy both performance objectives.

Our dynamic broadcasting scheme has a fundamentally different philosophy: it relies heavily on finding efficient PMNB algorithms that are used as a subroutine of the dynamic scheme. Furthermore, our scheme is very general: it applies to any network for which efficient PMNB algorithms can be found. Also for the hypercube network, our scheme has a stability region that tends to the

maximum possible as the number of nodes tends to infinity, while its average delay for any fixed load in the stability region is of the order of the diameter. Thus, our scheme compares favorably with Stamoulis' hypercube algorithms none of which meets optimally the stability and the delay objective.

Our dynamic broadcasting scheme and the corresponding performance analysis apply to any network for which we can find communication algorithms that execute the PMNB communication task in time

$$XM + V,$$

where M is the number of nodes that have a packet to broadcast, called *active nodes*, and X, V are scalars that are independent of M (they may depend on the size of the network). For networks where such PMNB algorithms exist, we can easily devise corresponding dynamic broadcasting schemes that satisfy some average packet delay and stability guarantees. The dynamic broadcasting scheme consists, merely, of executing successive PMNB algorithms, each starting after the previous one has finished. Our scheme is modelled after reservation and polling schemes for multiaccess communication. The network is viewed as a channel, and the nodes as users of the channel. For analytical purposes, the first V time units of the PMNB algorithm are considered as a reservation interval, where some organizational work is performed, and the following MX time units as a data interval, where users with reservations transmit a packet.

Our dynamic broadcasting scheme requires the existence of a partial multinode broadcast algorithm with certain properties. For the hypercube network, we will present such partial multinode broadcast algorithms. In particular, we will describe three different algorithms to execute a partial multinode broadcast in a hypercube. Each of them uses a different communication model. The first PMNB algorithm is a practical but suboptimal one. When this PMNB algorithm is used as a part of the dynamic broadcasting scheme, the latter does not provably meet the stability and average packet delay objectives that we have set. The second and third algorithms are near-optimal. A dynamic broadcasting scheme based on any one of these two algorithms meets our performance objectives: its stability region tends to the maximum possible as the number of nodes tends to infinity, and the average packet delay for any fixed load in the stability region is of the order of the diameter of the hypercube. In the companion paper [VaB92] we present near-optimal PMNB algorithms for d -dimensional meshes, which also give rise to efficient dynamic broadcasting schemes.

The structure of the paper is the following. In Section 2 we describe the dynamic broadcasting scheme in a given network, assuming that a PMNB algorithm with certain properties is available for that network. We also state the dynamic broadcasting theorem, which is the main result of the paper. In Section 3 we evaluate the performance of our dynamic broadcasting scheme. In particular,

in Subsection 3.1 we describe an auxiliary queuing system, which we will use to prove the dynamic broadcasting theorem. In Subsection 3.2 we prove the dynamic broadcasting theorem, which gives an estimate on the average packet delay of the dynamic broadcasting scheme. Section 4 describes three different algorithms to execute a partial multinode broadcast in a hypercube. Finally, Section 5 applies the dynamic broadcasting theorem to the case of the hypercube, using the results obtained in Section 4.

2. DYNAMIC BROADCASTING SCHEMES

In this section we will describe the dynamic broadcasting scheme for a general network. We will assume that an algorithm that executes the PMNB task in that network is given, and that it requires $XM + V$ time units, where M is the number of active nodes, that is, the nodes that have a packet to broadcast, and X, V are scalars independent of M . We also assume that during the PMNB algorithm each node learns the number of active nodes M .

Our scheme is merely a repetition of successive partial multinode broadcast algorithms, each starting when the previous one has finished (see Fig. 1). The time axis is, therefore, divided into PMNB intervals. Within each PMNB interval, a PMNB is executed, involving exactly one packet from each of the M nodes that are active at the start of the interval. Each PMNB interval is divided into two parts. The first part is called *reservation interval*. Its duration can be upper bounded by a known constant V that depends only on the size of the network, and is independent of the number of active nodes M . During the reservation interval each active node s can be viewed as making a reservation for the broadcast interval (as we will see in Subsections 4.2 and 4.3 for the hypercube this is done simply by setting $x_s = 1$ in the rank computation phase). Usually, in the reservation interval some global information is gathered at the nodes (e.g., the total number of active nodes M , and other information), and some additional organizational work is performed. For example, in the PMNB algorithms described in Subsections 4.2 and 4.3 for the hypercube, and in the algorithms described in the companion paper [VaB92] for the d -dimensional mesh, the packets move during the reservation interval to more favorable intermediate locations. The second part of a PMNB interval is called *broadcast interval*. Its duration is equal to XM , and is therefore known once M is known. The broadcast interval is empty if there are no packets to broadcast ($M = 0$). Thus, even though the duration of each partial multinode broadcast is random (because packet arrivals are random), it is known to all the nodes of the network, because each node learns during the broadcast interval the number M of active nodes and, from there, the duration of the following

broadcast interval. Therefore, if the nodes initiate the dynamic broadcast scheme at the same time, no further synchronization is needed. For the PMNB algorithms proposed in Subsections 4.2 and 4.3 for hypercubes (and in [VaB92] for d -dimensional meshes), in the broadcast interval the packets are broadcast from the intermediate locations to all other nodes. The details of the broadcast interval are, however, irrelevant, and all that matters for our purposes, is that the duration of the broadcast interval is less than or equal to MX .

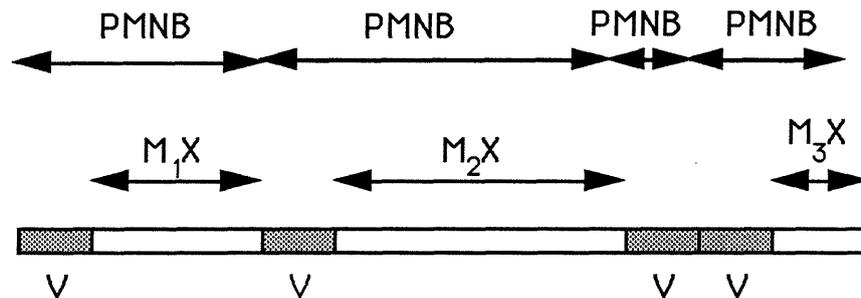


Figure 1: The dynamic broadcasting scheme. Each PMNB interval consists of two intervals: a reservation interval (marked by gray) of duration V , and a broadcast interval of duration MX , where M is the number of active nodes at the start of the PMNB interval.

It is important for the performance of the dynamic scheme that the duration of the PMNB algorithms is *linear* in the number of active nodes M , with the constant of proportionality being the smallest possible.

The main theorem that we prove in the paper is the following.

Dynamic Broadcasting Theorem: Assume that for a given N -processor network there exists an algorithm that executes the PMNB communication task in time

$$XM + V,$$

where M is the number of nodes that have a packet to broadcast and X, V are scalars that are independent of M (they may depend on the size of the network). Assume that during the PMNB algorithm each node learns the value of M . Then the dynamic broadcasting scheme that uses this PMNB algorithm as described above has the following performance characteristics. If the packets to be broadcast arrive at each node of the network according to a Poisson process with rate λ , independently of the other nodes, the average packet delay T satisfies:

$$T = W + X + aNX \leq W + X + \min\left(\frac{N-1}{2}X, \rho W\right),$$

where

$$W = \frac{\rho X}{2(1-\rho-\lambda V)} + \frac{(1-\rho)V}{2(1-\rho-\lambda V)} + \frac{(1-\rho a-\lambda V)V}{1-\rho-\lambda V},$$

3. Analysis of the Dynamic Broadcasting Scheme

$$\rho = \lambda NX,$$

and a is a scalar satisfying

$$\frac{\bar{M} + (\hat{M} - 1)(2\bar{M} - \hat{M})}{2N\bar{M}} - \frac{1}{2N} \leq a \leq \frac{1}{2} - \frac{1}{2N},$$

where

$$\bar{M} = \frac{\lambda NV}{1 - \rho},$$

and \hat{M} is the smallest integer which is strictly larger than \bar{M} .

Note that the dynamic broadcasting scheme is stable for $\rho \leq 1 - \lambda V$, or

$$\rho \leq \frac{1}{1 + V/(NX)}.$$

The dynamic broadcasting theorem is proved in the following section.

3. ANALYSIS OF THE DYNAMIC BROADCASTING SCHEME

In this section we will prove the dynamic broadcasting theorem. We first describe an auxiliary queueing system that will be used in the main proof given in Subsection 3.2.

3.1. Limited Service Gated Reservation System with Shared Reservation and Data intervals

In this subsection we describe an auxiliary queueing system, called *limited service gated reservation system with shared reservation and data intervals*, which is a reservation system for multiaccess communication. We are interested in the average delay required to serve a packet in this system. This average delay will be used in the next subsection to evaluate the average packet delay of the dynamic broadcasting scheme. The analysis of the auxiliary queueing system is based on unpublished research by D. P. Bertsekas and R. G. Gallager. We will give this analysis in detail, since it is important for our purposes and it does not appear anywhere else.

The auxiliary queueing system is defined as follows. Consider N traffic streams, each corresponding to a different user, which share a common channel. The channel is used both for packet transmissions and reservations. In particular, the time axis is divided into data intervals, where

3. Analysis of the Dynamic Broadcasting Scheme

actual data is transmitted, and reservation intervals used for scheduling future data. Each user has a separate queue, and the queues are served in cyclical order. Users make reservations during the same reservation interval, and transmit at most one packet each in the subsequent data interval. A packet can be transmitted in a data interval only if it arrived before the beginning of the previous reservation interval. For this system the following theorem holds.

Theorem 1: Let the arrival processes of packets at the users of the system be independent Poisson processes, each with rate λ . Let also \bar{X} , \bar{X}^2 be the first and second moments of the packet transmission times and \bar{V} , \bar{V}^2 be the first and second moments of the duration of a reservation interval. Then the mean waiting time in queue for this system is

$$W = \frac{\lambda N \bar{X}^2}{2(1 - \rho - \lambda \bar{V})} + \frac{(1 - \rho) \bar{V}^2}{2(1 - \rho - \lambda \bar{V}) \bar{V}} + \frac{(1 - \rho a - \lambda \bar{V}) \bar{V}}{1 - \rho - \lambda \bar{V}}, \quad (2)$$

where $\rho = \lambda N \bar{X}$ is the utilization factor, and a satisfies

$$\frac{\bar{K} + (\hat{K} - 1)(2\bar{K} - \hat{K})}{2N\bar{K}} - \frac{1}{2N} \leq a \leq \frac{1}{2} - \frac{1}{2N},$$

where

$$\bar{K} = \frac{\lambda N \bar{V}}{1 - \rho}$$

is the average number of packets per data interval, and \hat{K} is the smallest integer which is strictly larger than \bar{K} .

Proof: Consider the i th packet arrival into the system and suppose that the user associated with packet i is user j . This packet must wait in queue for the residual time R_i until the end of the current packet transmission or reservation interval. It must also wait for the transmission of the N_i packets that must be transmitted before packet i . Finally the packet must wait for the duration of reservation intervals. Thus, the expected waiting time of packet i is

$$E(W_i) = E(R_i) + E(N_i) \bar{X} + E(Y_i),$$

where Y_i is the duration of all the whole reservation intervals during which packet i must wait before being transmitted. The expected waiting time is therefore

$$W = \lim_{i \rightarrow \infty} E(W_i) = E(R_i) + E(N_i) \bar{X} + E(Y_i). \quad (3)$$

From Little's law we get

$$E(N_i) = \lambda N W. \quad (4)$$

Let Q_i be the number of packets in the queue of user j found by packet i upon arrival, and m_i be the number (0 or 1) of packets of user j that will start transmission between the time of arrival of

3. Analysis of the Dynamic Broadcasting Scheme

packet i and the end of the frame at which packet i arrives. Then

$$E(Y_i) = (1 + E(Q_i) - E(m_i)) \bar{V}. \quad (5)$$

From Little' law we have

$$Q = \lim_{i \rightarrow \infty} E(Q_i) = \lambda W. \quad (6)$$

The mean residual time $R = \lim_{i \rightarrow \infty} R_i$ can be calculated as in [BeG87] to be

$$R = \frac{\lambda N \bar{X}^2}{2} + \frac{(1 - \rho) \bar{V}^2}{2\bar{V}}. \quad (7)$$

Combining Eqs. (3)-(7) we get

$$W = \lambda N W \bar{X} + \left(1 + \lambda W - \lim_{i \rightarrow \infty} E(m_i)\right) \bar{V} + \frac{\lambda N \bar{X}^2}{2} + \frac{(1 - \rho) \bar{V}^2}{2\bar{V}}. \quad (8)$$

To find W , it remains to calculate $\lim_{i \rightarrow \infty} E(m_i)$.

There are two possibilities regarding the time of arrival of packet i .

a) packet i arrives during a reservation interval. This event, call it A , has steady state probability $1 - \rho$:

$$P(A) = 1 - \rho.$$

Since the ratio of the average data interval length to the average reservation interval length is $\rho/(1 - \rho)$ we see that the average steady state length of data interval is $\rho \bar{V}/(1 - \rho)$. Therefore, the average steady state number of packets per user in a data interval is

$$\frac{\rho \bar{V}}{(1 - \rho) N \bar{X}} = \frac{\lambda \bar{V}}{1 - \rho}.$$

This also equals the steady state value of $E(m_i | A)$ in view of the symmetry with respect to the users:

$$\lim_{i \rightarrow \infty} E(m_i | A) = \frac{\lambda \bar{V}}{1 - \rho}.$$

b) Packet i arrives during a data interval. This event, call it B , has steady state probability ρ :

$$P(B) = \rho.$$

Denote

$$a = \lim_{i \rightarrow \infty} E(m_i | B),$$

$$a_k = \lim_{i \rightarrow \infty} E(m_i | B, \text{ the data interval of arrival of packet } i \text{ contains } k \text{ packets}).$$

3. Analysis of the Dynamic Broadcasting Scheme

Assuming $k > 0$ packets are contained in the data interval of arrival, there is equal probability $1/k$ of arrival during the transmission of any of these packets. Therefore

$$a_k = \sum_{n=1}^k \frac{1}{k} \frac{k-n}{N} = \frac{k-1}{2N}.$$

Let $P(k)$ be the unconditional steady-state probability that a data interval contains k packets, and $E(k)$ and $E(k^2)$ be the corresponding first two moments. Then we have by Bayes' rule

$$\lim_{i \rightarrow \infty} P(\text{The data interval of arrival of packet } i \text{ contains } k \text{ packets}) = \frac{kP(k)}{E(k)}.$$

Combining the preceding equations we have

$$a = \sum_{k=1}^N \frac{kP(k)}{E(k)} a_k = \sum_{k=1}^N \frac{P(k)k(k-1)}{2E(k)N} = \frac{E(k^2)}{2NE(k)} - \frac{1}{2N}.$$

We have already shown as part of the analysis of case a) above that

$$E(k) = \frac{\lambda N \bar{V}}{1 - \rho} \quad (9)$$

so there remains to estimate $E(k^2)$. We have

$$E(k^2) = \sum_{k=1}^N k^2 P(k).$$

If we maximize the quantity above over the distribution $P(k)$, $k = 0, 1, \dots, N$, subject to the constraints $\sum_{k=0}^N P(k) = 1$, $\sum_{k=0}^N kP(k) = E(k)$, and $P(k) \geq 0$ (a linear programming problem) we find that the maximum is obtained for $P(N) = E(k)/N$, $P(0) = 1 - E(k)/N$, and $P(k) = 0$, $k = 1, 2, \dots, N-1$. Therefore

$$E(k^2) \leq NE(k).$$

Similarly if we minimize $E(k^2)$ subject to the same constraints we find that the minimum is obtained for $P(\hat{k}-1) = \hat{k} - E(k)$, $P(\hat{k}) = 1 - (\hat{k} - E(k))$ and $P(k) = 0$ for $k \neq \hat{k}-1, \hat{k}$, where \hat{k} is the integer for which $\hat{k}-1 \leq E(k) < \hat{k}$. Therefore

$$E(k^2) \geq (\hat{k}-1)^2(\hat{k} - E(k)) + (\hat{k})^2(1 - (\hat{k} - E(k))).$$

After some calculations this relation can also be written

$$E(k^2) \geq E(k) + (\hat{k}-1)(2E(k) - \hat{k}) \quad \text{for } E(k) \in [\hat{k}-1, \hat{k}), \hat{k} = 1, 2, \dots, N.$$

Note that the lower bound above is a piecewise linear function of $E(k)$, and equals $(E(k))^2$ at the breakpoints $\hat{k} = 1, 2, \dots, N$. Summarizing the bounds we have

$$\frac{E(k) + (\hat{k}-1)(2E(k) - \hat{k})}{2NE(k)} - \frac{1}{2N} \leq a \leq \frac{1}{2} - \frac{1}{2N}, \quad (10)$$

3. Analysis of the Dynamic Broadcasting Scheme

where \hat{k} is the positive integer for which

$$\hat{k} - 1 \leq E(k) < \hat{k}.$$

Note that as $E(k)$ approaches its maximum value N (i.e., the system is heavily loaded), the upper and lower bounds coincide. By combining the results for cases a) and b) above we have

$$\lim_{i \rightarrow \infty} E(m_i) = P(A) \lim_{i \rightarrow \infty} E(m_i | A) + P(B) \lim_{i \rightarrow \infty} E(m_i | B) = (1 - \rho) \frac{\lambda \bar{V}}{1 - \rho} + a\rho,$$

or finally

$$\lim_{i \rightarrow \infty} E(m_i) = \lambda \bar{V} + a\rho$$

where a satisfies Eq. (10). Using Eq. (8) and the expressions derived we obtain Eq. (2) where $E(K)$ is given by Eq. (9), and \hat{k} satisfies $\hat{k} - 1 \leq E(k) < \hat{k}$. **Q.E.D.**

Note that the formula for the mean waiting time becomes exact in the limit both as $\rho \rightarrow 0$ (light load), and as $\rho \rightarrow 1 - \lambda \bar{V}$ (heavy load), in which case $E(k) \rightarrow N$ and $a \rightarrow 1/2 - 1/(2N)$. The formula is also exact if $N = 1$ ([BeG87]).

3.2. Main Proof

We now complete the proof of the dynamic broadcasting theorem.

Proof of the Dynamic Broadcasting Theorem: In a PMNB period each node that has a packet to broadcast at the start of the period participates with exactly one packet. Let M be the number of such nodes at the start of a period. Each of these nodes can be viewed as making a reservation during the reservation interval. The duration of the subsequent broadcast interval is at most MX time units.

The dynamic broadcasting scheme will be called system “ b ” (for “broadcast”). We also consider the limited service gated reservation system with shared reservation and data intervals presented in the previous subsection. This system will be called system “ a ” (for “auxiliary”). Let the reservation interval of system “ a ” be constant and equal to V , and the service time of a packet be constant again and equal to X .

Consider the following analogy between systems “ a ” and “ b ”. Let a data interval of system “ a ” correspond to a broadcast interval of system “ b ”, a user of system “ a ” correspond to a node of system “ b ”, and a packet arrival of system “ a ” correspond to a broadcast request arrival of system “ b ”. Note the similarities between the two systems. During a data interval of system “ a ” (or

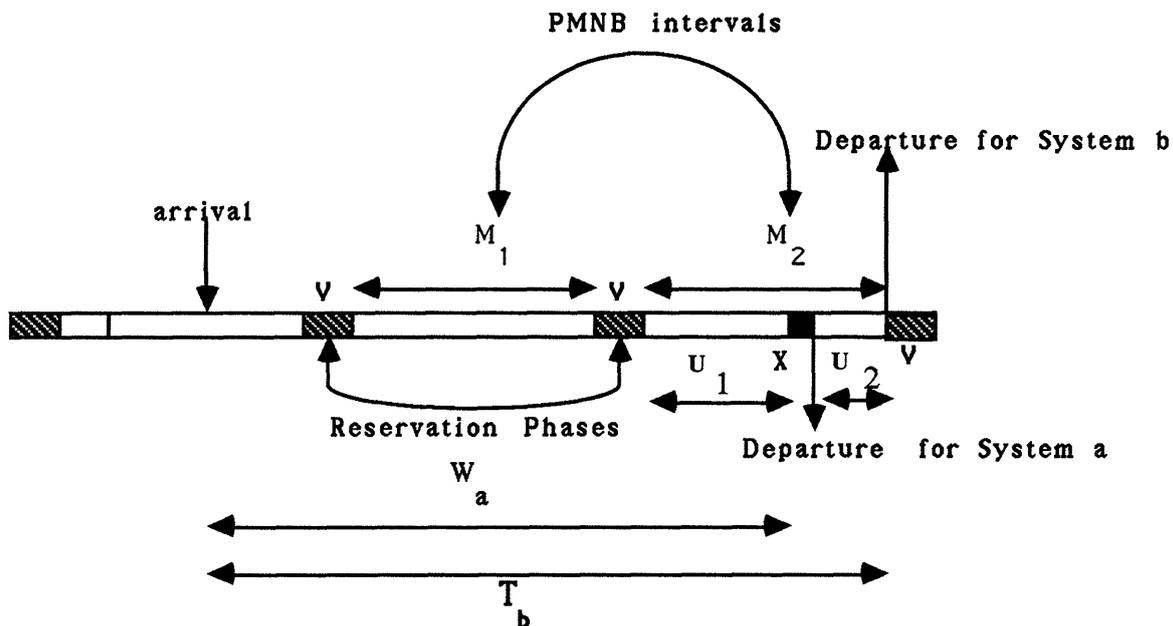


Figure 2: Reservation and broadcast (or data) intervals for the network broadcasting scheme, and the auxiliary queueing system.

broadcast interval of system “*b*”) at most one packet (or broadcast request, respectively) from each user (or node, respectively) can be served. It is easy to see that the probability distributions of the length of the reservation intervals, the data (or broadcast) intervals, and the number of users (or nodes) served in a data interval are identical for both systems. In particular, the length of a reservation interval of both systems is equal to V by construction. The length of a broadcast interval of system “*b*” is equal to MX , where M is the number of active nodes. Similarly, the length of a data interval of system “*a*” is equal to MX , where M is the number of non-empty queues. The only difference between the two systems is that in system “*b*” a broadcast request completes service at the end of a PMNB period, while in system “*a*” packets complete service at times jX , $j = 1, 2, \dots, M$, from the beginning of the data interval.

The waiting time W_a in queue for a packet of the auxiliary system is given from Eq. (2) of Subsection 3.1, with $\bar{X}, \bar{X}^2, \bar{V}, \bar{V}^2$ replaced by X, X^2, V, V^2 , respectively. The average delay (queueing plus service time) for the auxiliary system “*a*” is

$$T_a = W_a + X.$$

Let U_1 be the average time between the beginning of a data interval of system “*a*”, and the time that a packet served in this data interval starts transmission (see Fig. 2). Similarly, let U_2 be the average time between the end of the transmission of a packet of system “*a*” and the end of the data

4. Partial Multinode Broadcast in a Hypercube

interval in which it is served. Then it can be proved by using arguments similar to those used in the proof of Theorem 1 that

$$U_1 = U_2 = aNX \leq \frac{N-1}{2}X. \quad (11)$$

It can also be seen that

$$U_1 = U_2 \leq E(N_i)X = \lambda N W_a X = \rho W_a,$$

where Eq. (4) was used. The average packet delay T_b of the broadcasting scheme is

$$T_b = T_a + U_2 = W_a + X + aNX \leq W_a + X + \min\left(\frac{N-1}{2}X, \rho W_a\right). \quad (12)$$

This completes the proof. **Q.E.D.**

Note that for light load the dynamic broadcasting theorem gives

$$T \leq 1.5V + X, \quad \rho \approx 0.$$

Until now we did not have to assume any particular topology for the multiprocessor network. The dynamic broadcasting scheme and the dynamic broadcasting theorem apply to any network for which a PMNB algorithm with certain properties exists. The next section will present such algorithms for a hypercube network of processors. In [VaB92] we present corresponding PMNB algorithms for d -dimensional meshes. We believe that such algorithms exist for many other regular topologies (in later publications we will present such PMNB algorithms for folded-cubes and meshes of trees).

4. PARTIAL MULTINODE BROADCAST IN A HYPERCUBE

Beginning with this section we focus on a hypercube network of processors. In particular, we consider the partial multinode broadcast problem, where M arbitrary nodes of an N -processor hypercube have to broadcast a packet to all the other nodes. We call these nodes *active nodes*. The PMNB is, as we saw, an important component of the dynamic broadcasting scheme, but it is also an important problem on its own right.

We will say that a communication algorithm is *near-optimal* if the potential loss of optimality with respect to completion time is of strictly smaller order of magnitude than the optimal completion time itself. We generally prove that an algorithm is near-optimal by showing that the leading term of its worst case time complexity (including the corresponding constant factor) is the same with the

4. Partial Multinode Broadcast in a Hypercube

leading term of an expression which is a lower bound to the time required by any algorithm. We generally derive the optimal completion time by deriving a lower bound to the completion time of any algorithm and by constructing an algorithm that attains the lower bound; this latter algorithm is said to be *optimal*. We will say that an algorithm is of *optimal order* if its worst case time complexity is asymptotically within a constant factor from the optimal value.

Let T_{PMNB} be the optimal time required for a partial multinode broadcast in a hypercube. T_{PMNB} may actually depend on which are the M nodes that want to broadcast. A lower bound, however, is always

$$T_{PMNB} \geq \frac{M-1}{d}, \quad (13)$$

where d is the dimension of the hypercube. This can be seen by arguing that each node has to receive $M-1$ or M packets, and has only d input ports. If the splitting of packets is not allowed then a slightly stronger lower bound holds:

$$T_{PMNB} \geq \max \left(d, \left\lceil \frac{M-1}{d} \right\rceil \right),$$

where by $\lceil x \rceil$ we denote the smallest integer which is greater than or equal to x . This is because when packets are not split, the diameter of the network is a lower bound on the broadcast delay.

One way to execute the partial multinode broadcast is to perform a full multinode broadcast (with dummy packets for the nodes which have nothing to broadcast). The optimal completion time of the MNB in an d -dimensional hypercube with $N = 2^d$ nodes, when each packet requires one time unit (or slot) to be transmitted over a link, was found in [BOS91] (see also [BeT89]) to be $\lceil \frac{N-1}{d} \rceil$ time slots. Thus an upper bound for T_{PMNB} is

$$T_{PMNB} \leq \left\lceil \frac{N-1}{d} \right\rceil.$$

When $M \ll N$ the MNB algorithm is inefficient as the gap between the preceding inequality and the lower bound of Eq. (13) suggests. In the three subsections of this section we will provide three communication algorithms to execute the PMNB task in a hypercube. The first algorithm, presented in Subsection 4.1, has time complexity

$$T_{PMNB}^I \leq \left\lceil \frac{M-1}{m} \right\rceil + 2d + 2dt_p - m,$$

in the case $M = 2^m$ for some integer m , where t_p is the time required for a single parallel prefix step ($t_p \leq 1$). If M is not a power of 2 then m should be replaced in the above expression by $\lceil \log M \rceil$ and M by $2^{\lceil \log M \rceil}$. This algorithm is *not* of optimal order [except if $M = \Theta(N^c)$ for some positive constant c , or if $M = \Theta(\log N)$], but it is a simple and practical algorithm, as numerical examples indicate.

4. Partial Multinode Broadcast in a Hypercube

The second and the third algorithms, to be described in Subsections 4.2 and 4.3 respectively execute the PMNB in near-optimal time. In particular the second algorithm, which we call *near-optimal PMNB algorithm*, executes the task in time

$$T_{PMNB}^{II} \leq \frac{N-1}{N} \frac{M}{d} + 2dt_p + 2, \quad (14)$$

independently of the value of M and the location of the active nodes. This is the best existing algorithm for the PMNB task, having roughly half the complexity of the PMNB algorithm given in [Sta91]. Comparing Eq. (14) with the lower bound (13) we see that the leading terms of the right hand sides have the same coefficient. The algorithm assumes that packets can be split at the origin and recombined at the destinations. For the case where this is undesirable we modify the algorithm to achieve near-optimal completion time without the need of splitting and recombining the packets. This gives rise to a third PMNB algorithm that will be presented in Subsection 4.3. We call it *near-optimal PMNB without splitting of packets* and its time complexity can be bounded above by

$$T_{PMNB}^{III} \leq \left\lceil \frac{M}{d} \right\rceil + 2d + 4dt_p - 1,$$

for any M . Note that the MNB is a special case of the PMNB with $M = N$. The latter PMNB algorithm gives rise to a very efficient MNB algorithm with complexity $\lceil N/d \rceil + 2d - 1$, which does not use the splitting of packets. This MNB algorithm has not appeared in the literature before.

The benefits of using a partial multinode broadcast instead of a full multinode broadcast algorithm can be best illustrated by a numerical example.

Example:

We consider both the cases where $t_p=1$ and $t_p = 0$. The case $t_p = 0$ corresponds to the situation where $t_p \ll 1$ (a realistic assumption for computers which have an efficient implementation of the parallel prefix operation), or to the situation where the position of the active nodes is known in advance. Consider a hypercube of $N = 2^{16}$ nodes and a PMNB task in it involving $M = 2^{10}$ active nodes. A full MNB would take 4096 steps. Algorithm I requires 157 time units (or only 125 if $t_p = 0$). The near-optimal algorithm which allows splitting of packets requires 98 time units (or only 66 time units if $t_p = 0$). The near-optimal PMNB algorithm without splitting of packets requires 160 time units (96 if $t_p = 0$). The lower bound for the PMNB in this case is 64.

4.1. A Suboptimal PMNB Algorithm for the Hypercube

This algorithm consists of four phases. (There can be some pipelining between the phases, but we do not try to exploit this because the gain in completion time is small).

4. Partial Multinode Broadcast in a Hypercube

Let s_1, s_2, \dots, s_M , $M \leq N$, be the active nodes. The *rank* of a packet located at node s is defined as

$$r_s = \sum_{t < s} x_t - 1,$$

where x_t is equal to one if processor t has a packet to broadcast, and zero otherwise.

Phase 1 (Rank Computation Phase):

The rank r_s ($0 \leq r_s \leq M - 1$) of each active node s is computed. This can be done in $2d$ steps by performing a *parallel prefix operation* (see [Lei92]) on a tree P , called *parallel prefix tree*, embedded in the hypercube. The i th leaf of the tree from the left is the i th node of the hypercube. The operation is described in [Lei92], pp. 37-44. During each step only links of a particular dimension are used. The packets involved in a parallel prefix operation are small (one byte of information), and require only t_p time units to be transmitted over a link. Thus Phase 1 takes $2dt_p$ time units to be completed. It is reasonable to assume that $t_p \leq 1$, where one time unit is the time required to transmit a whole packet over a link; in fact it is reasonable to expect that in many parallel machines we have $t_p \ll 1$. Note that if the active nodes are known in advance then their ranks are also known and Phase 1 can be omitted.

Phase 2 (Packing Phase):

Processor s sends its packet to processor r_s . This is known in the literature as the *packing problem* (see [Lei92] for the case of a butterfly network), and can be done in d steps by using the following greedy algorithm; a packet during the i th step of the packet phase is transmitted over an i -dimensional link if the i th bit of its routing path is a one, or stays at the current node otherwise. Packets in the packing phase use disjoint paths. This can be seen by noting that when the dimensions of the hypercube are travelled in an ascending order, the hypercube resembles a butterfly, and using the well-known results for the packing problem in butterflies (see, for example, [Lei92], pp. 524-538).

At the end of the second phase the 2^m nodes with the smallest identities have a packet. In the last two phases, each of these packets will be broadcast to all the other processors. Note that the $M = 2^m$ nodes with the smallest identities form a subcube of dimension m , namely, the one obtained by fixing the $d - m$ most significant bits to 0.

Phase 3 (Subcube Single Node Broadcast Phase):

Processor r_s , $0 \leq r_s \leq 2^m - 1$, broadcasts its packet to the $d - m$ dimensional hypercube $(*^{d-m}r_s)$ obtained by fixing the m less significant bits to equal the binary representation of r_s . This is a single node broadcast in a $(d - m)$ -dimensional hypercube, and requires $d - m$ steps.

4. Partial Multinode Broadcast in a Hypercube

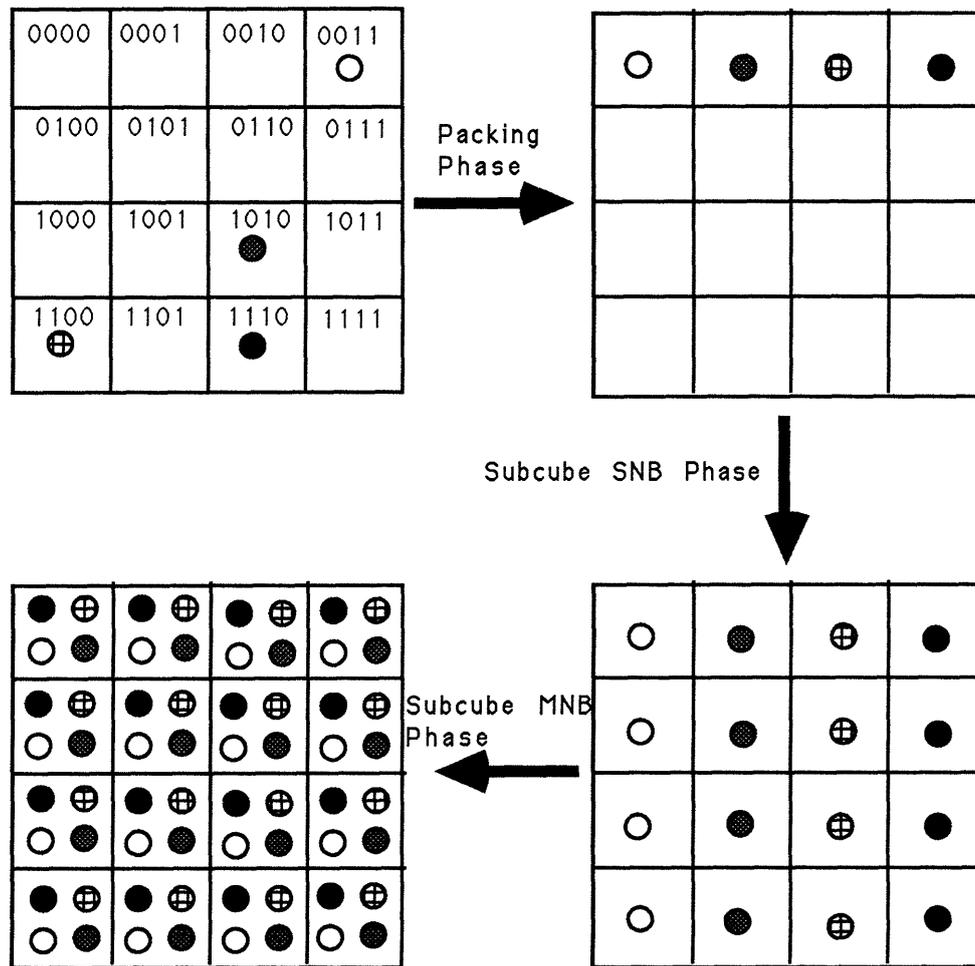


Figure 3: The first (suboptimal) PMNB algorithm for hypercubes ($N = 16$, $M = 4$). Each column or row corresponds to a subcube.

Phase 4 (Subcube Multinode Broadcast Phase):

At the beginning of this phase all processors wr_s , $w = 0, 1, \dots, 2^{d-m}$, have received the packet originating at node s . During Phase 4 processor wr_s broadcasts the packet to all the nodes in the subcube $(w * m)$. This is a full MNB in each one of these m -dimensional disjoint subcubes, and requires time $\lceil \frac{M-1}{m} \rceil$ (see [BeT89], Section 1.3).

Adding up the durations of Phases 1 through 4 we get

$$T_{PMNB}^I \leq \left\lceil \frac{M-1}{m} \right\rceil + 2d + 2dt_p - m.$$

Figure 3 shows how the preceding algorithm works for a 4-dimensional hypercube and $M = 4$ active nodes.

4.2. A Near-optimal PMNB Algorithm with Splitting of Packets

In this subsection we present a near-optimal algorithm to execute the partial multinode broadcast task in a hypercube. We will show that the time required by the algorithm satisfies

$$T_{PMNB}^H \leq \frac{N-1}{N} \frac{M}{d} + 2dt_p + 2,$$

where t_p is the time required for a single parallel prefix step.

The algorithm in this section assumes that packets can be split at the origin and recombined at the destination without any overhead. Each packet requires one time slot in order to be transmitted over a link. If a packet is split in d parts, each of these parts requires $1/d$ time units to be transmitted over a link. In the next subsection, we will present another near-optimal partial multinode broadcast algorithm, which does not require the splitting of packets.

We will start by presenting a suboptimal partial multinode broadcast algorithm. This algorithm will not make full use of the links of a hypercube. We will then modify the algorithm to achieve efficient link utilization and near-optimal completion time. The suboptimal algorithm consists of three phases:

Phase 1 (Rank Computation Phase):

The rank r_s of each active node is computed. This computation is done through a parallel prefix operation as in Phase 1 of the algorithm of the previous subsection. It requires time $2dt_p$, where t_p is the time required for a single parallel prefix step.

Phase 2 (Packing Phase):

The packet of node s and rank r_s is sent to processor r_s . This is done in d steps as described in Phase 2 of the algorithm of the previous subsection.

Phase 3 (Broadcast Phase):

The broadcast phase consists of d subphases $l = 1, 2, \dots, d$. During subphase l , every node $r = r_{d-1}r_{d-2} \dots r_0$ transmits to its $(d-l)$ -neighbor in any order the packets that were located at the node at the beginning of Phase 3 plus the packets that the node has received during all the previous subphases.

During subphase 0 the nodes have (at most) one packet and this is the only one they broadcast. Phase 3 is easy to implement since the current subphase l is easily known.

To prove that the algorithm delivers the packets to all the nodes, it is useful to introduce some new notation. Let $\beta = \beta_{d-1}\beta_{d-2} \dots \beta_0$ be a binary number of length d . We denote by $S_l(\beta) =$

4. Partial Multinode Broadcast in a Hypercube

($*^l \beta_{d-l-1} \beta_{d-l-2} \cdots \beta_0$) the subcube of the nodes whose $d-l$ less significant bits are equal to the $d-l$ less significant bits of β .

The next theorem proves that the previous algorithm actually executes the PMNB task.

Theorem 2: For each $\beta \in \{0, 1\}^d$, at the end of subphase l of Phase 3, $l = 1, 2, \dots, d$, each node in subcube $S_l(\beta)$ has received a copy of every packet located at the beginning of Phase 3 at some node in $S_l(\beta)$, completing a PMNB within each of these subcubes.

Proof: The proof will be done by induction on l . For $l = 0$ (i.e., at the beginning of Phase 3 of the algorithm) it holds trivially since every node has its own (if any) packet. Assume it is true for some l . Every subcube $S_l(\beta)$ is composed of the two subcubes $S_{l-1}(\beta_{d-1} \cdots \beta_{d-l+1} 0 \beta_{d-l-1} \cdots \beta_0)$ and $S_{l-1}(\beta_{d-1} \cdots \beta_{d-l+1} 1 \beta_{d-l-1} \cdots \beta_0)$. During subphase l every node in one of these subcubes sends to its $(d-l)$ -neighbor all the packets it has received during the previous subphases, together with its own packet. This combined with the induction hypothesis proves the theorem. **Q.E.D.**

Letting $l = d$ we find that at the end of subphase d each packet has been received by all the nodes, and therefore, the PMNB has been completed.

The next lemma calculates the time complexity of Phase 3.

Lemma 1: Phase 3 of the algorithm requires at most

$$\frac{N-1}{N} M + d$$

steps.

Proof: We denote by T_l the duration of subphase l , and we let $m = \lceil \log M \rceil$. At the beginning of Phase 3 only nodes $0, 1, \dots, M-1$ have a packet. From Theorem 2 we know that just before the beginning of phase l , node $s = s_{d-1} s_{d-2} \cdots s_0$ has received all the packets originally located at nodes in the subcube ($*^{l-1} s_{d-l} s_{d-l-1} \cdots s_0$). The number of these packets is equal to the cardinality of the set

$$\mathcal{W}_l(s) = \{w = w_{d-1} w_{d-2} \cdots w_0 \mid 0 \leq w \leq M-1, w_{d-l} = s_{d-l}, w_{d-l-1} = s_{d-l-1}, \dots, w_0 = s_0\}.$$

During subphase l , node s will send these packets to its $d-l$ -neighbor. Therefore, we have

$$T_l \leq \max_s |\mathcal{W}_l(s)|$$

where $|\cdot|$ denotes the cardinality of a set. Let $s' = s_{d-l} 2^{d-l} + s_{d-l-1} 2^{d-l-1} + \cdots + s_0$. The cardinality of $\mathcal{W}_l(s)$ is equal to the number of integers between 0 and $M-1-s'$, which are divisible by 2^{d-l+1} .

Thus

$$\max_s |\mathcal{W}_l(s)| \leq \max_s \left\lceil \frac{M-1-s'}{2^{d-l+1}} \right\rceil \leq \left\lceil \frac{M}{2^{d-l+1}} \right\rceil.$$

The total duration of Phase 3 satisfies

$$\begin{aligned} \text{Duration of Phase 3} &= \sum_{l=1}^d T_l \leq \sum_{l=1}^d \left\lceil \frac{M}{2^{d-l+1}} \right\rceil \\ &\leq d + M \sum_{l=1}^d \frac{1}{2^l} \\ &= d + M \left(1 - \frac{1}{N}\right). \end{aligned}$$

Q.E.D.

Adding up the duration of Phases 1, 2 and 3 we obtain the following lemma:

Lemma 2: The partial multinode broadcast task can be executed in a d -dimensional hypercube in

$$T_{PMNB} \leq M \frac{N-1}{N} + 2dt_p + 2d$$

time units, where M is the number of active nodes.

The PMNB algorithm that we described so far is not of optimal order as the gap between the lower bound (13), and the result of Lemma 2 indicates. In fact, it is suboptimal by a factor of roughly d . This is due to the fact that at each step only links of a particular dimension are used. In the next theorem we modify the algorithm so that all dimensions are used at the same time, and near-optimal completion time is achieved.

Theorem 3: The partial multinode broadcast task in a d -dimensional hypercube can be executed in

$$T_{PMNB} = \frac{M}{d} \frac{N-1}{N} + V_h \tag{15}$$

time units, where M is the number of active nodes, and

$$V_h = 2dt_p + 2$$

is the time required for the first two phases.

Proof: We call the PMNB algorithm analyzed in Lemmas 1 and 2 algorithm \mathcal{A}_0 . At each step of Phases 1, 2, and 3 of \mathcal{A}_0 , only links of a particular dimension are used. Indeed, in the parallel prefix or the packing phase only links of a particular dimension are used at each step. Similarly, during subphase l of the broadcast phase only links of dimension $d-l$ are used.

For any c , consider now another PMNB algorithm referred to as algorithm \mathcal{A}_c . According to \mathcal{A}_c a packet is transmitted over the link of dimension $(l+c) \bmod d$ of its current location, whenever the same packet would be transmitted under the \mathcal{A}_0 algorithm over the l th-dimensional link of its

current location. Since \mathcal{A}_c is identical to \mathcal{A}_0 after appropriately renaming the hypercube dimensions (and the nodes), and since \mathcal{A}_0 performs the PMNB independently of the location of the M active nodes, we conclude that \mathcal{A}_c also executes the PMNB task, and requires the same amount of time as \mathcal{A}_0 .

Using simultaneously all the algorithms $\mathcal{A}_0, \mathcal{A}_1, \dots, \mathcal{A}_{d-1}$ we can find a new algorithm which requires the amount of time claimed in the theorem. In particular, each packet is split into d parts, called *mini packets*. Each mini packet is assigned a distinct integer c between 0 and $d - 1$, called *class*. The mini packets of class c are routed according to algorithm \mathcal{A}_c . Packets of different classes use different hypercube dimensions at any time. According to our communication model, a mini packet requires $1/d$ time units for transmission over a link. The duration of the packing and the broadcast phase is thus reduced by a factor of d , while the duration of the parallel prefix phase remains the same. Therefore, the theorem follows from Lemma 2. **Q.E.D.**

The scalar V_h in Eq. (15) is growing linearly with the dimension d . In practice, however, $2dt_p$ is small, since t_p is very small. Indeed, at each step of a parallel prefix operation only one byte has to be transmitted between neighbors. Some parallel computers, such as the Connection Machine model CM-2 of Thinking Machines Corporation, the IBM/RP-3, and the NYU Supercomputer, have very efficient implementations of the parallel prefix, otherwise called “scan” operation ([Ble86]). Theoretically, however, the parallel prefix operation takes time proportional to the diameter.

No upper ceilings are needed in Eq. (15), since we allow fragmented slots. Note also that under the communication model used in this section a single multinode broadcast requires 2 time units, the same time that would be required if cut-through routing ([KeK79]) was used. A near-optimal PMNB algorithm that does not require the splitting of packets is presented in the next subsection.

4.3. A Near-optimal Hypercube PMNB Algorithm without Splitting of Packets

In this subsection we modify the algorithm of the preceding section in order to avoid the potential drawbacks of packet splitting. This is done at the expense of a slight increase in the complexity. Messages in this section require one time slot in order to be transmitted over a link, and are always transmitted as one packet.

The algorithm consists of two parts.

Class Computation Part:

The rank r_s , $0 \leq r_s \leq M - 1$, $s \in \{s_1, s_2, \dots, s_M\}$, of each packet is computed through a parallel

5. Performance of the Dynamic Broadcasting Scheme for Hypercubes

prefix operation. This requires $2dt_p$ time units. The packet of node s is assigned a *class number* $c = r_s \bmod d$.

Main Part:

The packets of class c are routed according to algorithm \mathcal{A}_c described in the proof of Theorem 3. Recall that algorithm \mathcal{A}_c consists of three phases: the rank computation, the packing, and the broadcast phase. Only packets of class c take part in the rank computation phase, or in any other phase of \mathcal{A}_c .

Each class has at most $\lceil M/d \rceil$ packets. Using Lemma 2 with $\lceil M/d \rceil$ instead of M , we get that the main part of the previous algorithm requires time less than $\lceil \frac{M}{d} \rceil + 2d + 2dt_p - 1$ (we have taken into account that the duration of the main part, excluding the parallel prefix operation, is an integer). Thus the total duration of the algorithm satisfies

$$T_{PMNB}^{III} \leq \left\lceil \frac{M}{d} \right\rceil + 2d + 4dt_p - 1.$$

The algorithm just presented for the PMNB task gives rise to an efficient algorithm for the MNB task. Indeed, a multinode broadcast can be treated as a partial multinode broadcast with $M = N$. The class computation part and the rank computation phase are not necessary any more, since the class number and the rank of each packet are known in advance. The packing and the broadcast phases alone can execute the MNB in time less than or equal to

$$\left\lceil \frac{N}{d} \right\rceil + 2d - 1,$$

which is optimal within $2d - 1$ time units. This MNB algorithm is apparently new.

5. PERFORMANCE OF THE DYNAMIC BROADCASTING SCHEME FOR HYPERCUBES

In this section we evaluate the average packet delay of the hypercube dynamic broadcasting scheme. It is easy to see that the PMNB algorithms of Subsections 4.2 and 4.3 both satisfy the conditions of the dynamic broadcasting theorem. We assume that one of these two algorithms is used as a component of the dynamic scheme, and we apply the dynamic broadcasting theorem.

The reservation interval of the hypercube PMNB algorithm of Subsection 4.2 consists of the rank computation and the packing phases, and has duration $V = 2dt_p + 2$. If the near-optimal PMNB algorithm of Subsection 4.3 that does not allow the splitting of packets is used then the reservation interval consists of the class computation part, and the rank computation and packing phases of

5. Performance of the Dynamic Broadcasting Scheme for Hypercubes

the main part, and has duration $2d + 4dt_p$. The analysis can be carried out for both cases. In what follows we will give the results for the case where the PMNB algorithm of Subsection 4.2 is used; if the algorithm of Subsection 4.3 is used, V should be replaced by $2d + 4dt_p$ throughout this section. During the reservation interval every node s that has a packet to broadcast makes a reservation for the broadcast interval by setting $x_s = 1$ in the parallel prefix phase of Subsection 4.2 (and in the class computation phase if the algorithm of Subsection 4.3 is used). In addition to that, during the packing phase the packets move to more favorable intermediate destinations. Following each reservation interval, there is a broadcast interval, which is the last phase of the algorithms of Subsection 4.2 (or Subsection 4.3), and has duration

$$\frac{M}{d} \frac{N-1}{N}$$

time units. Each node can participate with at most one packet per PMNB period, and this packet must have arrived prior to the beginning of that PMNB period.

We define the *hypercube utilization factor* as

$$\rho = \frac{\lambda(N-1)}{d}. \quad (16)$$

For a given load, ρ is equal to the ratio of the average total number of transmissions per unit of time necessary to execute the broadcasts (each broadcast requires $N - 1$ transmissions), over the total number of links of the hypercube. To find a necessary condition for stability for any broadcasting scheme, note that λN broadcast requests are generated on the average per unit of time in the hypercube, each requiring at least $N - 1$ transmissions to be completed. Since there are dN links, a necessary condition for stability is

$$\lambda N(N-1) \leq dN,$$

or else

$$\rho < 1.$$

As we will see, our scheme does not guarantee stability for any $\rho < 1$, but it does for ρ very close to 1.

The PMNB algorithm of Subsection 4.2 satisfies the conditions required by the dynamic broadcasting theorem to apply. Using the dynamic broadcasting theorem, and substituting X by $(N-1)/(dN)$, we see that the average delay T_h of the hypercube dynamic broadcasting scheme using the PMNB algorithm of Subsection 4.2 satisfies

$$T_h = W + \frac{\alpha(N-1)}{d} + \frac{N-1}{N} \frac{1}{d} \leq W + \frac{1}{d} + \min\left(\rho W, \frac{N-1}{2d}\right), \quad (17)$$

5. Performance of the Dynamic Broadcasting Scheme for Hypercubes

with

$$\begin{aligned} W &= \frac{\rho}{2d(1-\rho-\lambda V)} \frac{N-1}{N} + \frac{(1-\rho)V}{2(1-\rho-\lambda V)} + \frac{(1-\rho a - \lambda V)V}{1-\rho-\lambda V} = \\ &= \frac{N-1}{N} \frac{\rho}{2d(1-\rho-\lambda V)} + \frac{V(1.5-0.5\rho-\rho a - \lambda V)}{1-\rho-\lambda V}, \end{aligned} \quad (18)$$

where

$$V = 2dt_p + 2,$$

ρ is given by Eq. (16), and

$$\begin{aligned} \frac{\bar{M} + (\hat{M} - 1)(2\bar{M} - \hat{M})}{2N\bar{M}} - \frac{1}{2N} \leq a \leq \frac{1}{2} - \frac{1}{2N}, \\ \bar{M} = \frac{\rho V d}{1-\rho}, \end{aligned}$$

where \hat{M} is the smallest integer which is larger than \bar{M} . (\bar{M} is the average number of broadcast requests served in a PMNB period).

For any fixed load which satisfies $1-\rho-\lambda V > 0$, we find from Eqs. (17) and (18) that the average packet delay is $\Theta(V) = \Theta(d \cdot t_p)$. For an almost empty hypercube, $\rho \approx 0$, Eqs. (17) and (18) give

$$T_h \leq 1.5V + \frac{1}{d} = 3dt_p + 3 + \frac{1}{d}, \quad \rho \approx 0.$$

The following theorem gives the order of magnitude of the average delay of the scheme at heavy load.

Theorem 4: In the limit, as $\rho \rightarrow 1 - \lambda V$, we have

$$T_h = O\left(\frac{V}{1-\rho-\rho dV/(N-1)}\right).$$

Proof: Let $\rho = 1 - \lambda V - \epsilon$ with $\epsilon \rightarrow 0$. Then, as shown in Subsection 3.1, $a \rightarrow 0.5 - 0.5/N$. Let $a = 0.5 - 0.5/N - \delta$ with $\delta \rightarrow 0$. Then

$$\begin{aligned} T_h &\leq \frac{\rho}{2d\epsilon} + \frac{V(0.5 + \epsilon + 0.5\rho/N + \rho\delta)}{\epsilon} + \frac{1}{d} + \min\left(\frac{V(0.5 + \epsilon + 0.5\rho/N + \rho\delta)}{\epsilon}, \frac{N-1}{2d}\right) \\ &= O\left(\frac{V}{\epsilon} + \frac{\rho}{d\epsilon}\right). \end{aligned}$$

Q.E.D.

As proved in [Sta91], the average delay of any dynamic broadcasting scheme grows at least as $\Omega((1-\rho)^{-1})$. Since the term $\rho dV/(N-1)$ goes to 0 (very fast) as $N \rightarrow \infty$, our dynamic broadcasting

scheme has good behavior for heavy load (ρ close to 1) when the number of nodes of the hypercube is large.

The hypercube broadcasting scheme is stable for

$$\rho \leq \frac{1}{1 + (2dt_p + 2)d/(N - 1)},$$

which is very close to the maximum ρ that can be accommodated by *any* scheme (which is the unit), and tends to it as $N \rightarrow \infty$. The reason we get this remarkable result is the efficiency of the PMNB algorithm of Subsection 4.2, and the small overhead introduced by the reservation intervals (parallel prefix and packing phases). A similar result would be true if the algorithm of Subsection 4.3 were used; the corresponding stability region is given by

$$\rho \leq \frac{1}{1 + (2d + 2dt_p)d/N}.$$

REFERENCES

- [BeG87] Bertsekas, D. P., and Gallager R., *Data Networks*, Prentice-Hall, 1987.
- [BeG91] Bertsekas, D. P., and Gallager R., unpublished research.
- [BOS91] Bertsekas, D. P., Ozveren, C., Stamoulis, G. D., Tseng, P., and Tsitsiklis, J. N., "Optimal Communication Algorithms for Hypercubes," *J. Parallel Distrib. Comput.*, Vol. 11, pp. 263-275.
- [BeT89] Bertsekas, D. P., and Tsitsiklis, J. N., *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, Englewood Cliffs, N.J., 1989.
- [Ble86] Blelloch, G. E., "Scans as Primitive Parallel Operations," *Proc. Int'l Conf. Parallel Processing*, pp. 355-362, August 1986.
- [Ho90] Ho, C. T., "Full Bandwidth Communications on Folded Hypercubes," Research Report RJ 7434 (69605), IBM Almaden Research Center, April 1990.
- [Joh89] Johnsson, S. L., and Ho, C. T., "Optimum Broadcasting and Personalized Communication in Hypercubes," *IEEE Trans. Computers*, Vol. C-38, 1989, pp. 1249-1268.
- [KeK79] Kermani, P., and Kleinrock, L., "Virtual Cut-Through: A New Computer Communicating Switching Technique," *Comput. Networks*, Vol. 3, pp. 267-286, 1979.
- [Lei92] Leighton, F. T., *Introduction to Parallel Algorithms and Architectures: Arrays - Trees - Hypercubes*, Morgan Kaufmann, San Mateo, CA, 1992.
- [LEN90] Lan, Y., Esfahanian, A.-H., and Ni, L., "Multicast in Hypercube Multiprocessors," *J. Parallel Distrib. Comput.*, pp. 30-41.

- [RaS90] Ranka, S., and Sahni, S., *Hypercube Algorithms with Applications to Image Processing and Pattern Recognition*, Springer-Verlag, New York, 1990.
- [Sta91] Stamoulis, G. D., "Routing and Performance Evaluation in Interconnection Networks," Ph.D. Thesis, MIT, Report LIDS-TH-2035, May 1991.
- [VaB90] Varvarigos, E. A., and Bertsekas, D. P., "Multinode Broadcast in Hypercubes and Rings with Randomly Distributed Length of Packets," Laboratory for Information and Decision Systems Report LIDS-P-2006, M.I.T., Cambridge, MA, November 1990, to appear in *IEEE Trans. on Paral. and Distrib. Systems*.
- [VaB92] Varvarigos, E. A., and Bertsekas, D. P., "Partial Multinode Broadcast and Partial Exchange in d -dimensional Wraparound Meshes," Laboratory for Information and Decision Systems Report LIDS-P-2087, M.I.T., Cambridge, MA, January 1992.