AN IMPROVEMENT ON THE VALIANT-BREBNER HYPERCUBE DATA BROADCASTING TECHNIQUE

By
NASARUDDIN ZENON

0.0 INTRODUCTION

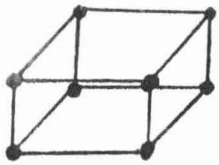
At the end of the writer's earlier article: 'A Model Solution for the Radar Surveillance Problem' [7] it is mentioned that a slight modification of the formulation given in the paper can be used to solve the problem of data broadcasting in multi processor computer system. The intended meaning of broadcasting there is the manner each datum is distributed among the processors in a computer system. However, once the datum is processed by a particular processor it has to be passed to the next processor/ s for further processing or to be combined with the whole data array (the broadcast procedure split the data set).

This paper is motivated by a multi node broadcast technique for the hypercube architecture developed in 1982 by Valiant and Brebner [6] which the author feels can be further improved in terms of speed and performance. The writer tries to improve this algorithm because it is the only known algorithm for the hypercube machine that has the probability of more than i (log n) processors will simultaneously try to transmit a message through a given processor decreases exponentially with i. (Please refer to [4] for detail.)

The arrangement of this paper is as follows. In section 1.0 a description of the hypercube topological characteristics will be given which can be used to modify the algorithm. Section 2.0 provides the description of the Valiant and Brebner (V-B) algorithm. In Section 3.0 we will propose a modification to the V-B algorithm. An analysis of the improved algorithm in comparison to te V-B algorithm is provided at the end of section 3.0.

1.0 THE HYPERCUBE TOPOLOGY

The hypercube connection scheme is the interconnection of standard procesors into a parallel architecture. Any computer system which employs the hypercube architecture is usually called a hypercube machine [1]. A d-dimensional hypercube (d-cube) machine consists of $p = 2^d$ processors. The addresses of the processors are binary numbers in the range of 0 to 2^d-1 with each address containing d bits. The d neighbours of P_i ($o \leq i \leq 2^d-1$) are defined as follows: Processor P_j ($0 \leq j \leq 2^d-1$, and $j \neq i$) is connected to processor P_i if and only if the binary representation of j is a single bit complement of i (i.e. i differs from j by exactly 1 bit). Two examples of the hypercube are given in Figure 1 below.



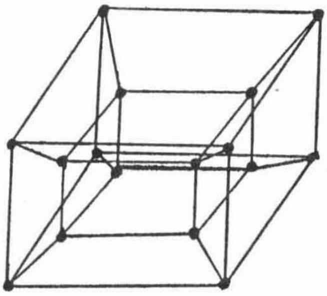


Figure 1 (a) a 3-cube (b) a 4-cube

Note: Circular dots represent the processors and straight lines represent the interconnection circuitry.

The hypercube architecture is usually implemented in Single - Instruction-Multiple-Data (SIMD) computers such as MIT/TMI Connection Machine (CM-1), NCUBE and FPS (further reference in [1]). However, recently Intel has begun to use the hypercube connection in its Multiple-Instruction-Multiple-Data (MIMD) computers by connecting different kinds of processors as the hypercube constituents.

The two topological structures of the hypercube that are needed in this paper are as follows:

(a) The maximum distance between any pair of nodes or the *diameter* is d (= log p)
(b) The node-connectivity which is a measure of the number of independent paths connecting a pair of nodes is equal to d.

The values given in (a) and (b) follows immediately from the connection scheme used for the hypercube.

Bertsekas and Tsitsiklis [2] pointed out that for a network of diameter d, the time for a packet to travel between two nodes is *O*(d) (where *O* is the big Oh notation indicating a linear relationship between time and diameter d), assuming no queuing dealy at the links. But, as we shall see later the highly interconnection scheme of the hypercube presents the problem of path conflicts between packets which results in communication delay. While this problem is not so apparent in a single node broadcast (the sending of the same packet from a given processor to every other processors), communication delay is the main obstacle to be overcome in designing a multi node broadcast algorithm.

The multi node broadcast problem is a routing problem defined as follows:

Each processor P_i on the hypercube is to send a message m_i to processor Pj ($0 \leq i,j \leq 2^d - 1$ and $i \neq j$) All messages have to be transmitted concurrently. Find an optimum routing scheme that will route all of the messages quickly under the condition that each processor can deliver only one message at a time to any number of processors.

The difficulty involved in the above defined problem for the hypercube is that some links may belong to several spanning trees routed at each P_i. This results in conflicts arising from several packets arriving simultaneously at a node. To overcome these conflicts some of the packets must be buffered, thus causing a delay in transmission. Consequently, the multi node problem also involved the minimization of conflicts at each node and the minimization of the buffer space requirements.

2.0 THE VALIANT-BREBNER (V-B) ROUTING ALGORITHM

Valiant and Brebner (1981) give a probabilistic routing algorithm for the hypercube which is a randomized version of the *greedy algorithm* [8]. This algorithm attains the O (p), where p is the number of processors, lower-bound on transmission with a probability arbitrarily close to 1 (as a function of p) [4].

The idea behind V-B algorithm is to divide the broadcasting step into two phases. In the first phase the initial distribution of the messages is randomized among an intermediate processor, chosen uniformly (independently of the destination) among all other processors. Thereafter, the messages are routed to their destination using a deterministic greedy procedure. The technique as summarized by McHugh [4] is as follows:

(1) Each processor containing a message m to be routed, randomly generates an intermediate destination address r(m) for m
(2) The greedy algorithm is used to route each message to its random intermediate target.
(3) Once a message m arrives at r(m), we route m to its original destination T(m) using the deterministic greedy method.

The following theorem summarized the performance of the algorithm

V-B d-cube Routing Theorem [5]

For the d-cube the probability that V-B routing takes more than 8d steps is less than $O(0.74^d)$.

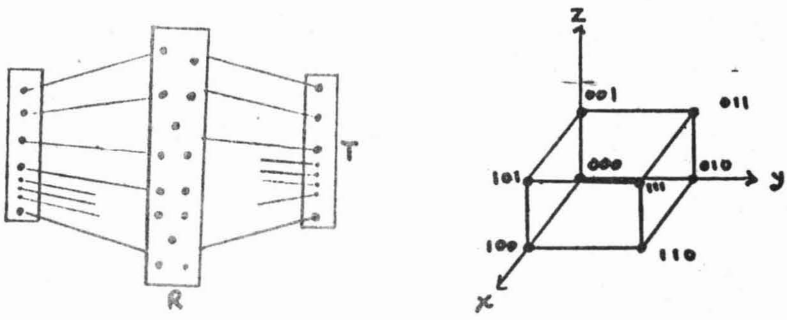
(Please refer [5] for the proof.)

One drawback of the algorithm is that, a subset of processors may try to receive too many messages at one time, thus resulting in a deadlock. The algorithm may have to be restarted in order to change the random value previously assigned for each intermediate processor. However, since the choice of which bit to complement in determining the next processor address is made locally, we can modify the algorithm so that the initial and the final distributions of the messages will follow a fixed bit stream pattern.

In the next section the writer will show that this algorithm can be improved by minimizing the use of randomization and by avoiding the use of greedy algorithm whenever the messages can be transmitted in a deterministic manner.

3.0 AN IMPROVED ALGORITHM

As mentioned earlier, the rich connectivity of the hypercube architecture can be fully exploited in order to construct a routing technique with the number of data collisions a minimal. The key idea is to divided the processors in the hypercube into d sets. Set S consists of processors with messages to be sent to set T, the destination proeessors. Set R consists of processors which are not in S or T (i.e. the intermediate processors). (Please refer to Figure 2 (a))



(a) (b)

Figure 2.

All paths from S to R require only links that are incident with processors in S and in R. Likewise, each processor in R is separated from each of those in T by at most one link. Threfore, we can assign a fixed path pattern for messages travelling from S to R and from R to T. However in travelling from one processor in R to another which is also in R, a message is transmitted randomly because it usually requires more than one link. Nevertheless, we can restrick the random assignments to processors which are in R only.

The effect of the above modification is that we reduce the probability that a message will have to travel along paths which are not the shortest paths between any two processors in R. In contrast, the previous V-B technique may result in a message having to travel along a path that is unnecessarily long because the intermediate processor is chosen at random independently of the destination. Figure 2.0 (b) is used to describe the new technique for the 3-cube.

In phase I of the algorithm let the processors in the xy-plane send messages along the positive z-axis, processors in xz-plane send messages along the positive y-axis and processors in yz-plane send messages along the positive direction of the x-axis. Although all the above operations are done concurrently, path conflicts do not occur because messages do not cross path with each other. The next step is to reverse all the paths made previously.

Phase I allows all messages to be broadcasted from any processor P_i to its nearest neighbours (processors which are adjacent to P_i). At the same time the target processors in phase I are given addresses I (m_i). In phase 2, processors $I(m_i)$ will generate random addresses of processors whose distance are two or more edges away from the originators of messages m_i. While routing m_i from addresses $I(m_i)$ to the random processors $r(m_i)$, path conflicts are resolved by following the V-B technique which is by giving incoming messages m_i higher priority access to network linkages over messages that have already reached $r(m_i)$.

In the last step of phase 2 we repeat the steps in phase I to send messages from $r(m_i)$ to $F(m_i)$, the final destination of m_i. The technique is summarized below.

Phase 1:

For processors that a adjacent to each other, messages are routed in parallel according to the direction of the arrows.

(1) P(0,y,z,)—> P(1,y,z)
P(x,0,z) —> P(x,1,z)
P(x,y,0) —> P(x,y,1)

(2) P(1,y,z)—> P(0,y,z)
P(x,1,z)—> P(x,0,z)
P(x,y,1)—> P(x,y,0)

Phase 2:

For each $P(m_i) \in S$ and $I(m_i) \in R$,

(1) $P(m_i) \longrightarrow I(m_i)$ (follow scheme in phase 1)
(2) $I(m_i)$ generate $r(m_i) \in R$ (locally)
(3) $r(m_i) \longrightarrow F(m_i) \in T$ (follow scheme in phase 1)

Note: (i) x,y and z assume the binary values of 0 and 1.

(ii) Statements in each step of phase 2 are executed in parallel.

One advantage of the above technique is that we have avoided the use of greedy algorithm in sending messages in phase 1 and in the last step of phase 2. As pointed out by McHugh [4] the problem with the greedy algorithm is that messages can be greatly delayed due to access contention at nodes along their routes.

To prove the correctness of the technique we only need to observe that there is a bipartite matching between S and T. By carefully mapping the hypercube into the n-dimensional Cartesian plane we will see that the addresses of the most distant processors from the sources of m_i are the bits complement in every dimension. Thereafter, we just trace the Hasse diagram from processors in S to those in T. As for the movements of messages in R, the correctness of the technique follows immediately from the correctness of the V-B technique.

REFERENCE

[1] *Almasi, G. S., and A. Gottlieb, Highly Parallel Computing, Benjamin/Cummings, Redwood City, CA, 1989.*

[2] *Bertsekas, D.P., and J. Tsitsiklis, Parallel And Distributed Computation, Prentice-Hall Inc., Englewood Cliffs, NJ, 1989.*

[3] *Akl, S.G., The Design Of Parallel Algorithms, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1989*

[4] *McHugh, J.A., Algorithmic Graph Theory, Prentice-Hall, Inc., Englewood Clliffs, NJ, 1990.*

[5] *Valiant, L. G., "A Scheme for fast parallel communication," SIAM Journal on Computing, 11 (May 1982), pp. 350-361.*

[6] *Valiant, L.G., and G.J. Brebner, "Universal schemes for parallel communication, "13th Annual ACM Symposium on Theory of Computing, Milwaukee (May 1981), pp. 263-277.*

[7] *Zenon, N., and R. Ali, "A model solution for the radar surveillance problem," GADING, J2, B1, Kuantan (JUN 1990), pp. 5-14.*

Penghargaan diberikan kepada En. Che Ram Zainon
kerana telah sudi membantu menyunting artikel ini.

Rasulullah Bersabda;

'Sesungguhnya rezeki itu benar-benar mencari tuannya sebagaimana juga ajalnya.'

(At-Thabrani)

Rasulullah Bersabda;

'kelebihan ilmu (makrifat) mengatasi kelebihan ibadah.'

(At-Thabrani)

Rasulullah Bersabda;

'Ilmu yang sedikit lebih baik dari ibadat yang banyak tanpa ilmu.'

(At-Thabrani)

Rasulullah Bersabda;

'Barang siapa yang melalui satu jalan kerana tujuan mencari ilmu nescaya dimudahkan dia oleh Allah satu jalan ke syurga.'

(Muslim)

Allah berfirman maksudnya;

'Jika kamu bersabar dan bertakwa, maka sesungguhnya yang demikian itu termasuk urusan yang patut diutamakan.'

(Ali-Imran 182)

Sabda Junjungan maksudnya;

'Barang siapa yang memakai pakaian kerana mencari kemegahan di dunia ini nescaya Allah akan memakaikan dia pakaian kehinaan pada hari kiamat dan dinyalakan apa di dalamnya.'

(Ibn-Majah)