JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING 3, 344–351 (1986)

An Asynchronous Algorithm for Scattering Information between the Active Nodes of a Multicomputer System

ZVI DREZNER* AND AMNON BARAK†

Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan 48109

Received April 11, 1985

In this paper we present an asynchronous algorithm for scattering information between the active nodes of a multicomputer system, having a large number of independent computers and workstations that are interconnected by a local area communication network. This algorithm is useful when it is desired to reduce the number of messages and the time delay necessary to transmit information from any node to all the active nodes of the system. The algorithm that we develop is based on one-way messages which are sent by each node, every unit of time, to a randomly selected node. The main advantage of this routing is that it overcomes inactive or faulty machines. We show that for an N node multicomputer in which n nodes are active, it is possible to scatter information to all the active nodes in approximately $(1.693 + 1.414(1 - n/N))\log_2 n$ steps. © 1986 Academic Press, Inc.

1. INTRODUCTION

Consider a multicomputer system which consists of N independent computers and workstations (nodes) that are interconnected by an Ethernet-like local area communication network which allows a direct communication link between any pair of nodes as well as broadcast. To utilize such a system efficiently, each node must have knowledge about some of the other nodes. For example, information about the availability and location of resources, length of queues, and the current load of some nodes allows other nodes to improve their performance by making better scheduling decisions.

Assume that at a given time, a subset of n out of the N nodes is actually active and that one or more nodes possess information which has to be dispersed among all the (active) nodes. A simple algorithm for scattering this information is to have each node send asynchronously one message every unit

* Present address: School of Business and Economics, California State University, Fullerton

† On leave from The Hebrew University of Jerusalem, Israel.

0743-7315/86 $3.00

of time. We assume that the nodes use the same unit of time but are not synchronized. In order to speed up the propagation of new information, each node includes recently arrived information in its next few messages. Note that neither the originating nor the receiving nodes know the source and the final destination of the information; therefore we require that all the nodes use the same algorithm.

There are two parameters for measuring the effectiveness of a scattering algorithm: the number of messages, and the time delay until all the nodes receive the scattered information. For example, in the broadcast mechanism which is supported by many local area networks, the total number of messages is n^2 in each unit of time while the time delay is one unit. Another alternative is to select one node to receive and send all the messages. However, this scheme, like the broadcast mechanism, is nonscalable due to limited capacity of a single node.

In this paper we propose an asynchronous algorithm for efficient scattering of information between the nodes of a multicomputer. This algorithm has the following characteristics:

1. Dynamic: it overcomes changes in the availability of the node machines.
2. Decentralized: all the nodes use the same algorithm and there is no central control.
3. Message broadcasts are not used.
4. Low communication overhead: one message is sent by each node in each unit of time.
5. No synchronization is assumed.

Let C denote a measure for the complexity of the scattering algorithm. Assume that all the nodes use (asynchronously) the same unit of time. Then we can define

$$C = ST,$$

where S is the total number of messages sent each unit of time, and T is the number of units of time necessary to scatter a given information to all the nodes. For example, when using broadcasts, $T = 1$ and $C = n^2$.

When all the nodes are active, the complexity of scattering a message in a network, using a ring topology, is $C = O(N^2)$. This follows from the fact that the network diameter (maximal distance between any pair of nodes, where the distance is the number of steps in the minimal path connecting the nodes) is half the number of nodes. When using message routing along trivalent (cubic) graphs in which each node has at most three neighbors, $S = 3N$. For example, the diameter of the Cube-Connected Cycles network [3] is $5/2 \log N + O(1)$, thus $C \approx 15N/2 \log N$ (where log denotes the base 2 logarithm). An improvement of this result can be obtained by using a

message routing along the family of dense trivalent graphs that are discussed by Leland and Solomon in [2]. In these cases the diameter of the network is $3/2 \log N + O(1)$, thus $C \approx 9N/2 \log N$. Further special cases of high density graphs for processor interconnection are discussed by Leland *et al.* in [1]. As pointed out in [2], the diameter of these graphs is bounded by $1.1 \log N$, therefore the complexity of the corresponding scattering algorithm is $C = 3.3N \log N$. Another possible topology is to connect each node to $\log N$ nodes. Say, node j is connected to node $(j + 2^{i-1}) \bmod N$, for $i = 1, \ldots, \log N$. When each node sends $\log N$ messages in each unit of time then $C = N \log^2 N$. Note that if synchronization is assumed then each node may send only one message each unit of time, and C can be reduced to $N \log N$. An interesting open problem is to find an asynchronous interconnection scheme for which C is minimized.

In the above cases, the information routing is deterministic because each node sends and receives the information from a predetermined set of nodes. The main drawback of a deterministic routing is that it does not respond to inactive or faulty nodes. It may result in the isolation of subsets of nodes. An alternative approach which overcomes this difficulty is to use a non-deterministic routing scheme by which each node sends its information to a randomly selected node. Such a scheme adapts itself to changes in the availability of nodes and does not require synchronization. In this paper we show that for large values of N and n, the complexity of this probabilistic scattering algorithm is approximately

$$\frac{1 + N/n \ln(1 + n/N)}{\log(1 + n/N)} n \log n.$$

In particular, for $N = n$, the complexity of the algorithm is $1.693\, n \log n$, which is better than all the above asynchronous deterministic algorithms.

2. Probabilistic Information Scattering

Suppose that the nodes of the multicomputer are numbered $1, 2, \ldots, N$ and that n nodes are actually active. Assume that during every unit of time, each active node selects another node (whether active or not) at random and sends it a message (color). Initially, one node is colored. We find the time delay, T (measured in units of time from the time this node sends the first message), necessary for the color to propagate to all the other active nodes with a given probability. Note: each node that has the color includes it in its future messages. However, since there is no synchronization between the nodes, we assume that each node sends the color only if the color was available to the node at the beginning of its current unit of time. Thus we give a worst case analysis.

LEMMA 1. *Suppose that we independently draw l random integers in $[1, m]$. Let $Q(m, l)$ denote the probability that each of the m integers is drawn at least once. Then for $l \geq m > 1$;*

$$Q(m, l) = \sum_{i=1}^{l-m+1} \binom{l}{i} \left(\frac{1}{m}\right)^i \left(1 - \frac{1}{m}\right)^{l-i} Q(m-1, l-i). \tag{2.1}$$

Proof. The probability that a specific integer is drawn i times is

$$\binom{l}{i} \left(\frac{1}{m}\right)^i \left(1 - \frac{1}{m}\right)^{l-i}.$$

Under the conditions of the lemma this integer must have been drawn at least once. If this integer was drawn i times, then the remaining $m - 1$ integers are drawn $l - i$ times. The probability that each of these $m - 1$ integers is drawn at least once is $Q(m - 1, l - i)$ and the lemma follows. ■

Note that since $Q(1, l) = 1$, $Q(m, l)$ can recursively be calculated by (2.1) for every $l \geq m > 1$.

Let $P(j, k)$ be the probability that exactly k active nodes are colored after j iterations (units of time).

LEMMA 2. *Given that at the beginning of iteration j exactly k nodes are colored. Then*

$$P(j + 1, k + m) = \binom{n - k}{m} R(m, k),$$

where

$$R(m, k) = \sum_{l=m}^{k} \binom{k}{l} \left(\frac{m}{N-1}\right)^l \left(\frac{N - n + k - 1}{N - 1}\right)^{k-l} Q(m, l). \tag{2.2}$$

Proof. Let C_j be the subset of the k colored nodes at the beginning of iteration j. Suppose that at the beginning of iteration $j + 1$ additional m out of the remaining $n - k$ active nodes are colored. Let $M_j = C_{j+1} - C_j$ be this subset. Note that there are $\binom{n-k}{m}$ possible M_j sets. The probability that a certain node chooses another given node is $1/(N - 1)$. During iteration j, there are k colored messages which are sent from C_j. Some of the messages are received by C_j, some are sent to the $N - n$ nonactive nodes, and the remaining are received by M_j. The probability that l colored messages are received by M_j is

$$\binom{k}{l} \left(\frac{m}{N-1}\right)^l \left(\frac{N - n + k - 1}{N - 1}\right)^{k-l}.$$

Therefore, the probability $R(m, k)$ that M_j is colored is given by (2.2) and the probability that exactly $k + m$ nodes are colored after the iteration is

$$P(j + 1, k + m) = \binom{n - k}{m} R(m, k). \quad \blacksquare$$

The next theorem establishes the recursive relationship between the probabilities of two consecutive iterations.

THEOREM 1. *Let v be the largest integer not greater than $i/2$. Then*

$$P(j + 1, i) = \sum_{m=0}^{v} \binom{n - i + m}{m} R(m, i - m) P(j, i - m).$$

Proof. To get i colored nodes at the end of iteration j we must have at least $i/2$ colored nodes at the beginning of the iteration. The result is obtained by adding the probabilities of having $k = i - m$ colored nodes at the beginning of the iteration, each multiplied by the corresponding probability from Lemma 2 of adding m colored nodes. $\blacksquare$

In Table I we give sample values of the probabilities $P(j, n)$ for the case $N = n$.

TABLE I
SAMPLE VALUES OF $P(j, n)$ FOR $N = n$

j	$n = 4$	$n = 8$	$n = 16$	$n = 32$	$n = 64$	$n = 128$
2	0.2222	0.0	0.0	0.0	0.0	0.0
3	0.7160	0.0061	.	.	.	.
4	0.9099	0.2433	.	.	.	.
5	0.9726	0.6158	0.0243	.	.	.
6	0.9918	0.8443	0.2495	0.0002	.	.
7	0.9976	0.9430	0.5934	0.0385	.	.
8	0.9993	0.9800	0.8249	0.2806	0.0009	.
9	0.9998	0.9931	0.9326	0.6167	0.0613	.
10	0.9999	0.9977	0.9753	0.8355	0.3395	0.0029
11	1.0000	0.9992	0.9911	0.9363	0.6657	0.1020
12	.	0.9997	0.9968	0.9763	0.8600	0.4204
13	.	0.9999	0.9989	0.9913	0.9461	0.7240
14	.	1.0000	0.9996	0.9968	0.9799	0.8875
15	.	.	0.9998	0.9988	0.9926	0.9570
16	.	.	0.9999	0.9996	0.9973	0.9840
17	.	.	1.0000	0.9998	0.9990	0.9940
18	.	.	.	0.9999	0.9996	0.9978
19	.	.	.	1.0000	0.9998	0.9991
20	.	.	.	.	0.9999	0.9996
21	.	.	.	.	1.0000	0.9998
22	.	.	.	.	.	0.9999
23	.	.	.	.	.	1.0000

3. Asymptotic Behavior of the Algorithm

We now develop an asymptotic formula for the number of iterations, j, such that with a given probability, all the active nodes are colored. Let $T(\alpha; n, N)$ be the number of iterations needed to get an expected number of $n - \alpha$ colored nodes, $0 < \alpha < 1$, when starting with one colored node.

LEMMA 3. *$T(\alpha; n, N)$ is an upper bound for the number of steps needed to color all the active nodes with a probability of $1 - \alpha$, i.e., $P(T(\alpha; n, N), n) \geq 1 - \alpha$.*

Proof. For $j = T(\alpha; n, N)$,

$$n - \alpha = \sum_{i=1}^{n} iP(j, i) \leq nP(j, n) + (n - 1)(1 - P(j, n))$$
$$= n - 1 + P(j, n).$$

Therefore, $n - \alpha \leq n - 1 + P(j, n)$, which yields $P(j, n) \geq 1 - \alpha$. ■

In the following analysis we assume large values for N and n.

LEMMA 4. *Assume that at the beginning of an iteration there are cn colored nodes, $0 \leq c \leq 1$. Then the expected number of colored nodes after τ iterations $(0 \leq \tau \leq 1)$ is $\hat{c}n$, where*

$$\hat{c} = 1 - e^{-nc\tau/N}(1 - c).$$

Proof. Assume that during the course of the iteration there are xn colored nodes, $c \leq x \leq \hat{c}$. The probability that a node sends a message with the color is c. The probability that during the iteration an uncolored node receives the color is $n(1 - x)/N$. Therefore, the expected increase in the number of colored nodes after each message is $nc(1 - x)/N$. Thus, after one message the expected ratio of colored nodes is $x + c(1 - x)/N$. Assuming that $1/n$ is infinitesimally small, we define $1/n = \Delta t$. Thus

$$\Delta x = n/Nc(1 - x)\Delta t. \tag{3.1}$$

We now integrate (3.1) for t between 0 and τ, and x between c to $\hat{c}$:

$$\frac{N}{nc} \int_c^{\hat{c}} \frac{dx}{1 - x} = \int_0^{\tau} dt.$$

This leads to $\ln(1 - c) - \ln(1 - \hat{c}) = nc\tau/N$, or $1 - \hat{c} = e^{-nc\tau/N}(1 - c)$, and the lemma follows. ■

At the beginning, the algorithm sets $c = c_0 = 1/n$. At the end of iteration j;

$$c_{j+1} = 1 - e^{-nc_j/N}(1 - c_j). \tag{3.2}$$

Let $\theta = 1 + n/N$.

LEMMA 5. $T(\alpha; n\theta, N\theta) = T(\alpha; n, N) + 1 + N/n \ln \theta$.

Proof. Let $c_j(n, N)$ be the sequence defined by (3.2) with $c_0(n, N) = 1/n$. Since $c_0(n\theta, N\theta)$ is small, $c_1(n\theta, N\theta) \approx \theta c_0(n\theta, N\theta) = c_0(n, N)$. Therefore, $c_{j+1}(n\theta, N\theta) \approx c_j(n, N)$. Let $\mu = T(\alpha; n, N)$. Then $c_\mu(n, N) = (n - \alpha)/n = 1 - \alpha/n$. Thus, $c_{\mu+1}(n\theta, N\theta) \approx 1 - \alpha/n$. Since $c_{\mu+1}(n\theta, N\theta) \approx 1$, then by Lemma 4, for any $0 \le \tau \le 1$, $c_{\mu+1+\tau}(n\theta, N\theta) \approx 1 - \alpha/ne^{-\tau n/N}$. For $\tau = N/n \ln \theta$ (note that $0 \le \tau \le 1$), $c_{\mu+1+\tau}(n\theta, N\theta) \approx 1 - \alpha/(n\theta)$. Therefore, $T(\alpha; n\theta, N\theta) = \mu + 1 + \tau$, and the lemma follows. ■

As a result of Lemma 5:

THEOREM 2. $T(\alpha; n, N)$, *the number of iterations needed to get an expected number of* $n - \alpha$ *colored nodes, when starting with one colored node, is*

$$T(\alpha; n, N) \approx \frac{1 + N/n \ln \theta}{\log \theta} \log n.$$

Proof. First, it is easy to verify that $n = \theta^{\log n/\log \theta}$. Therefore $T(\alpha; n, N) = T(\alpha; \theta^{\log n/\log \theta}, N/n\theta^{\log n/\log \theta}) \approx T(\alpha; 1, N/n) + \log n/\log \theta\ (1 + N/n \ln \theta)$. Since $T(\alpha; 1, N/n) \approx 0$, the theorem follows. ■

COROLLARY 1. $n/N \approx 1$, $T(\alpha; n, N) \approx (1.693 + 1.414(1 - n/N))$ $\log n$.

COROLLARY 2. *For* $n \ll N$, $T(\alpha; n, N) \approx 2 \ln 2N/n \log n \approx 1.386N/n \log n$.

In the previous analysis we assumed that a node sends the color only if the color was available to that node at the beginning of the current unit of time. Starting with one colored node, let $\hat{T}(\alpha; n, N)$ be the number of steps needed to get an expected number of $n - \alpha$ colored nodes, when each node sends the color if it had it before sending the next message. Note that Lemma 3 is true for $\hat{T}(\alpha; n, N)$ too.

THEOREM 3. $\hat{T}(\alpha; n, N) \approx 2 \ln 2N/n \log n \approx 1.386N/n \log n$.

Proof. The analysis is similar to that of Lemma 4. Equation (3.1) becomes

$$\Delta x = N/nx(1 - x)\Delta t, \tag{3.3}$$

because the probability that a node sends a colored message is x rather than c. Integrating (3.3) from $t = 0$ to $t = j$, then substituting $c_0 = 1/n$, yields

$$\frac{c_j}{1 - c_j} = \frac{e^{nj/N}}{n - 1}.$$

Solving for j when $c_j = (n - \alpha)/n$ yields $e^{nj/N} = (n - 1)(n - \alpha)/\alpha$, or $j = \hat{T}(\alpha; n, N) \approx N/n(2 \ln n - \ln \alpha) = N/n(2 \ln 2 \log n - \ln \alpha)$, and the theorem follows. ■

REFERENCES

1. Leland, W. E., Finkel, R. A., Qiao, L., Solomon, M. H., and Uhr, L. High density graphs for processor interconnection. *Inform. Process. Lett.* **12,** 3 (June 1981), 117–120.
2. Leland, W. E., and Solomon, M. H. Dense trivalent graphs for processor interconnection. *IEEE Trans. Comput.* **C-31** (Mar. 1982), 219–222.
3. Preparata, F. P., and Vuillemin, J. The cube-connected cycles: A versatile network for parallel computation. *Comm. ACM* **24,** 5 (May 1981), 300–309.