Efficient All-to-All Broadcast Schemes in Distributed-memory Parallel Computers

Eunseuk Oh* Iyad A. Kanj†

Abstract

Distributed-memory parallel computers refers to parallel computers in which each processor has its own private memory. In such a system, processors communicate by message passing via interconnected network. One of important method to share distributed data is broadcasting. All-to-all broadcasting is the process that each processor broadcasts its distinct message to all other processors and receives different messages from all other processors. We investigate two all-to-all communication costs: time and message to measure the performance on the considered distributed-memory parallel model. We develop efficient communication scheme to reduce not only the communication times but also the minimal number of messages under the model with the half-duplex and full-duplex communication capability. The measurement is compared with the known bounds to verify the efficiency of the suggested scheme.

Key words all-to-all broadcast, distributed parallel computing system, inter-processor communication

1 Introduction

Due to the rapid progress in hardware technology, designing a distributed parallel computing system connecting autonomous microprocessors has become a feasible task. In such a system, high-performance microprocessors are communicated by message passing instead using a shared memory and a global clock. In general, the broadcasting messages between processors is the problem of dissemination of information across processors in the multiprocessor system. It is not difficult to see that broadcasting stands as a pivot for many applications on parallel computing systems. To list few of these applications that use broadcasting, we mention computing Fast Fourier Transformation (FFT), parallel matrix algorithms, parallel graph algorithms, and distributed algorithms.

For a communication scheme, its communication costs are characterized by the number of communication steps needed to complete the data transfer and the total number of messages incurred. A recent effort to investigate the performance based on these communication measure in distributed computing systems is shown in [4]. For a distributed-memory parallel computers of p processors with single-port communication capability, they extract minimum bounds $\Theta(\lceil logp \rceil)$ communication steps and $\Theta(np + p - 2^n)$ messages needed to complete an all-to-all broadcast, where $n = \lceil logp \rceil$. To apply their scheme, a model should be represented by an their addressing scheme on the balanced binary partitioning tree. In the same effort to investigate the performance based on time and message communication costs on distributed parallel computers, we consider the model which cannot be applied by an addressing scheme derived in [4]. The typical model constructed by their addressing scheme is hypercube model and its variations. It has been known that hypercube can not be embedded to star model [1, 2] with dilation 1. To investigate further communication costs in distributed parallel computing systems, we consider star model which has

*Corresponding author. Department of Computer Science, Texas A&M University, College Station, TX 77843-3112. E-mail: eunseuko@cs.tamu.edu.

†School of Computer Science, Telecommunications, and Information Systems, DePaul University, 243 S. Wabash Avenue, Chicago, IL 60604. E-mail: ikanj@cs.depaul.edu.

attracted considerable interest in the parallel processing research community due to its many desirable properties for building a large parallel computer systems. Since our graph model can not applied by their addressing scheme, we regard that their bounds is not tight bounds for communication costs to measure the efficiency of the suggested scheme.

Broadcasting schemes vary according to the communication capability on channel of link. Under the single-port communication capability, every processor can simultaneously send and receive at most one message in one communication step. Also, a channel or a link maybe full-duplex or half-duplex, where full-duplex meaning that the processor can send a message and receive another message along the same link simultaneously, and half-duplex meaning that a processor cannot send a message and receive another message along the same link simultaneously. Cost measurement of the suggested scheme is provided both under half-duplex and full-duplex communication capability. Followed by terminology in [4], our scheme is termed "NODUP" in the sense that there is no duplication of information on messages carried during communication process.

The paper is organized as follows. In Section 2 we introduce the graph model and assumptions made. In Section 3 we present our broadcasting scheme and analyze its running time. In Section 4 we present a comparative study between the suggested model and other models. The known bound for communication costs is also discussed. In Section 5, we conclude the paper and give some remarks.

2 Model and Assumptions

The graph model: The graph model considered is an undirected n-star graph(SG_n) consisting of $n!$ processors labeled with the $n!$ permutations on the symbols $\{1, 2, \ldots, n\}$. There is a communication link between two processors p_i and p_j in SG if and only if the permutation label for p_j can be obtained from the permutation label for p_i by exchanging the symbol in the first position in p_i with the symbol in some other position in p_i. If p_j is obtained from p_i by exchanging the first position in p_i with the kth position in p_i, then p_i and p_j are said to be connected along communication link k.

The interconnected pattern between processors on SG can be viewed recursively as follows. SG_1 is a trivial network with one processor. Suppose that SG_{n-1} has been defined inductively. Then SG_n is composed of n copies, SG^i_{n-1}, $i = 1 \ldots n$, where each SG^i_{n-1} is an isomorphic copy of SG_{n-1} with symbols $\{1, \ldots, n\} - i$, and with symbol i appearing as the nth symbol in each processor in SG^i_{n-1}.

The communication model supported in a parallel computers depends on the communication hardware and the memory bus bandwidth. Most of the commercial systems support the single-port model. In the single-port communication model, a processor can send a message on at most one port and receive a message on at most one port at a time. The sending and the receiving ports are not necessarily the same. The system model we consider is as follows.

Model M_s
(1) The system is completely connected with synchronous communication.
(2) A processor sends a message to a connected processor in one communication step.
(3) NODUP schemes: each message carries only new information in communication process.
(4) Single-port communication: duplex or half-duplex capability.

In the system model M_s, one message might contain many distinct informations originated by processors. Also, we assume that a message takes one communication step to be sent.

3 An all-to-all broadcast scheme

The all-to-all broadcasting is performed with p processors in SG_r, and then extends the broadcast to cover the whole processors, where $p = O(r!)$. To avoid sending a message twice to the same processors in the network, a private memory for each processor will be divided into two parts: the local storage, and the buffer. The local storage space will be used to store the messages that no longer need to be sent, whereas the buffer space will store the new messages that need to be sent to the other processors. Thus, the messages possessed by a processor at a certain time are the concatenation of the messages in its local storage and the messages in its buffer at that time.

Base of the scheme up to SG_4 under full-duplex model: The algorithm first calls itself recursively to perform broadcasting on each of SG_3^1, SG_3^2, SG_3^3, and SG_3^4. The base of the scheme is when the network is a SG_2, which is basically an exchange of message by the two processors. Then each processor in the SG_2 sends its message along communication link 3, and saves it in its local storage. Each processor sends its received message along communication link 2, and broadcasting in SG_3 is terminated. Each SG_3 in SG_4 has performed its all-to-all broadcast within itself. At this point every processor computes its message by concatenating the messages in its local storage with the messages in its local buffer and empties its local storage. Then, every processor sends its meta message along communication link 4, and saves its meta message in its local storage. Once a processor receives the meta message sent to it along communication link 4, it only needs to broadcast it across the copy of SG_3 which it belongs to. It starts that by sending it along communication link 3, and 2, meanwhile storing the messages that it receives in its local storage.

Base of the scheme up to SG_4 under half-duplex model: Every processor consisting of SG_3 send a message to a processor connected in clockwise order. After sending a message, each processor saves it in its local storage, and sends a message received from a left processor to a right processor in clockwise order. A message sent to a right processor is saved after sending it. After 5 communication steps of this phase, all processors in a SG_3 share same messages. At this point every processor computes its message by concatenating the messages in its local storage with the messages in its local buffer and empties its local storage. Processors in SG_3^i and SG_3^j send messages along communication link 4 to each direction at a time. After that, each processor only needs to broadcast a message across the copy of SG_3 which it belongs to. For each processor, it starts that by sending a message to its right processor, meanwhile storing the message received from its left processor in local buffer.

The scheme for SG_r, $r > 4$ under full-duplex or half-duplex model: Suppose inductively that an all-to-all broadcast has been completed within each SG_{r-1}^i, $i = 1 \ldots r$, and let us discuss how this can be extended easily to an all-to-all broadcast in SG_r. Since an all-to-all broadcast has been completed in each SG_{r-1}^i, each processor in S_{r-1}^i has received the messages of all other processors in SG_{r-1}^i, and hence all processors in SG_{r-1}^i share the same information. Denote the collection of all the messages in SG_{r-1}^i by the *meta* message Δ_{r-1}^i. Hence, every processor in SG_{r-1}^i has the meta message Δ_{r-1}^i stored in its buffer storage space. Let $\Delta_r = \bigcup_{i=1}^r \Delta_{r-1}^i$, then an all-to-all broadcasting in SC_r is achieved once every processor in SG_r possesses Δ_r in its storage space. The broadcasting is done as follows. In the first stage every processor p in SG_{r-1}^i, $i = 1 \ldots r$, broadcasts its meta message Δ_{r-1}^i along communication link r and saves it in its local storage. After this stage, each SG_{r-1}^i contains all messages in Δ_r among its processors. Thus, the only thing left to be done in the second stage is to propagate the information within each SG_{r-1}^i. This step needs to be done with some care so that to avoid sending a message more than once to the same processor. Once p receives the meta message Δ_{r-1}^j, $j \neq i$, along communication link r, it propagates Δ_{r-1}^j across SG_{r-1}^i by sending it along communication links $r-1, r-2, \ldots, 2$,

ATAB

if $r > 1$ **then**
1. each SG^i_{r-1} performs ATAB, recursively;
   /* At this point, all processors in SG^i_{r-1} have Δ^i_{r-1}, for $i = 1 \ldots r$, in their local buffer and have their local storage empty. */
2. each processor in SG^i_{r-1}, $i = 1 \ldots r$, sends Δ^i_{r-1} along communication link r and saves Δ^i_{r-1} in its local storage;
3. $d = r - 1$
4. once a processor in SG^i_{r-1} receives Δ^j_{r-1} from a processor in S^j_{r-1}, where $j \neq i$, it performs the following
   while $d \geq 2$ **do**
     if $d \leq 3$ and SG is half-duplex model **then**
       send Δ^j_{r-1} to the right processor;
       save the message received from the left processor in SG^i_{r-1} in the local storage;
     send Δ^j_{r-1} to the neighbor along communication link d in SG^i_{r-1};
     save the message received from the processor in SG^i_{r-1} in the local storage;
     $d = d - 1$;
5. every processor concatenates the messages in its buffer and local storage and empties its local storage;

Figure 1: The All-to-All-Broadcast scheme ATAB

respectively. It also stores at each time the message that it receives from a processor in SG^i_{r-1} in its local storage. For SG with half-duplex communication capability, it is done by simulating full-duplex by a delay factor of 1 on communication links up to link 4. After sending messages along communication link 4, it follows the ending phase in the base of scheme up to SG_4. An all-to-all broadcast scheme ATAB is shown in Figure 1.

We prove next the correctness of the algorithm and analyze its running time. Before we do that, we need the following key lemmas.

Lemma 3.1 *When ATAB on input $r > 0$ terminates, each processor in SG_r possesses the meta message Δ_r.*

Assume inductively that when ATAB on input $r-1$ terminates, each processor in SG_{r-1} contains the meta message Δ_{r-1}. When ATAB is called on input r, ATAB calls itself recursively on each SG^i_{r-1}, $i = 1 \ldots r$. Since each SG^i_{r-1} is a copy of SG_{r-1}, by the inductive hypothesis, when each of these recursive calls terminates, each processor in SG^i_{r-1}, $i = 1 \ldots r$, possesses the meta message Δ^i_{r-1}. An illustration of step 2 is shown in Figure 2. From the recursive definition of SG_r given in Section 2, each SG^i_{r-1} is linked to all SG^j_{r-1}, $j \neq i$ by exactly $(r-2)!$ links along communication link r. It follows that after the execution of step 3 of ATAB, exactly $(r-2)!$ processors in SG^i_{r-1} possess each meta message Δ^j_{r-1}, $j = 1 \ldots r$, $j \neq i$. Also, each processor in SG^i_{r-1} has Δ^i_{r-1} stored in its local storage space. It follows that each SG^i_{r-1} contains copies of all the meta messages. Let p_1 be a processor in S^i_{r-1}, and let Δ^j_{r-1}, $j \neq i$, be a meta message. From the above discussion, we know that there are exactly $(r-2)!$ processors in SG^i_{r-1} that possess the meta message Δ^j_{r-1}. If p_1 is one of these processors then we are done. Suppose this is not the case. Now from step 4 in ATAB, we know that each of these processors will send Δ^j_{r-1} along communication links $r-1, \ldots, 2$, respectively. We first claim that no processor p_1 in SG^i_{r-1} is the neighbor of two distinct processors p_2 and p_3 that possess the meta message Δ^j_{r-1} at the beginning of step 4. Suppose, to get a contradiction, that this were not the case. Let $p_1 = \sigma_1 \ldots \sigma_r$, and suppose that p_1 is the neighbor of p_2 and p_3 along communication links x and $y \in \{1, \ldots, r-1\}$, respectively. Notice first that $x \neq y$ since a processor is connected to exactly one processor along any

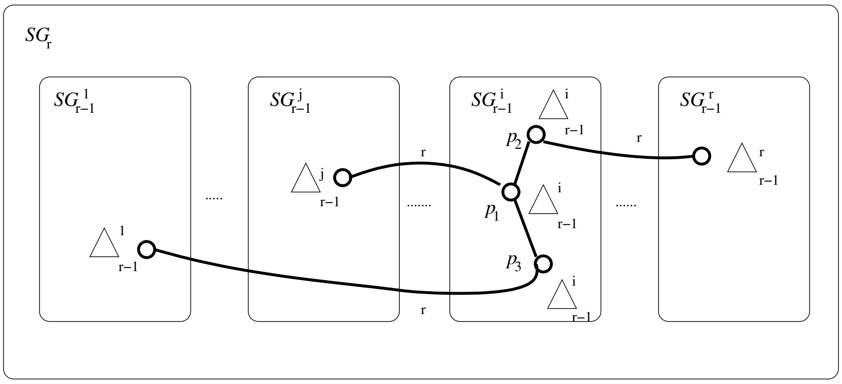

Figure 2: All-to-all broadcast scheme in SG_r(step 2)

given communication link. Since p_1 is connected to p_2 along communication link x, $p_2 = \sigma_x, \ldots, \sigma_1, \ldots \sigma_r$. Similarly $p_3 = \sigma_y, \ldots, \sigma_1, \ldots \sigma_r$. Since both p_2 and p_3 have the meta message Δ^j_{r-1} with $j \neq i$, both p_2 and p_3 are connected to SG^j_{r-1} along communication link r. Now all processors in SG^j_{r-1} have the same rth symbol. Since p_2 and p_3 are the neighbors of two processors in SG^j_{r-1} along communication link r, it follows that both p_2 and p_3 have the same first symbol and $\sigma_x = \sigma_y$, a contradiction, since σ_x and σ_y are two symbols in the representation of processor p_1, and hence must be distinct.

It follows from the above claim that the neighbors of the processors possessing the meta message Δ^j_{r-1} at the beginning of step 4 of ATAB are distinct. Now each processor that possesses Δ^j_{r-1} broadcasts this meta message to exactly $r-2$ neighbors along communication links $r-1, \ldots, 2$. Since all these neighbors are distinct, the number of processors in SG^i_{r-1} that receive the meta message Δ^j_{r-1} from a processor in SG^i_{r-1} possessing this meta message at the beginning of step 4 is $(r-2)(r-2)!$. Thus, the total number of processors in SG^i_{r-1} that possess meta message Δ^j_{r-1} at the end of ATAB is $(r-2)! + (r-2)(r-2)! = (r-1)!$. It follows that all processors in SG^i_{r-1} possess meta message Δ^j_{r-1} and in particular p_1. Since p_1 and j were arbitrarily chosen, every processor in SG^i_{r-1} possesses every Δ^j_{r-1} at the end of ATAB, and hence possesses $\bigcup_{j=1}^{r} \Delta^j_{r-1} = \Delta_r$.

Theorem 3.2 *ATAB performs all-to-all broadcasting in SG of $n!$-processor systems with a single-port, full-duplex communication capability in time $\frac{n(n-1)}{2}$ and by incurring $\Theta(n^2 n!)$ messages.*

When ATAB is called on SG_r, it terminates after exactly $\frac{r(r-1}{2}$ communication steps. Let T_r be the number of communication steps on SG_r performed by ATAB, and M_r be the number of messages incurred on SG_r by ATAB. Each S^i_{r-1} performs an all-to-all broadcast within itself and then sends a single message along communication link r, and then along communication links $r-1, \ldots, 2$. Thus, the number of communication steps performed by each SG^i_{r-1} is $r-1$, and. the number of messages incurred by each SG^i_{r-1} is $r! + r(r-1)!(r-2)$. Since all the SG^i_{r-1}'s do this in parallel, the number of communication steps for SG_r is the same as the number of communication steps performed by each SG^i_{r-1}. The number of communication steps performed when ATAB terminates on SG^i_{r-1} is T_{r-1}. Thus, the total number of communication steps performed by each S^i_{r-1}, and hence, by the whole network is given by the recurrence $T_r = T_{r-1} + r - 1$. This gives $T_r = \frac{r(r-1)}{2}$. Similarly, The number of messages incurred when ATAB terminates on SG^i_{r-1} is M_{r-1}. Thus, the total number of messages incurred by each SG^i_{r-1}, and hence, by the whole network is given by the recurrence $M_r = rM_{r-1} + r! + r!(r-2)$, $M_3 = 18$. This gives $M_r = \Theta(n^2 n!)$.

Table 1: Topological properties of models of inter-connecting processors

Network Model	Size	Degree	Diameter	Links
SG_n	$n!$	$n-1$	$\lceil \frac{3(n-1}{2} \rceil$	$\frac{(n-1)n!}{2}$
$Cube_n$	2^n	n	n	$n2^{n-1}$
$MCT_r(2^h-1)$	$(2^h-1)^r$	Min r Max $3r$	$2r(h-1)$	$r(2^h-1)^(r-1)$
D_n	2^n	2 or 4	n	2^{n+1}
$HS_{n,m}$	$n!m!$	n+m-2	$\lceil \frac{3(n-1)}{2} \rceil + \lceil \frac{3(m-1)}{2} \rceil$	$\frac{(n+m+2)n!m!}{2}$

Under the half-duplex communication capability of SG_r, the total number of communication steps is given by the recurrence $T_r = T_{n-1} + 2(n-3) + 2$, $T_3 = 5$, and the total number of messages incurred by the whole network is $M_r = rM_{r-1} + 2(r! + r!(r-2))$, $M_3 = 30$.

Theorem 3.3 *ATAB performs all-to-all broadcasting in SG of $n!$-processor systems with a single-port, half-duplex communication capability in time $n(n-3)+5$ and by incurring $\Theta(n^2 n!)$ messages.*

4 Related works

In this section we present a comparative study between SG and other inter-connected models. Compared models are hypercube $Cube_n$, MCT(mesh connected trees [5]), the deBruijn network D_n [9], hyperstar $HS_{n,m}$ (Cartesian product network of hypercube and star). Basic parameters such as size, degree, diameter are shown in Table 1 and comparison of communication steps and messages required for model whose size is similar to SG is shown in Table 2 based on communication costs shown in [4]. Under the half-duplex communication capability, we can consider that ATAB time complexity will be increased by factor 2.

Chen, Yu and Wu [4] developed an algorithm that for a distributed parallel computers of p processors with the single-port capability, it completes an all-to-all broadcast in time $\Theta(\lceil logp \rceil)$ by incurring $\Theta(np + p - 2^n)$ messages, where $n = \lceil logp \rceil$. It has been known that the star model cannot be embedded to hypercube model with dilation 1. Thus, their addressing scheme based on hypercube model cannot be directly applied to SG. We regard that their bounds are not tight lower bounds of communication time and the number of messages. Comparison are made with other models whose size is similar to SG. Under the NODUP scheme by Chen, Yu and Wu, the lower bound for communication time of SG_n is $\lceil logn! \rceil = \Theta(nlogn)$, and the suggested scheme takes $O(n^2)$. The number of messages incurred by the lower bound of SG_n is $nlogn! + n! + 2^{logn!} = O(n! + 2^{nlogn})$, and the suggested scheme incurs $\Theta(n^2 n!)$ messages.

5 Conclusions and Remarks

An efficient all-to-all NODUP broadcast scheme is developed to investigate communication performance of distributed-memory parallel computers. The existing NODUP all-to-all scheme for distributed-memory parallel computers is designed for only interconnection models which can be represented by addressing scheme developed in [4]. Thus, other systems which is not variations of hypercube model would lead to very different solutions. The emphasis was on how to complete the all-to-all broadcast in the system that cannot be directly applied by the existing NODUP scheme, trying to reduce not only the communication

Table 2: Comparison of communication costs for parallel computer models of similar sizes(full-duplex)

Model	Size	Degree	Diameter	ATAB time complexity	ATAB message complexity
SG_5	120	4	6	7	832
$Cube_7$	128	7	7	7	896
$MCT_4(3)$	81	12	16	7	520
D_7	128	4	7	7	896
$HS_{4,3}$	144	5	7	8	1040
SG_6	720	5	7	10	6896
$Cube_{10}$	1024	10	10	10	10240
$MCT_6(3)$	729	18	12	10	6995
D_{10}	1024	4	10	10	10240
$HS_{5,3}$	720	6	9	10	6896

times but also the minimal number of messages. Similar to other solutions closely depends on the parallel computer models assumed, the suggested schemes derived in this paper are results from the SG model. Under a NODUP scheme for a system of p processors, the minimum number of message transmissions needed to complete an all-to-all broadcast is $p(p-1)$. The suggested scheme reaches that message transmission bound $n!(n!-1)$ during the entire scheme in SG, which means no duplication of information in any messages incurred during the entire scheme. The suggested scheme for SG model completes an all-to-all broadcast in communication time $\Theta(n^2)$ by incurring $\Theta(n^2 n!)$ messages. It can be justified that the best known results [3, 8] for one-to-all broadcast take $\sum_{i=2}^{n} \lceil logi \rceil = O(nlogn)$ communication time on SG with the single-port capability. Thus, all-to-all broadcast would take at least $O(nlogn)$ with possibility that messages can be delayed on communication links. As an example, if a constant factor is 3, our scheme is faster as long as n reached the value of approximately 30. Further, Tseng and Sheu [10] suggested that $\max\{\lceil logn! \rceil, \lfloor \frac{3(n-1)}{2} \rfloor\}$ communication time is an lower bound to perform an all-to-all broadcast on SG model. The bound $\lfloor \frac{3(n-1)}{2} \rfloor$ is not tight either. Even though a message can be sent along the spanning tree with length of $\lfloor \frac{3(n-1)}{2} \rfloor$, it would have more time because the communication link has the single-port capability. Under the full-duplex capability, if a message is sent on communication links by delay factor of $n-1$, our scheme is faster by a factor of 3. Under the half-duplex capability, if a message is sent on communication links by delay factor of $2(n-1)$, our scheme is still faster by a factor of 3. Finally, the assumption that a message take one communication step to be sent can be justified by the technique of virtual cut-through for communication [7], which can be incorporated into multicomputer systems such as iPSC/2 [6].

References

[1] S. B. Akers, D. Harel, and B. Krishnamurthy, The star graph: an attractive alternative to the n-cube, *Proc. Int'l. Conf. of Parallel Processing*, (1987), pp. 393-400.

[2] S. B. Akers and B. Krishnamurthy, The fault tolerance of star graphs, *Proc. 2nd Int'l. Conf. on Supercomputing*, (1987), pp. 270-276.

[3] S. G. Akl, K. Qiu, and I. Stojmenovic, Fundamental algorithms for the star and pancake interconnection networks with applications to computational geometry, *Networks*, (1993), vol.23, no. 4, pp. 215-225.

[4] M. -S. Chen, P. S. Yu, and K. -L. Wu, Optimal NODUP all-to-all broadcast schemes in distributed computing systems, *IEEE Trans. Parallel and Distributed Systems*, (1994), pp. 1275-1284.

[5] K. Efe and A. Fernandez, Computational properties of mesh connected trees: Versatile architecture for parallel computation, *Int'l Conference on Parallel Processing*, (1994), pp. 72-76.

[6] Intel Corp. iPSC/2 User's Guide, Intel Corp., Mar, 1988.

[7] P. Kermani and L. Kleinrock, Virtual cut-through: A new computer communication switching technique, *Comput. Netw.* (1979), vol.3, pp. 267-286.

[8] J. -P. Sheu, C. -T. Wu, and T.-S. Chen, An optimal broadcasting algorithm without message redundancy in star graphs, *IEEE Trans. Parallel and Distributed Systems* (1995), vol.6, no. 6, pp. 653-658.

[9] M. Samatham and D. K. Pradhan, The De Bruijn multiprocessor networks: A versatile parallel processing and sorting network for VLSI, *IEEE Trans. Computers*, (1989), vol. 38, no. 4, pp. 567-581.

[10] Y. -C. Tseng and J. -P. Sheu, Toward optimal broadcast in a star graph using multiple spanning trees, *IEEE Trans. Computers* (1997), vol. 46, pp. 593-599.