Name: A. Shiva Surya Saran
Roll No.: BE-B 20

High Performance Computing

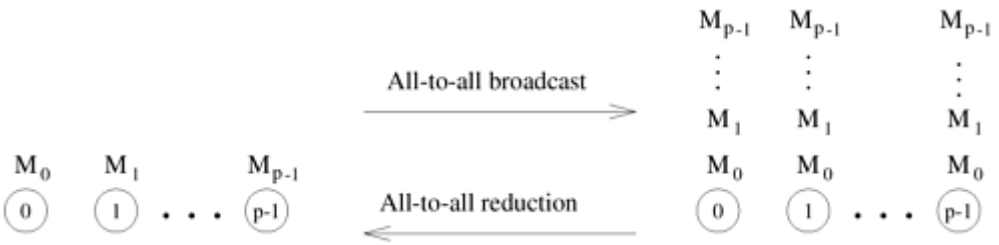
Assignment 3

Unit 3: Basic Communication Model

1. Explain the All-to-All Broadcast and Reduction algorithm.

All-to-all broadcast is a generalization of one-to-all broadcast in which all p nodes simultaneously initiate a broadcast. A process sends the same m-word message to every other process, but different processes may broadcast different messages.

All-to-all broadcast is used in matrix operations, including matrix multiplication and matrix-vector multiplication. The dual of all-to-all broadcast is all-to-all reduction, in which every node is the destination of an all-to-one reduction. Below figure illustrates all-to-all broadcast and all-to-all reduction.



All-to-all broadcast and all-to-all reduction.

One way to perform an all-to-all broadcast is to perform p one-to-all broadcasts, one starting at each node. If performed naively, on some architectures this approach may take up to p times as long as a one-to-all broadcast. It is possible to use the communication links in the interconnection network more efficiently by performing all p one-to-all broadcasts simultaneously so that all messages traversing the same path at the same time are concatenated into a single message whose size is the sum of the sizes of individual messages.

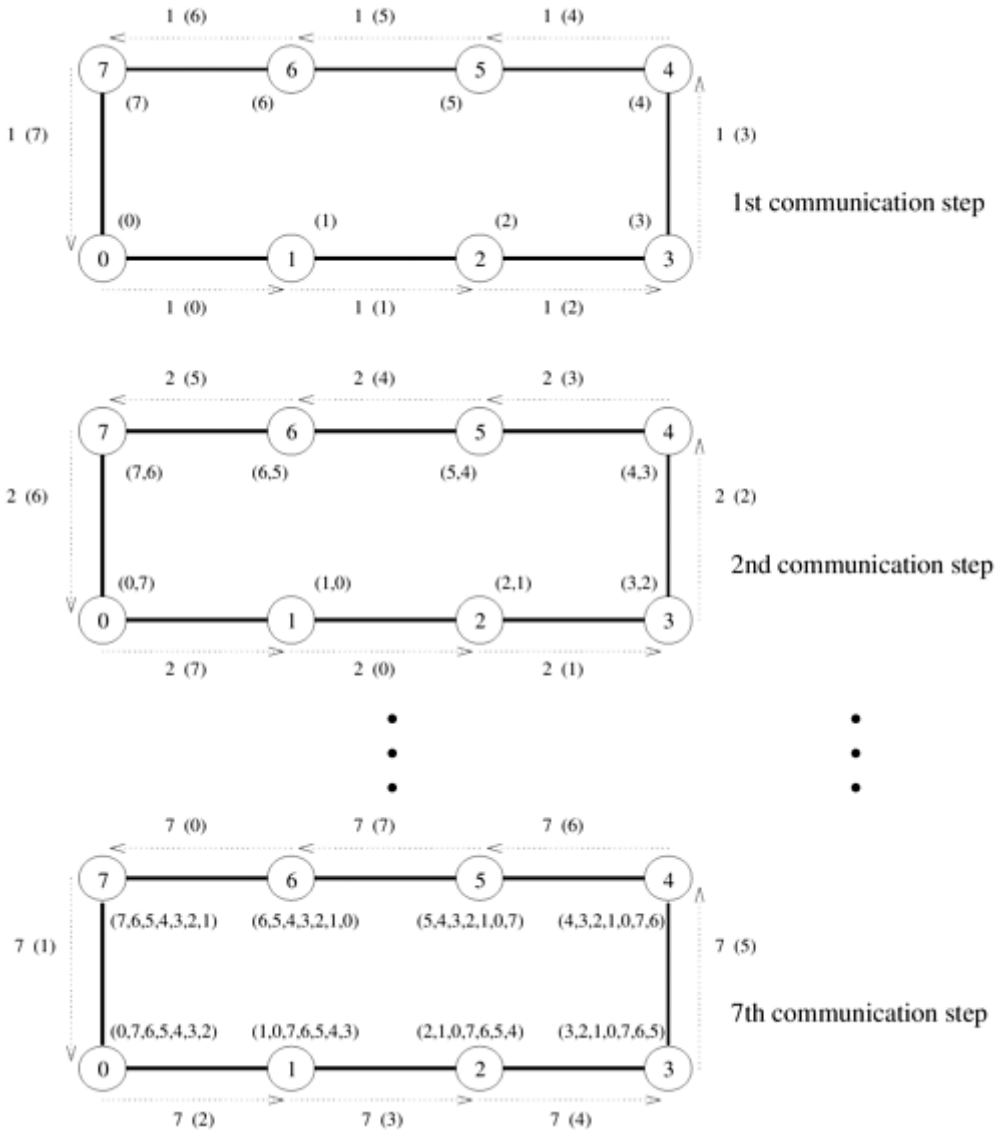
The following describe all-to-all broadcast on linear array, mesh, and hypercube topologies.

Linear Array and Ring

While performing all-to-all broadcast on a linear array or a ring, all communication links can be kept busy simultaneously until the operation is complete because each node always has some information that it can pass along to its neighbour.

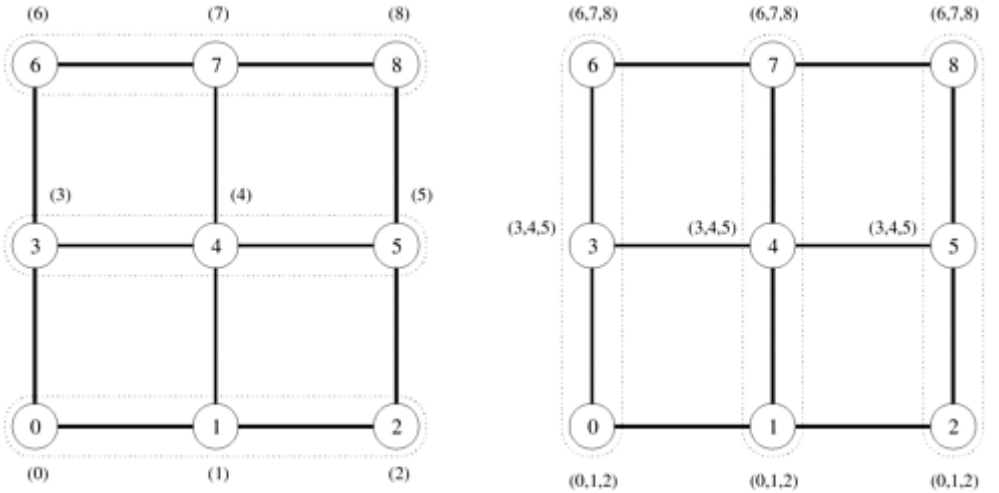
Each node first sends to one of its neighbours the data it needs to broadcast. In subsequent steps, it forwards the data received from one of its neighbours to its other neighbour.

Below figure illustrates all-to-all broadcast for an eight-node ring. The same procedure would also work on a linear array with bidirectional links. As with the previous figures, the integer label of an arrow indicates the time step during which the message is sent. In all-to-all broadcast, p different messages circulate in the p-node ensemble. In Figure, each message is identified by its initial source, whose label appears in parentheses along with the time step. For instance, the arc abelled 2 (7) between nodes 0 and 1 represents the data communicated in time step 2 that node 0 received from node 7 in the preceding step. As Figure shows, if communication is performed circularly in a single direction, then each node receives all (p – 1) pieces of information from all other nodes in (p – 1) steps.



Mesh

Just like one-to-all broadcast, the all-to-all broadcast algorithm for the 2-D mesh is based on the linear array algorithm, treating rows and columns of the mesh as linear arrays. Once again, communication takes place in two phases. In the first phase, each row of the mesh performs an all-to-all broadcast using the procedure for the linear array.

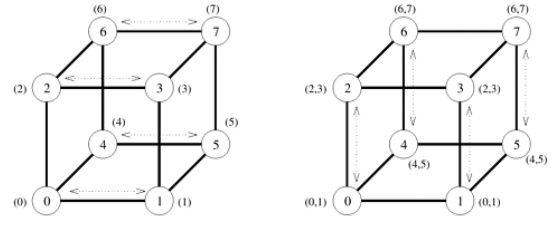


(a) Initial data distribution (b) Data distribution after rowwise broadcast

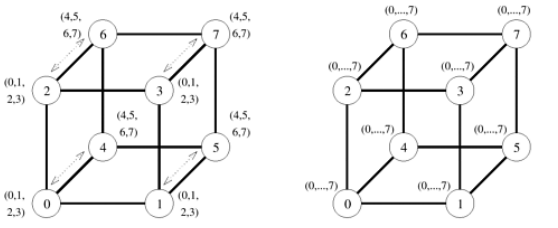
Hypercube

The hypercube algorithm for all-to-all broadcast is an extension of the mesh algorithm to log p dimensions. The procedure requires log p steps. Communication takes place along a different dimension of the p-node hypercube in each step.

In every step, pairs of nodes exchange their data and double the size of the message to be transmitted in the next step by concatenating the received message with their current data. Figure shows these steps for an eight-node hypercube with bidirectional communication channels.



(a) Initial distribution of messages (b) Distribution before the second step



(c) Distribution before the third step (d) Final distribution of messages

2. Describe Cost Analysis of Broadcast and Reduction algorithm.

The broadcast or reduction procedure involves log p point-to-point simple message transfers, each at a time cost of ts + twm.

The total time is therefore given by:

$$T = (t_s + t_w m) \log p.$$

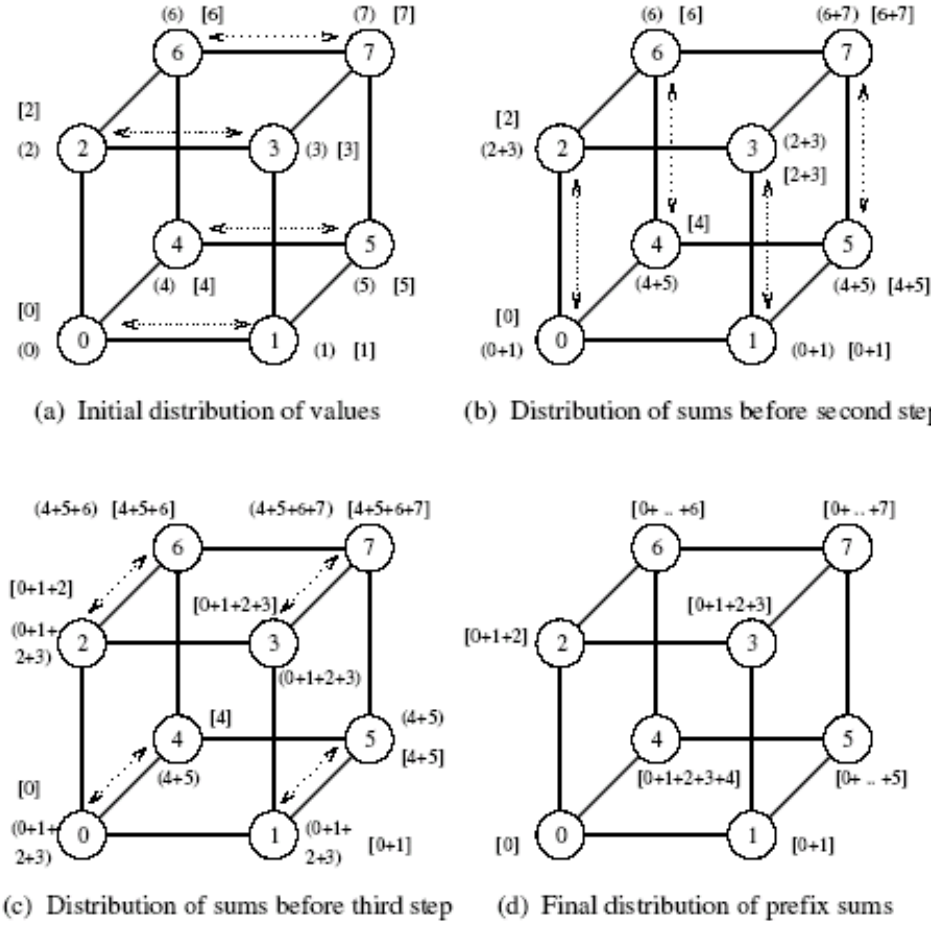
3. Explain All-Reduce and Prefix-Sum Operations.

All-reduce

- Here each node starts with a buffer of size m and the final results of the operation are identical buffers of size m on each node that are formed by combining the original p buffers using an associative operator.
- Identical to all-to-one reduction followed by a one-to-all broadcast. This formulation is not the most efficient. Uses the pattern of all-to-all broadcast, instead. The only difference is that message size does not increase here.
- Time for this operation is on a hypercube (ts + twm) log p.
- Different from all-to-all reduction, in which p simultaneous all-to-one reductions take place, each with a different destination for the result.

Prefix – Sum Operation

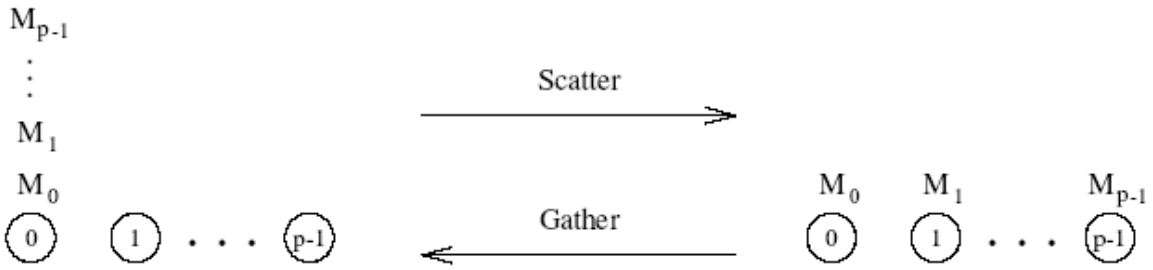
- Given p numbers n0,n1,…,np-1 (one on each node), the problem is to compute the sums sk = ∑ik= 0 ni for all k between 0 and p-1 .
- Initially, nk resides on the node labeled k, and at the end of the procedure, the same node holds Sk.
- The operation can be implemented using the all-to-all broadcast kernel.
- We must account for the fact that in prefix sums the node with label k uses information from only the k-node subset whose labels are less than or equal to k.
- This is implemented using an additional result buffer. The content of an incoming message is added to the result buffer only if the message comes from a node with a smaller label than the recipient node.
- The contents of the outgoing message (denoted by parentheses in the figure) are updated with every incoming message.



Computing prefix sums on an eight-node hypercube. At each node, square brackets show the local prefix sum accumulated in the result buffer and parentheses enclose the contents of the outgoing message buffer for the next step.

4. Describe Scatter and Gather operation in communication.

In the scatter operation, a single node sends a unique message of size m to every other node (also called a one-to-all personalized communication).
In the gather operation, a single node collects a unique message from each node.
While the scatter operation is fundamentally different from broadcast, the algorithmic structure is similar, except for differences in message sizes (messages get smaller in scatter and stay constant in broadcast). The gather operation is exactly the inverse of the scatter operation and can be executed as such.



Scatter and gather operations

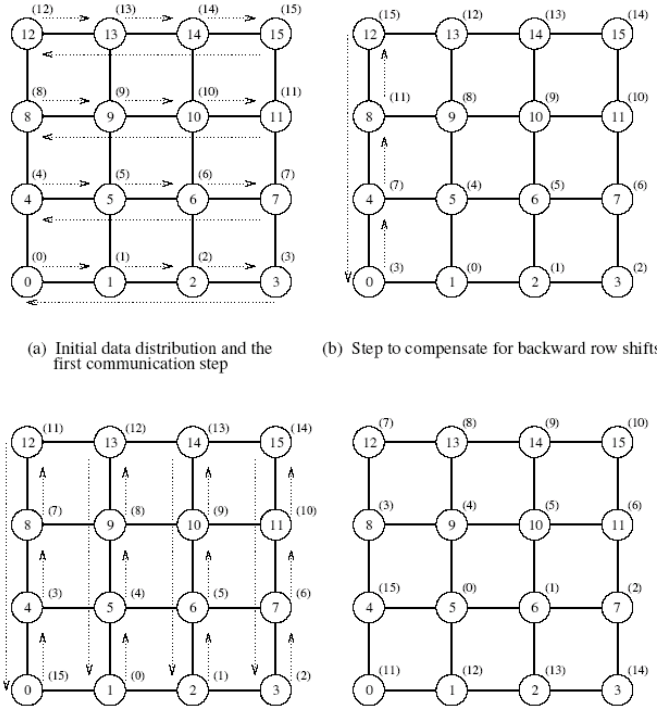
5. Explain Circular shift on a Mesh and on Hypercube.

A special permutation in which node i sends a data packet to node (i + q) mod p in a p-node ensemble (0 ≤ q ≤ p).

Circular Shift in Mesh

The implementation on a ring is rather intuitive. It can be performed in min{q,p – q} neighbor communications. Mesh algorithms follow from this as well. We shift in one direction (all processors) followed by the next direction. The associated time has an upper bound of:

$$T = (t_s + t_w m)(\sqrt{p} + 1).$$



(a) Initial data distribution and the first communication step

(b) Step to compensate for backward row shifts

(c) Column shifts in the third communication step

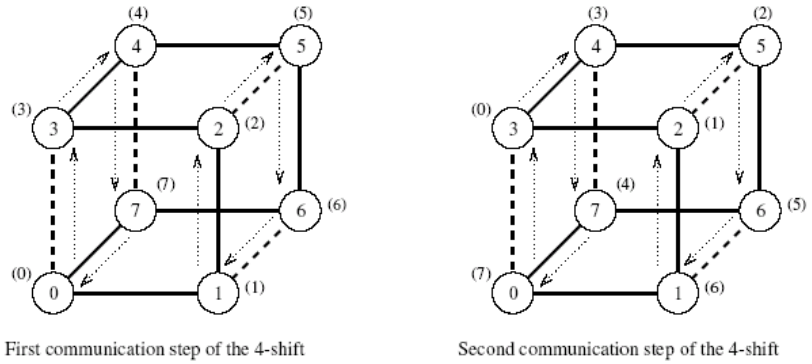
(d) Final distribution of the data

Circular Shift on a Hypercube

Map a linear array with 2d nodes onto a d-dimensional hypercube.
To perform a q-shift, we expand q as a sum of distinct powers of 2.
If q is the sum of s distinct powers of 2, then the circular q-shift on a hypercube is performed in s phases.
The time for this is upper bounded by:

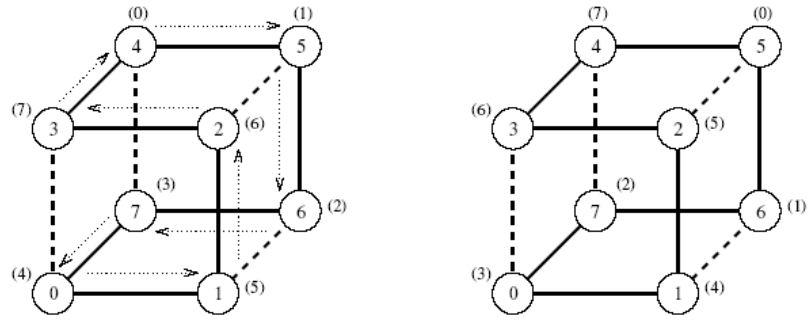
$$T = (t_s + t_w m)(2 \log p - 1).$$

If E-cube routing is used, this time can be reduced to $T = t_s + t_w m.$



First communication step of the 4-shift — Second communication step of the 4-shift

(a) The first phase (a 4-shift)



(b) The second phase (a 1-shift) — (c) Final data distribution after the 5-shift

The mapping of an eight-node linear array onto a three-dimensional hypercube to perform a circular 5-shift as a combination of a 4-shift and a 1-shift.

6. *Explain the strategies to improve the speed of communication operations.*

Splitting and Routing Messages in Parts

- ***One-to-All Broadcast***
  Consider broadcasting a single message M of size m from one source node to all the nodes in a p-node ensemble. If m is large enough so that M can be split into p parts M0, M1, ..., M p-1 of size m/p each, then a scatter operation can place Mi on node i in time ts log p + tw(m/p)(p - 1). This all-to-all broadcast can be completed in time ts log p + tw(m/p)(p - 1) on a hypercube. Thus, on a hypercube, one-to-all broadcast can be performed in time

- ***All-to-One Reduction***
  All-to-one reduction is a dual of one-to-all broadcast. Therefore, an algorithm for all-to-one reduction can be obtained by reversing the direction and the sequence of communication in one-to-all broadcast. Therefore, using the notion of duality, we should be able to perform an all-to-one reduction by performing all-to-all reduction (dual of all-to-all broadcast) followed by a gather operation (dual of scatter).

- **All-Reduce**
  Since an all-reduce operation is semantically equivalent to an all-to-one reduction followed by a one-to-all broadcast, the asymptotically optimal algorithms for these two operations presented above can be used to construct a similar algorithm for the all-reduce operation. Breaking all-to-one reduction and one-to-all broadcast into their component operations, it can be shown that an all-reduce operation can be accomplished by an all-to-all reduction followed by a gather followed by a scatter followed by an all-to-all broadcast.

All-Port Communication

In a parallel architecture, a single node may have multiple communication ports with links to other nodes in the ensemble. For example, each node in a two-dimensional wraparound mesh has four ports, and each node in a d-dimensional hypercube has d ports. In single-port communication, a node can send data on only one of its ports at a time. Similarly, a node can receive data on only one port at a time. However, a node can send and receive data simultaneously, either on the same port or on separate ports. In contrast to the single-port model, an all-port communication model permits simultaneous communication on all the channels connected to a node.