Parallel and Distributed Computing
Chapter 5: Basic Communications Operations

Jun Zhang

Laboratory for High Performance Computing & Computer Simulation
Department of Computer Science
University of Kentucky
Lexington, KY 40506

5.1a: Communication in Parallel System

- **Nearest neighbor communication:** Communication between two directly link nodes
- **Remote node communication**: With more than one links between the communicating nodes

  1.) Store-and-forward routing

  2.) Cut-through routing

5.1b: Basic Communication Operations

- One to one communication
- One to all communication (broadcasting)
- All to all communication
- All to one communication (reduction)

- These basic communication operations are commonly used on various parallel architectures
- It is crucial that they be implemented efficiently on a particular parallel system

5.1c: Commonly Used Interconnections

- Linear array
- Two-dimensional mesh
- Hypercube

5.1d: Mesh Topology

- A large number of processors can be connected relatively inexpensively with mesh topology
- Many applications map naturally onto a mesh network
- The disadvantage of high diameter of mesh topology can be diminished for networks with cut-through routing
- Several commercially available parallel computers are based on mesh network
- T3D, SGI, IBM Blue Gene

5.1e: Hypercube Topology

- The fastest hypercube algorithms are asymptotically as fast as the fastest PRAM algorithms
- Hypercubes tap maximum concurrency and impose data locality
- The best hypercube algorithm is also the best for other networks such as fat trees, meshes, and multistage networks
- Hypercube has an elegant recursive structure that makes it attractive for developing a wide class of algorithms

5.2a: Basic Assumptions

- Network supports store-and-forward routing and cut-through routing
- The communication links are bidirectional
- Single-port communication model

  One node can only send one message at a time

  It can only receive one message at a time

  Send and receive can be done simultaneously

5.2b: Dual Operations

- A **dual** of a communication is the opposite of the original operation
- It can be performed by reversing the direction and sequence of messages in the original operation

E.g., All-to-one communication (reduction) is the dual of one-to-all broadcast.

5.3a: One-to-All Broadcast and All-to-one Reduction (Single Node Accumulation)

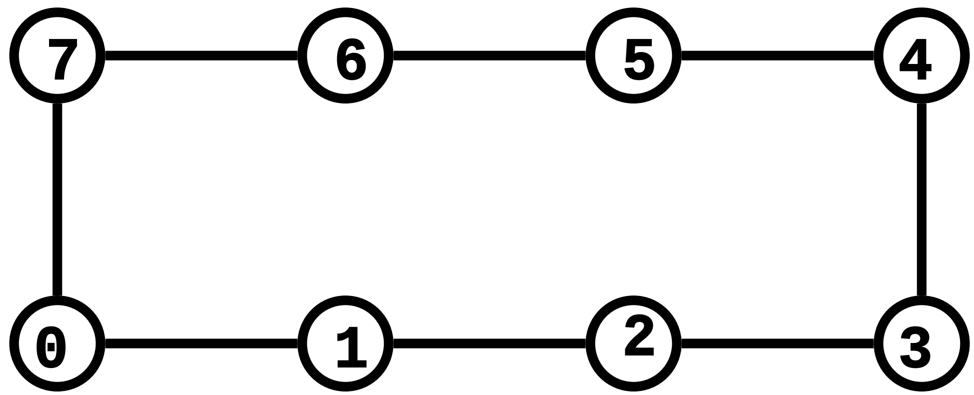
- A single process sends identical data to all other processes or to a subset of them

  e.g., distributing common parameters
- All-to-one reduction: Each of the *p* participating processes sends a data of size *m* to be accumulated at a single destination process into one *m* word data

  e.g., sum, maximum, inner product, etc.

5.3b: Ring Network

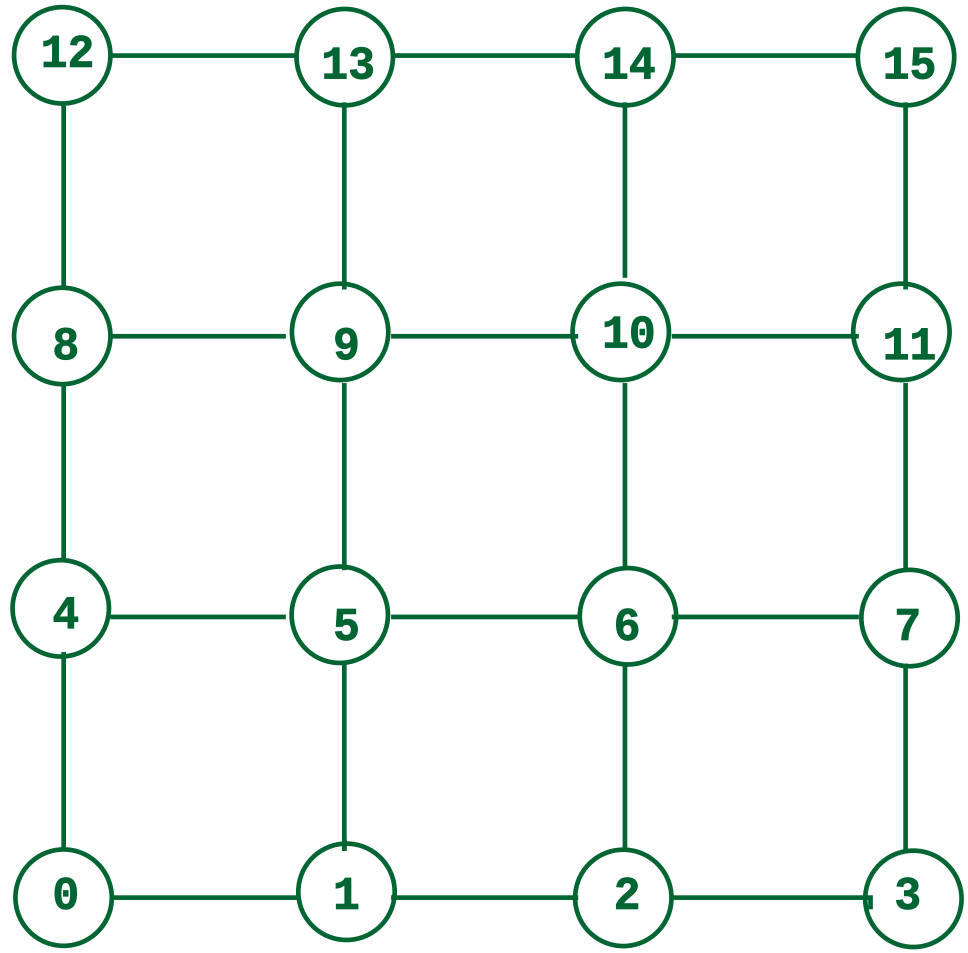
Message of size *m* at node 0, to be sent to all other nodes in the network



Naïve algorithm
Better algorithm
Fast algorithm
Recursive doubling

5.3c: Mesh Network

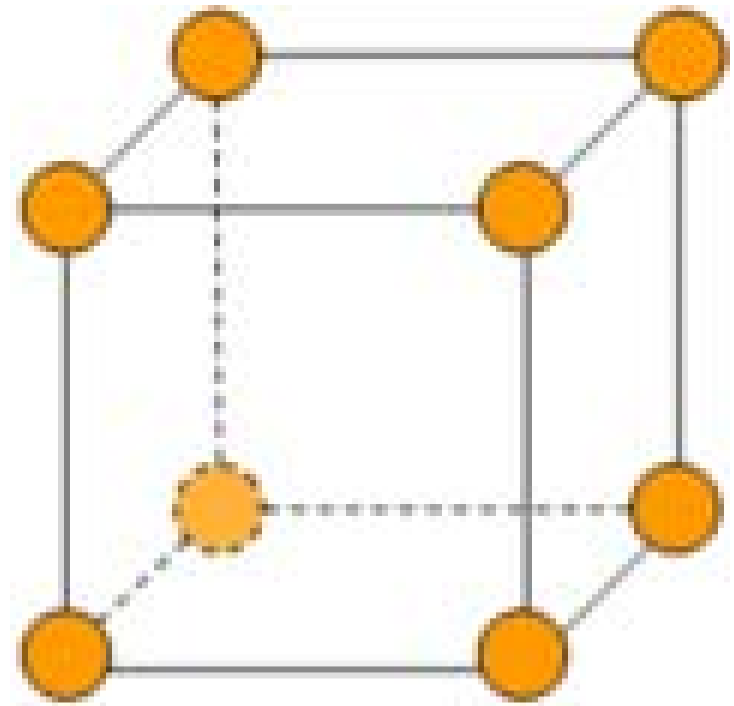
One to all broadcast in a 2D mesh can be performed in two steps, each step is a one to-all broadcast using the ring algorithm



5.3d: Hypercube

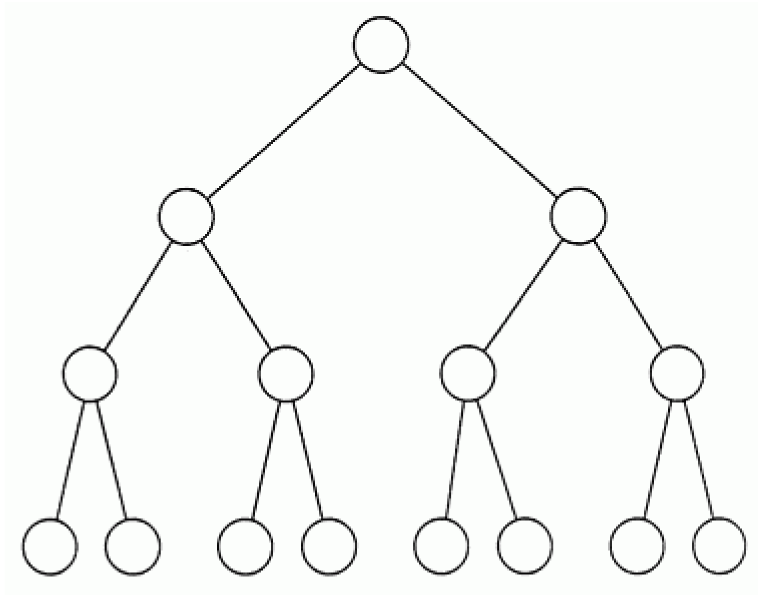
Use recursive doubling algorithm

No difference with different routing algorithm



5.3e: Balanced Binary Tree

Only the root nodes are processing nodes, map the hypercube algorithm directly



5.3f: Communication Cost

- If we assume cut-through routing and ignore the per hop time, all one-to-all broadcast communications can be viewed as log *p* steps of point-to-point communications.
- The communication cost for all networks is the same:

$$T_{comm} = (t_s + t_w m) \log p$$

5.4a: All-to-All Broadcast and Reduction

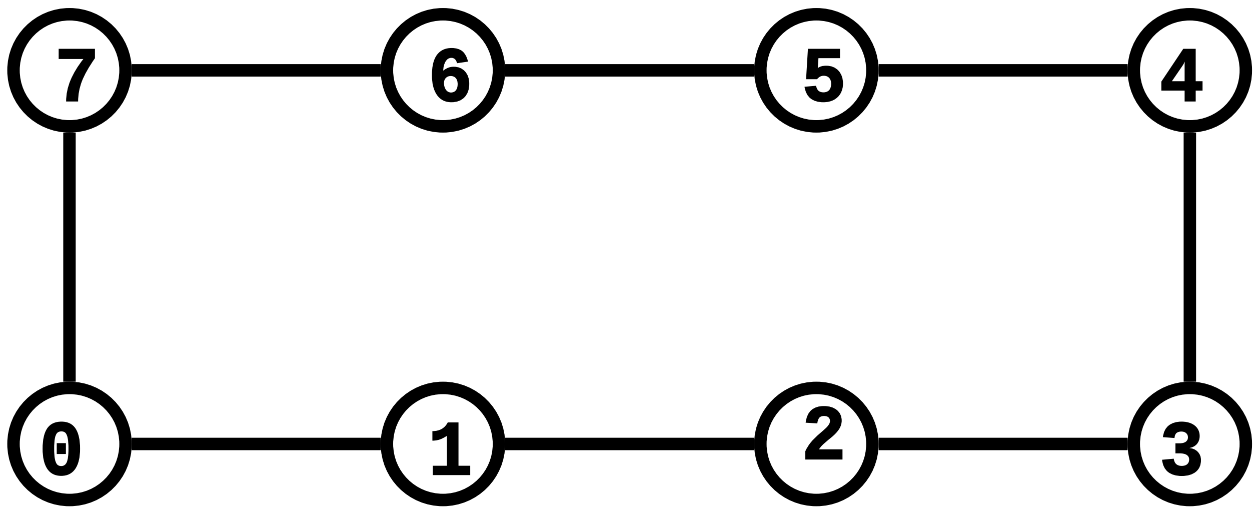
- **All-to-all broadcast** can be viewed as a generalization of one-to-all broadcast
- All *p* nodes simultaneously t_w initiate a broadcast
- Each node sends the same *m*-word message to every other nodes
- Different node may broadcast different messages
- **Applications** include matrix-multiplication and matrix-vector multiplication
- The dual of all-to-all broadcast is **all-to-all reduction**
- Every node is the destination of an all-to-one reduction

5.4b: Ring and Linear Array

- All-to-all broadcast is achieved by a pipelined point-to-point nearest neighbor communication
- For linear array, bi-directional link is necessary
- For all-to-all reduction, the procedure is reversed, each node needs to perform the operation at each step
- The total communication cost is:

$$T_{ring} = (t_s + t_w m)(p - 1)$$

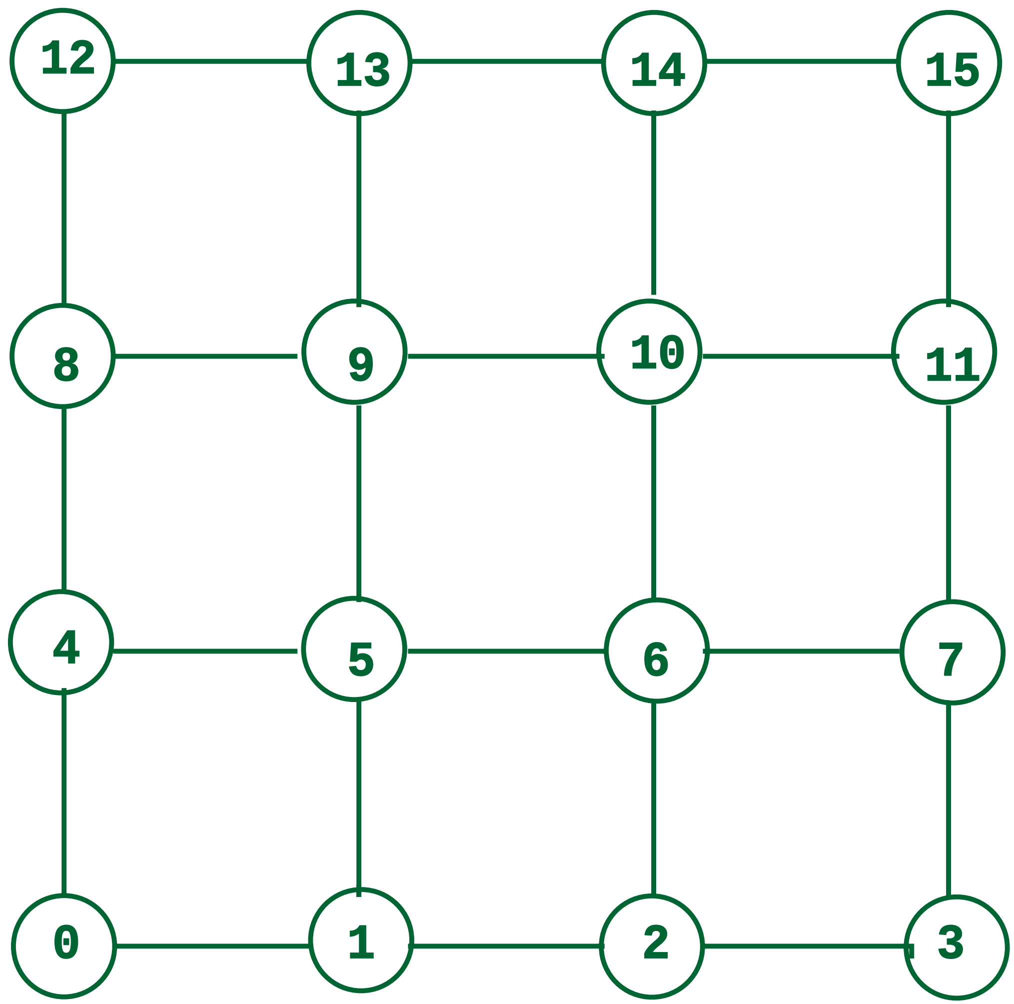
5.4c: All-to-All on a Ring Network



5.4d: 2D Mesh Network

- All-to-all broadcast algorithm for the 2D mesh is based on the ring algorithm
- The rows and columns of the mesh are treated as rings in two steps
- First, each row of the mesh performs an all-to-all broadcast using the ring algorithm, collecting $\sqrt{p}$ messages corresponding to the $\sqrt{p}$ nodes of their respective rows
- Second, each column performs an all-to-all broadcast, with a single message of size $m\sqrt{p}$

5.4e: Illustration of 2D Mesh All-to-All



5.4f: Cost of 2D Mesh All-to-All

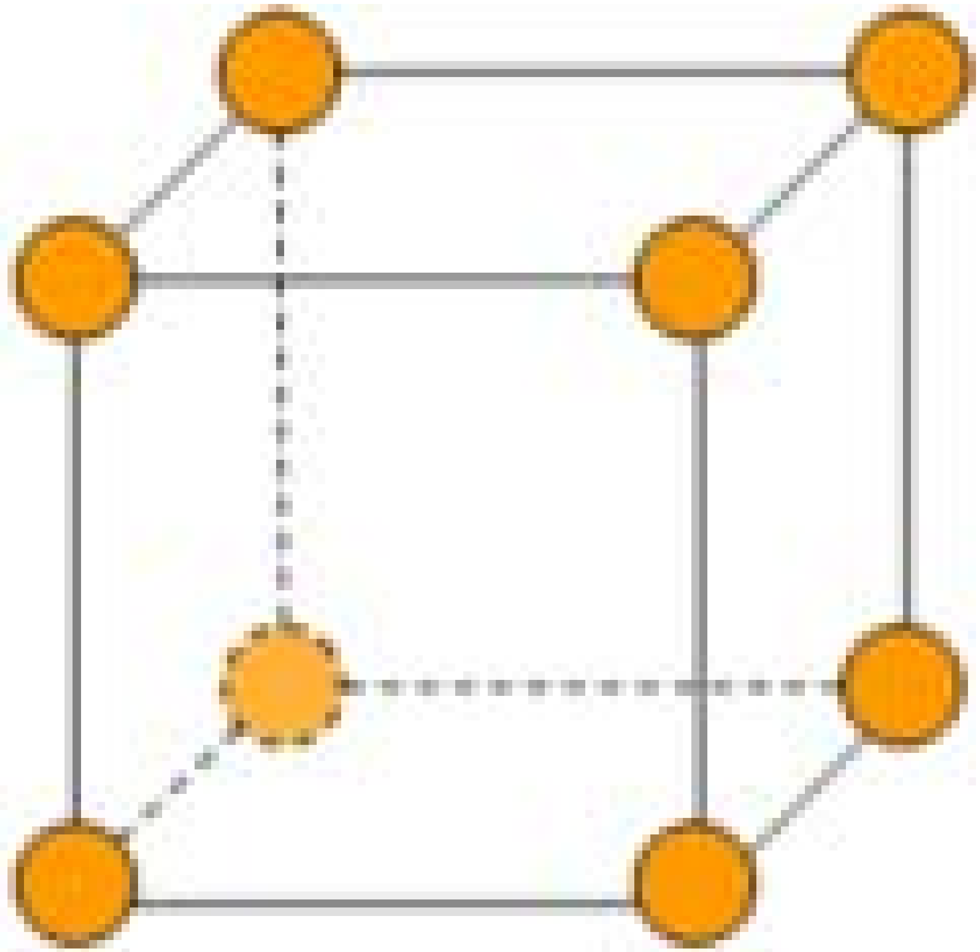
- In the first phase, the message size is *m*, the number of links is $(\sqrt{p}-1)$
- In the second phase, the message size is $m\sqrt{p}$, and the number of links is $(\sqrt{p}-1)$
- The total communication cost is the sum of both phases:

$$
\begin{aligned}
t_{comm} &= (t_s + t_w m)(\sqrt{p}-1) + (t_s + t_w m\sqrt{p})(\sqrt{p}-1) \\
&= 2t_s(\sqrt{p}-1) + t_w m(p-1)
\end{aligned}
$$

5.4g: All-to-All on Hypercube

- The all-to-all broadcast needs log *p* steps
- Communication takes place along a different dimension of the hypercube at each step
- At each step, pairs of processors exchange their data
- The size of the message to be transmitted at the next step is doubled by concatenating the received message with their current data

5.4h: All-to-All on Hypercube Illustration



5.4i: Cost of All-to-All on Hypercube

- $\log p$ steps in which message size doubles at every step
- The total communication cost is:

$$T_{comm} = \sum_{i=1}^{\log p} (t_s + t_w m 2^{i-1})$$

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

5.4j: Comments on All-to-All Broadcast

$$t_w m(p-1)$$

- This is the lower bound for the communication time of all-to-all broadcast on all network
- Each node will receive *m(p-1)* words of data, regardless of the architecture
- The hypercube algorithm cannot be mapped to run on the ring network, due to congestion
- All communication procedures are nearest neighbor communication, there is no difference between cut-through routing and store-and-forward routing

5.5a: All Reduce Operation

- Each node starts with a buffer of size *m*
- 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
- It can be done by an all-to-one reduction, followed by a one-to-all broadcast
- All-reduce operation can be used to implement barrier synchronization on a message-passing computer

5.5b: All-Reduce Implementation

- On a hypercube, one to all broadcast and all-to one reduction cost the same, the total cost of all reduce is:

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

- By performing an all-to-all broadcast and performing the associative operation after each step at each node, the message size does not increase and the total cost is:

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

5.6: Prefix Sum Operation

- Initially, each processor has a data
- Finally, each processor collect the sum of its data and the data from all processors with lower labels
- This operation can be performed by an all-to-all broadcast, with data being summed locally in each processor
- Each processor needs two copies of data, one for its own sum, the other to send out

5.7a: One-to-All Personalized Communication

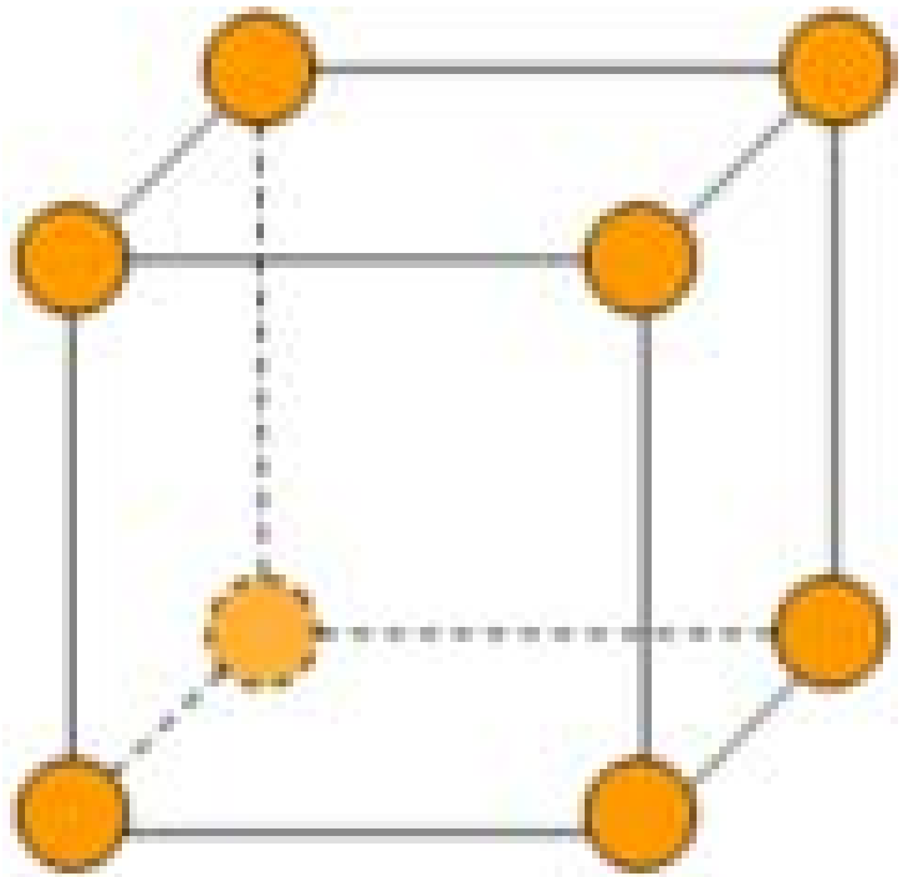
- The source node starts with p unique messages, one is intended for each node
- One-to-all personalization does not involve any duplication of data
- This operation is commonly called **scatter** operation
- The dual of the scatter operation is the **gather** operation, or **concatenation**
- No reduction operation is performed

5.7b: Implementation of Scatter Operation on Hypercube

- Use all-to-all broadcast procedure with log *p* steps
- At each step, the nodes have data send a half of their data to a directly linked node
- Each step, the size of the messages communicated is halved.
- The total communication cost is:

$$T_{comm} = t_s \log p + t_w m(p-1)$$

5.7c: Illustration of Scatter Operation



5.7d: Scatter Operations for Ring and Mesh

- The hypercube algorithm for one-to-all personalized communication can be mapped to ring and mesh networks with the same cost
- The gather operation can be performed analogously
- Note that the communication time lower bound is still:

$$t_w m(p-1)$$

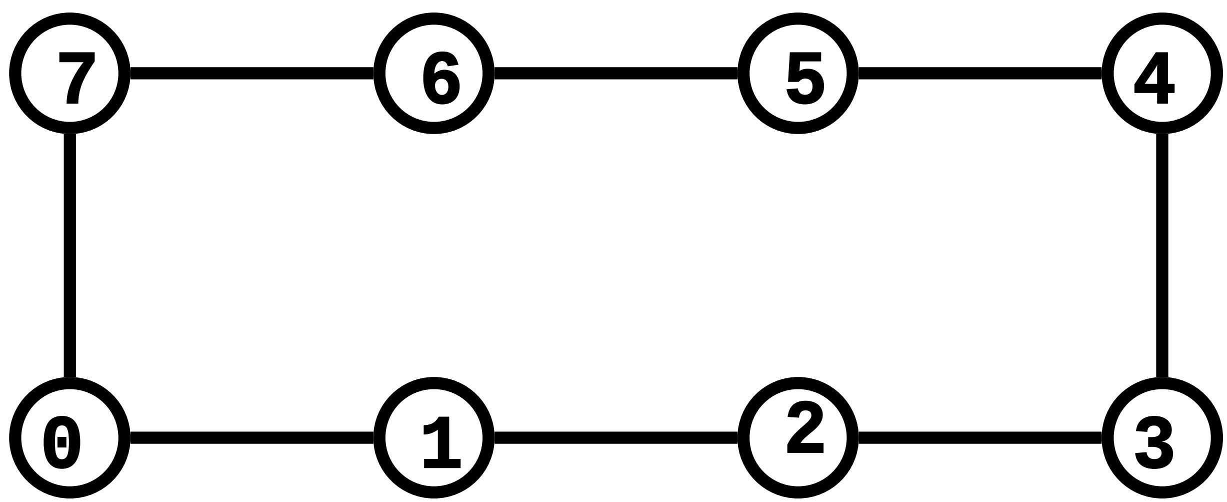
5.8a: All-to-All Personalized Communication

- Each node sends a distinct message of size *m* to every other node
- This is **not** an all-to-all broadcast operation
- All-to-all personalized communication is also called **total exchange**
- It can be used in fast Fourier transform, matrix transpose, sample sort, etc., applications

5.8b: All-to-All Personalized Communication on Ring Network

- The procedure is the same as the all-to-all broadcast, only the size of the data communicated is changed
- It uses pipelined communication, each node sends data to its neighboring node in *(p-1)* steps
- Each node receives data from its neighboring node, extracts the piece belongs to it, and forwards the remaining part to its neighboring node
- At the end of the procedure, every node has the same data of ensemble

5.8c: Illustration of All-to-All Personalized Communication on a Ring



5.8d: Cost of All-to-All Personalized Communication

- Note that there are log *p* steps of nearest neighbor communication
- At each step, the message size is reduced by *m* words. The total cost is:

$$T_{comm} = \sum_{i=1}^{p-1} (t_s + t_w m(p-i))$$

$$= t_s(p-1) + \sum_{i=1}^{p-1} i t_w m$$

$$= (t_s + t_w m p/2)(p-1)$$

5.8e: Optimality in the Ring Algorithm

- Each node sends $m(p-1)$ words of data
- The average distance of communication is $p/2$
- The total traffic on the network is

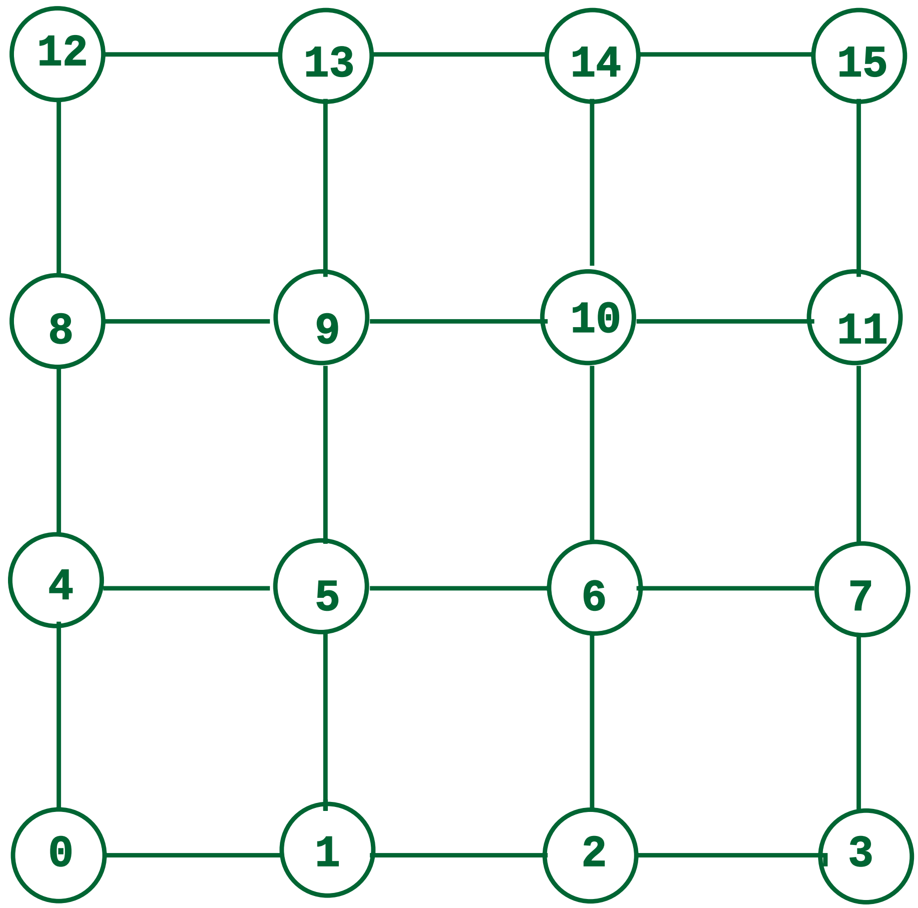
  $m(p-1)*p/2*p.$
- The total number of communication link is p
- The lower bound for the communication time is

$$(t_w \times m(p-1)p^2/2)/p = t_w m(p-1)/2$$

5.9a: All-to-All Personalized Communication on a 2D Mesh

- Using the ring algorithm twice, one with respect to the rows, another to the columns
- Each node assembles its message into $\sqrt{p}$ groups of $\sqrt{p}$ messages
- Row operation is performed simultaneously with clustered messages of size $m\sqrt{p}$
- Regroup is required after the row operation, so that regrouped messages are column oriented

5.9b: Illustration of All-to-All Personalized Communication on 2D Mesh



5.9c: Cost of All-to-All Personalized Communication on 2D Mesh

- Use ring algorithm with $\sqrt{p}$ nodes, and message size of $m\sqrt{p}$, the time of the first step is:

$$T_{comm} = (t_s + t_w mp / 2)(\sqrt{p} - 1)$$

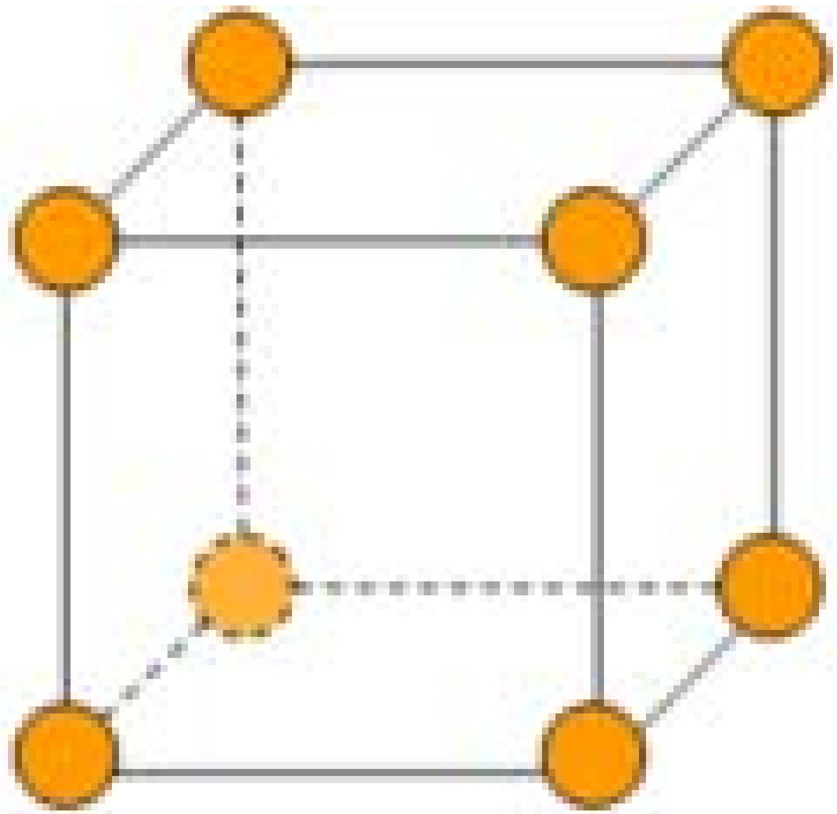
The column-wise communication step costs the same, so the total cost of all-to-all personalized communication is:

$$T_{comm} = (2t_s + t_w mp)(\sqrt{p} - 1)$$

5.10a: All-to-All Personalized Communication on Hypercube

- Communication takes place in log *p* steps
- Each step along a different communication link (different dimension)
- At each step, every node sends *p/2* of consolidated packets, meant for other half of the hypercube
- Data are re-grouped every step so that appropriate data pieces are sent to the correct nodes

5.10b: Illustration of All-to-All Personalized on Hypercube



5.10c: Cost of All-to-All Personalized Communication on Hypercube

- Log *p* directly connected node communication
- At each step, a half of the *p* pieces of data are exchanged
- The total cost is:

$$T_{comm} = (t_s + t_w mp / 2)\log p$$

5.10d: Non-optimality in Hypercube algorithm

- Each node send m(p-1) words of data
- Average distance of communication (log *p*)/*2*
- Total network traffic is $p*m(p-1)*(\log p)/2$
- Total number of links is (*p* log *p*)/*2*
- The lower bound for communication time is

$$T = \frac{t_w pm(p-1)(\log p)/2}{(p \log p)/2} = t_w m(p-1)$$

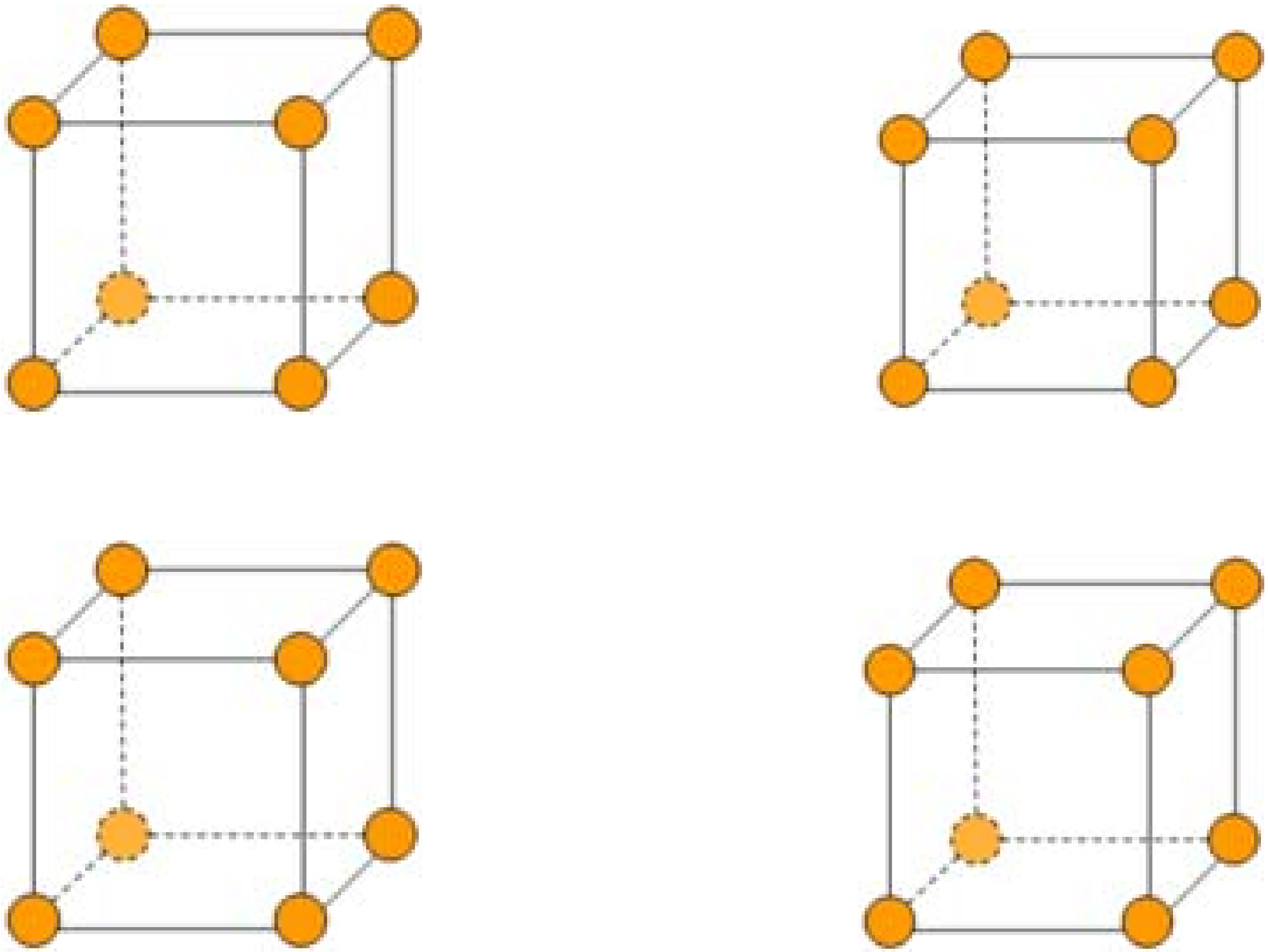
5.10e: Optimal Algorithm for All-to-All Personalized Hypercube Communication

- Allows all pairs of nodes exchange some data
- Every pair of nodes directly communicate with each other, with a total of (*p-1*) steps
- A congestion-free schedule can be used

1.) At the *j*-th step communication, node *i* exchanges data with node (*I* XOR *j*)

2.) Use E-cube routing scheme to establish communication paths

5.10f: Illustration of Optimal Algorithm



5.10g: Cost of the Optimal Algorithm

- Based on the communication pattern, the cost of the optimal algorithm, with (*p-1*) pair-wise communication is:

$$T_{comm} = (t_s + t_w m)(p - 1)$$

The start-up time term has a larger factor, but the per-word time has a smaller factor.
For large size message communication, this algorithm is seen to be better