Midterm 1 Review

Scott Cheng

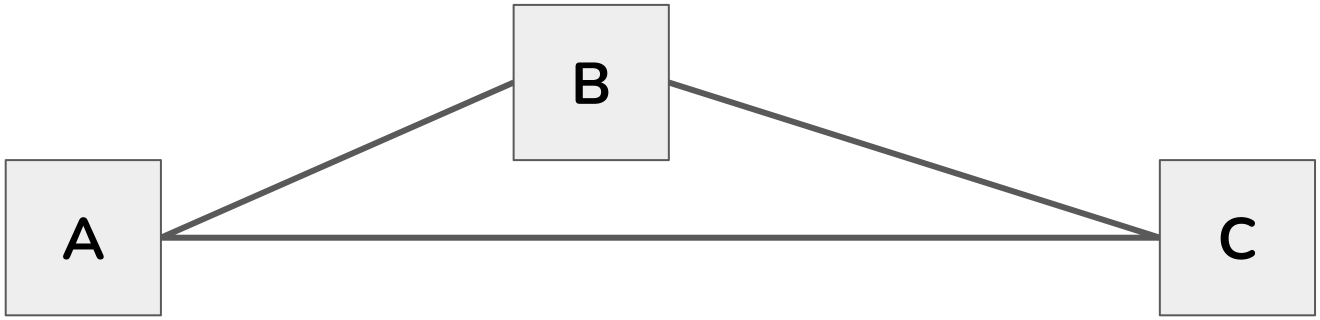
Jamboard Link:
https://jamboard.google.com/d/1fXiHXHQgxWwHwdqLH8SBhlsYSnphPKfS88Ed11Njmzg/edit?usp=sharing

Midterm 1

- slide-group-1 ~ 7
- (openmp-supp, mpi-supp)

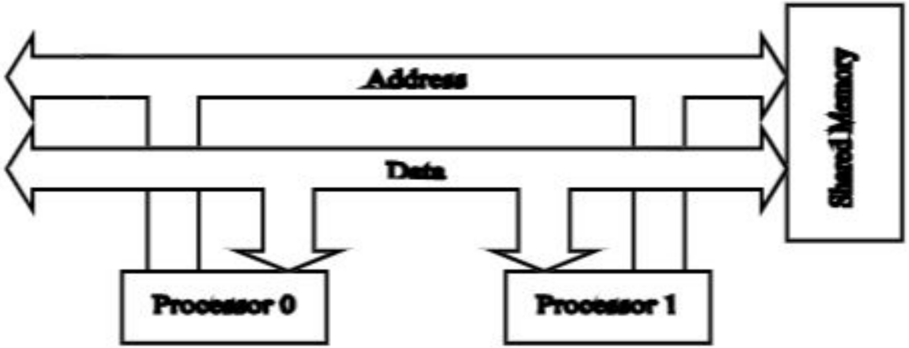
Terms related to Interconnection Networks

- Distance between two nodes
- Diameter
- Bisection Width
- Arc Connectivity



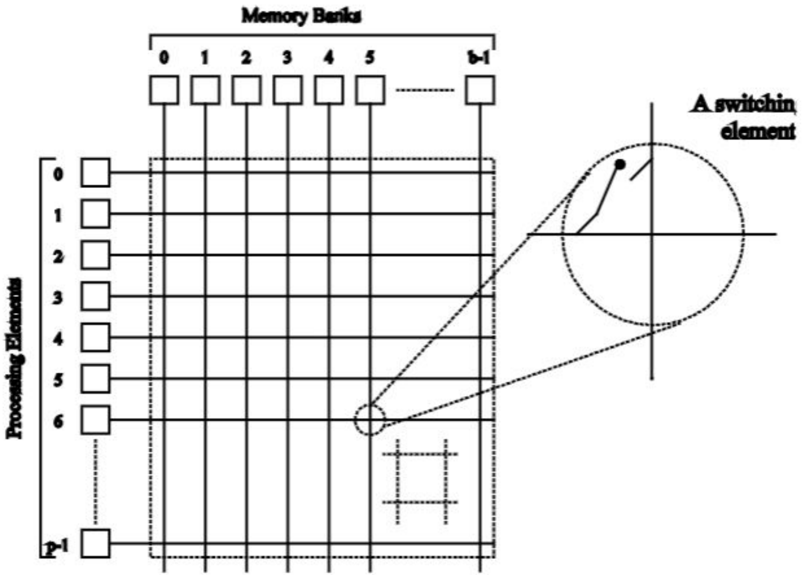
Network Topologies: Buses

- What's the distance between any two nodes?
- Bandwidth of the shared bus is a major bottleneck
  - => How to improve?



Network Topologies: Crossbars

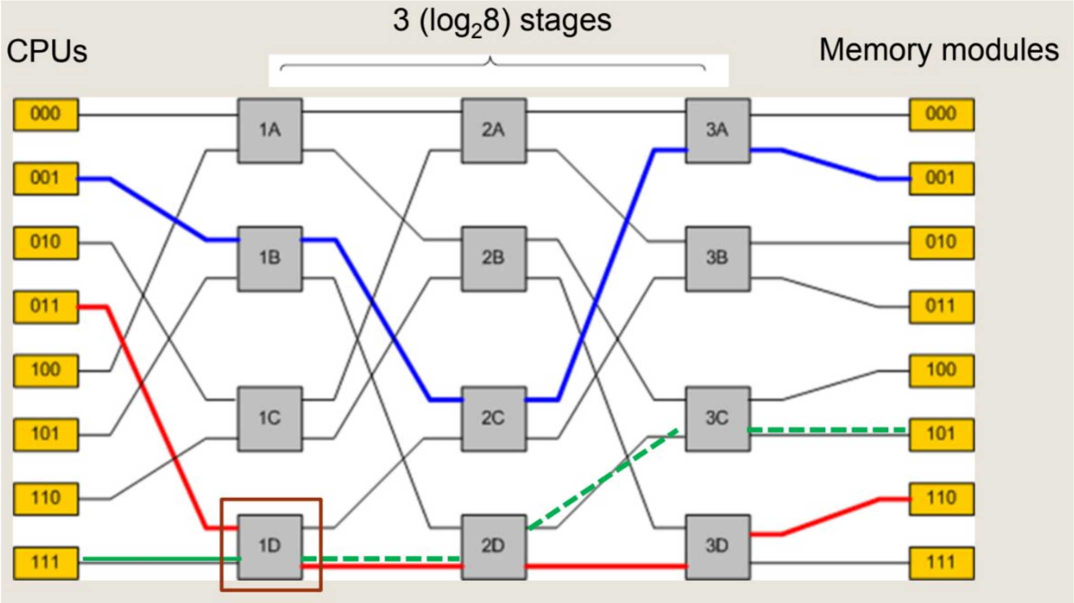
- The cost of a crossbar of p processors grows as `___O(?)___`
- Blocking or non-blocking?



Network Topologies: Multistage Omega Networks

- The cost of an omega network of p processors grows as ___O(?)___
- Blocking or non-blocking?

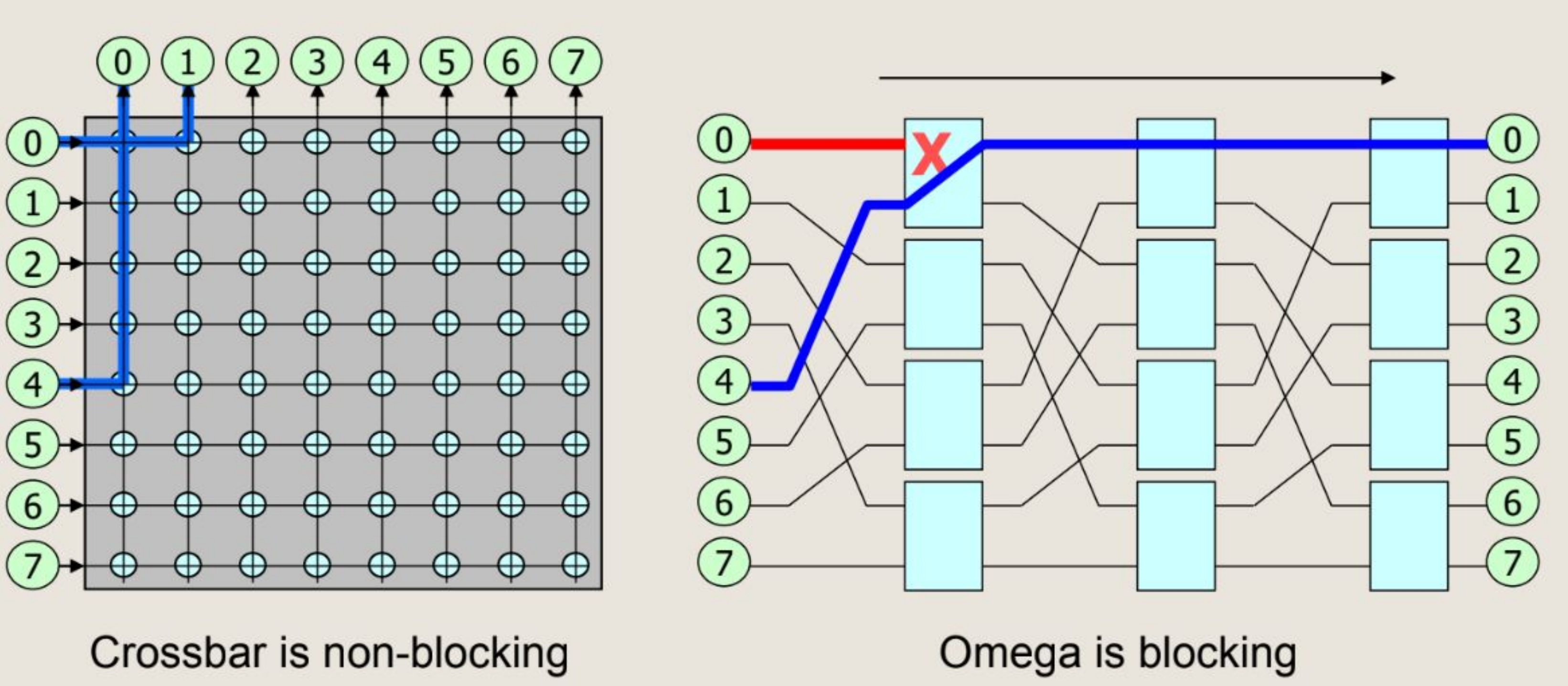
Example: an 8x8 Omega Network (with 2x2 crossbars)



- log2(p) = 3 stages
- each stage has ? crossbars
- total number of crossbars?
- total number of links?

In general, multistage networks are blocking

Can you construct a non-blocking multistage network?



Butterfly?

Network	Diameter	Bisection Width	Arc Connectivity	Cost (No. of links)
Completely-connected	1	$p^2/4$	$p-1$	$p(p-1)/2$
Star	2	1	1	$p-1$
Complete binary tree	$2\log((p+1)/2)$	1	1	$p-1$
Linear array	$p-1$	1	1	$p-1$
2-D mesh, no wraparound	$2(\sqrt{p}-1)$	$\sqrt{p}$	2	$2(p-\sqrt{p})$
2-D wraparound mesh	$2\lfloor\sqrt{p}/2\rfloor$	$2\sqrt{p}$	4	$2p$
Hypercube	$\log p$	$p/2$	$\log p$	$(p\log p)/2$
Crossbar	1	p	1	p^2
Omega Network	$\log p$	$p/2$	2	$p/2$
Dynamic Tree	$2\log p$	1	2	$p-1$

Performance Metrics for Parallel Systems: Total Parallel Overhead

- Let T_{all} be the total time collectively spent by all the processing elements.
- T_S is the serial time.

T_p = ?

The overhead function:

T_o = ?

Speedup:

S = ?

Efficiency:

E = ?

Cost Optimality?

Isoefficiency function

Problem size W is defined as the asymptotic number of operations associated with the best serial algorithm to solve the problem.

$$W = KT_o(W, p)$$

Example: Sorting

Consider a sorting algorithm that uses ***n*** processing elements to sort the list in time (**log *n***)^2

- Since the serial runtime of a (comparison-based) sort is ***n* log *n***, the speedup and efficiency of this algorithm are given by __?__ and __?__ respectively.

- The ***p T_P*** product of this algorithm is __?__

Cost Optimal?

Example: Adding n numbers

Consider the minimum execution time for adding ***n*** numbers.

$$T_P = \frac{n}{p} + 2\log p.$$

$t_s + t_w w.$

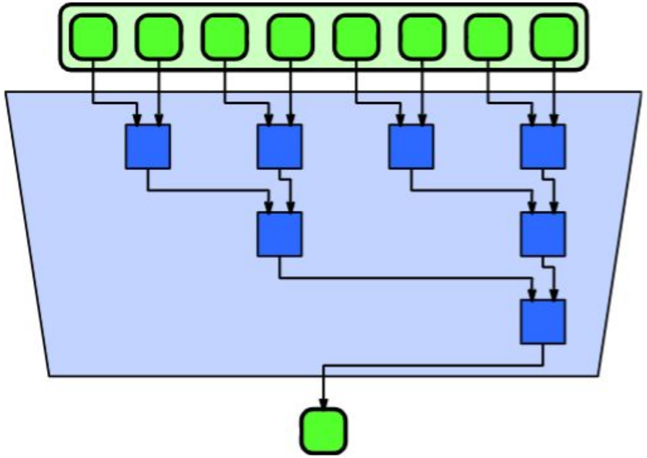
T_o = ?

Isoefficiency function?

T_P^{min} = ?

$T_P^{cost\_optimal}$ = ?

Reduction



Examples: averaging of Monte Carlo samples; convergence testing; image comparison metrics; matrix operations.

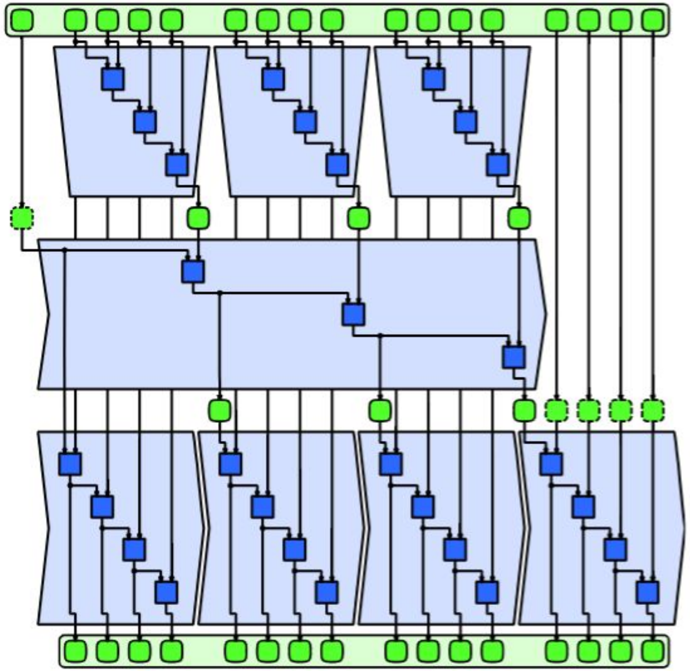
Example: Adding N numbers

- *Reduction* combines every element in a collection into one element using an *associative* operator.

```
b = 0;
for (i=0; i<n; ++i) {
    b += f(B[i]);
}
```

- Reordering of the operations is often needed to allow for parallelism.
- A tree reordering requires associativity.

Scan



Examples: random number generation, pack, tabulated integration, time series analysis

Example: Prefix sum

- *Scan* computes all partial reductions of a collection

```
A[0] = B[0] + init;
for (i=1; i<n; ++i) {
 A[i] = B[i] + A[i-1];
}
```

- Operator must be (at least) associative.
- Diagram shows one possible parallel implementation using three-phase strategy
- We'll consider different implementations later

Example: FFT

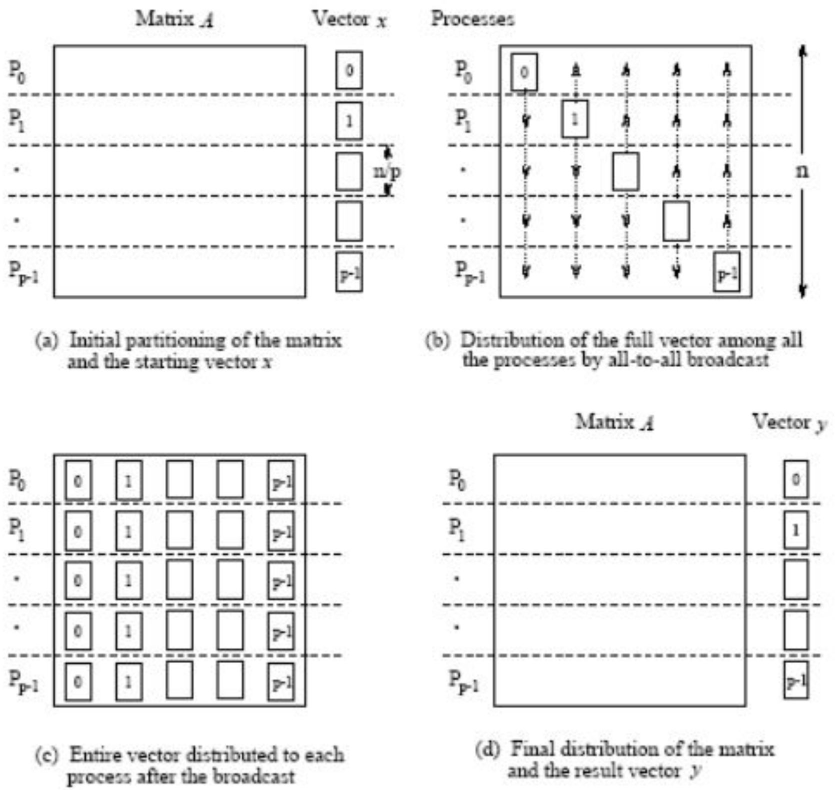
The parallel runtime of a parallel implementation of the FFT algorithm with p processing elements is given by Tp = (n/p) log n + 10(n/p) log p for an input sequence of length n.

Cost Optimal?

$T_P^{min} = ?$

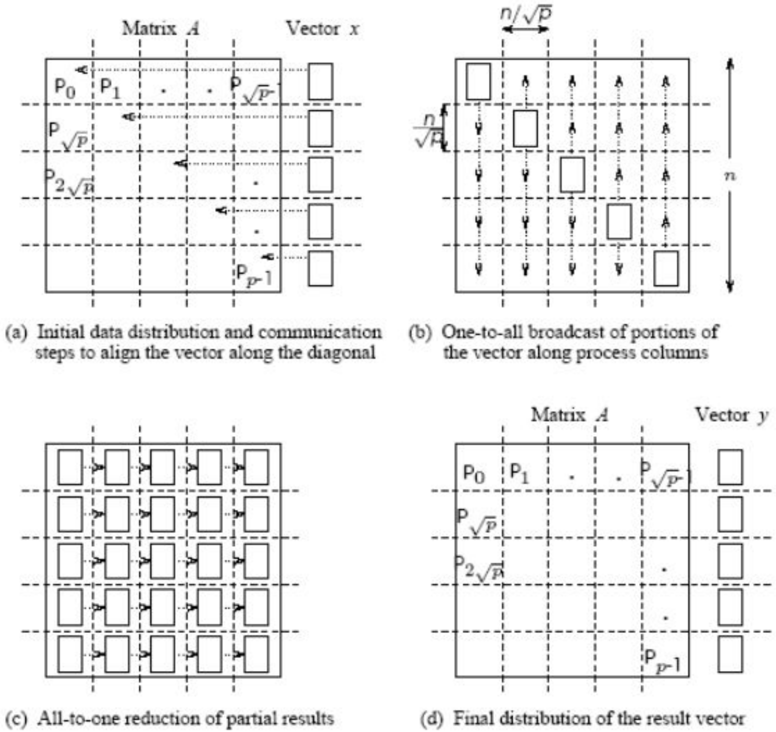
Example: Matrix Multiplication (Rowwise 1D Partitioning)

$$T_P = \frac{n^2}{p} + t_s \log p + t_w n$$



Example: Matrix Multiplication (2D Partitioning)

$$T_P \approx \frac{n^2}{p} + t_s \log p + t_w \frac{n}{\sqrt{p}} \log p$$



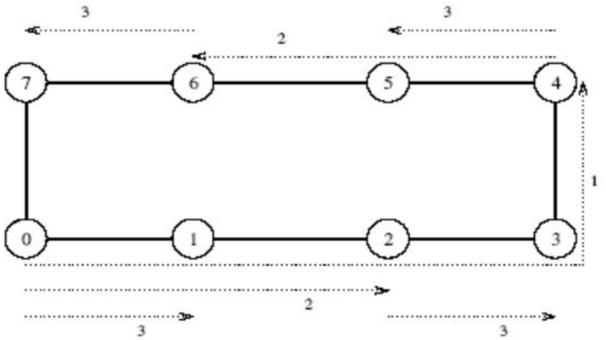
Communication Patterns

- One-to-All Broadcast and All-to-One Reduction
- All-to-All Broadcast and Reduction
- All-Reduce and Prefix Sum
- Scatter and Gather
- All-to-All Personalized Communication
- Circular Shift

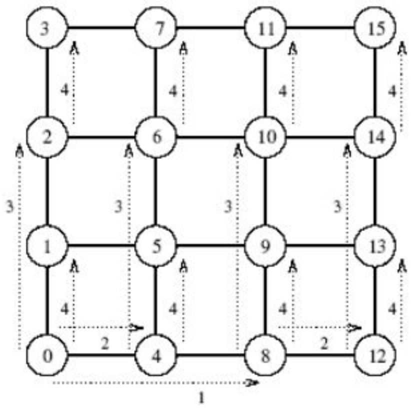
One-to-All Broadcast and All-to-One Reduction

- Ring: recursive doubling

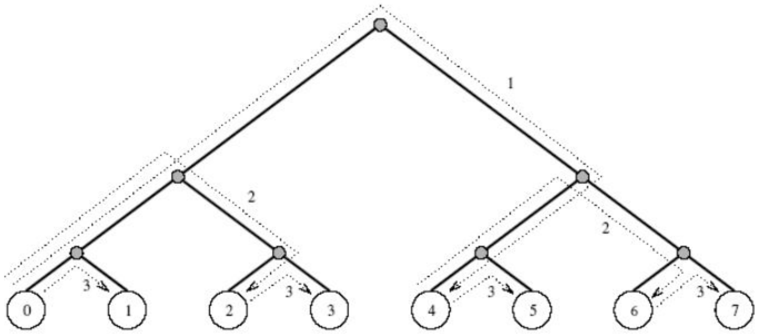
$$T = (t_s + t_w m)\log p$$



- Mesh/Hypercube



- Balanced binary tree

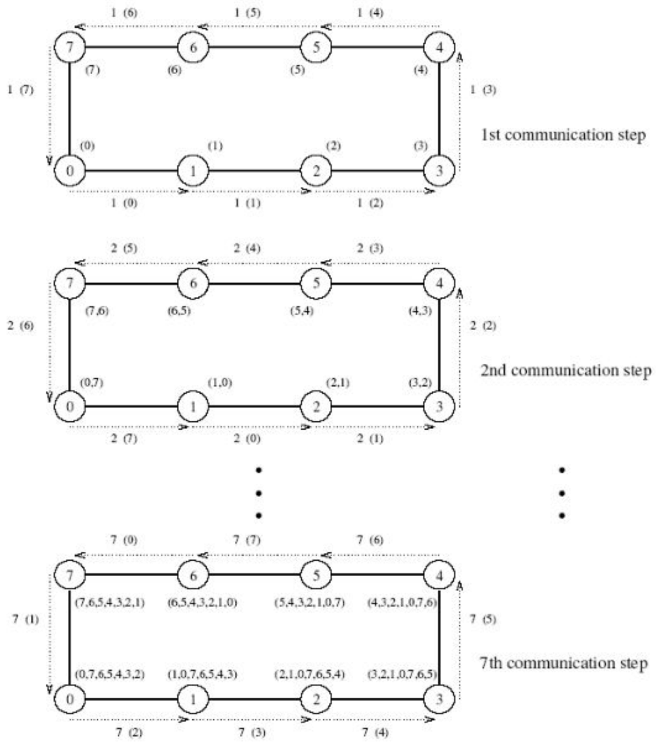


All-to-All Broadcast and Reduction

- On a ring, the time is given by: $(t_s + t_w m)(p-1)$.
- On a mesh, the time is given by: $2t_s(\sqrt{p} - 1) + t_w m(p-1)$.
- On a hypercube, we have:

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

- Balanced binary tree $(t_s + t_w mp/2) \log p$



Embedding

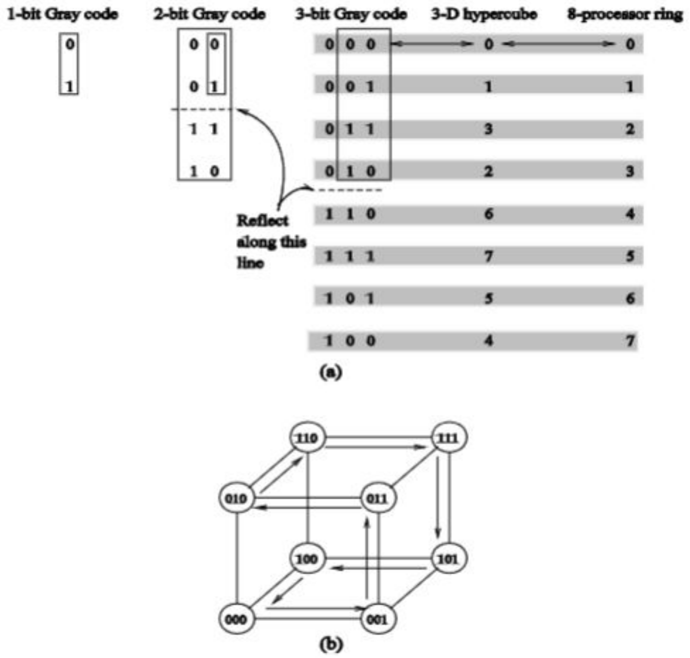
```
G(V, E) -> G’(V’, E’)
```

- Congestion?
- Dilation?
- Expansion?

Embedding

- Embedding a ring into a hypercube
    - binary reflected gray code (RGC)
- Embedding a mesh into a hypercube
    - Concatenated binary RGC
- Embedding a mesh into a ring
- Embedding a hypercube into a mesh?
- Embedding a complete binary tree into a hypercube?
- Embedding a mesh of trees into a hypercube?
- Embedding a ring into a complete binary tree?

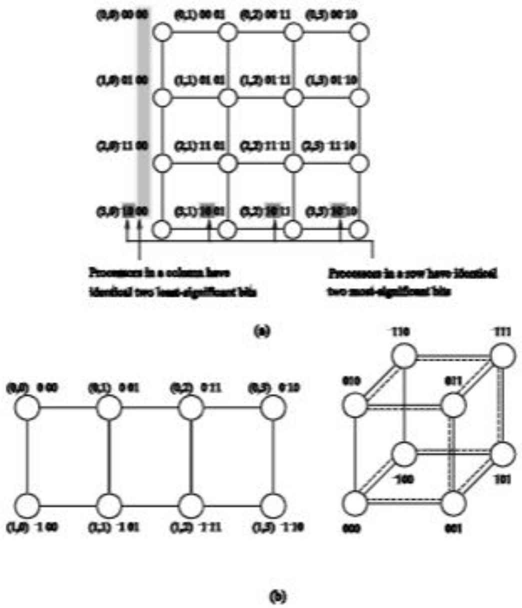
Embedding a Linear Array into a Hypercube: Example



(a) A three-bit reflected Gray code ring; and (b) its embedding into a three-dimensional hypercube.

- Congestion?
- Dilation?

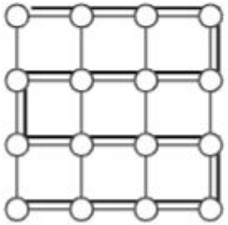
Embedding a Mesh into a Hypercube



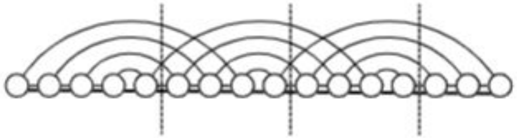
- Congestion?
- Dilation?

(a) A 4 × 4 mesh illustrating the mapping of mesh nodes to the nodes in a four-dimensional hypercube; and (b) a 2 × 4 mesh embedded into a three-dimensional hypercube.

Embedding a Mesh into a Linear Array: Example



(a) Mapping a linear array into a 2D mesh (congestion 1).



(b) Inverting the mapping - mapping a 2D mesh into a linear array (congestion 5)

(a) Embedding a 16 node linear array into a 2-D mesh; and (b) the inverse of the mapping. Solid lines correspond to links in the linear array and normal lines to links in the mesh.

- Congestion?
- Dilation?