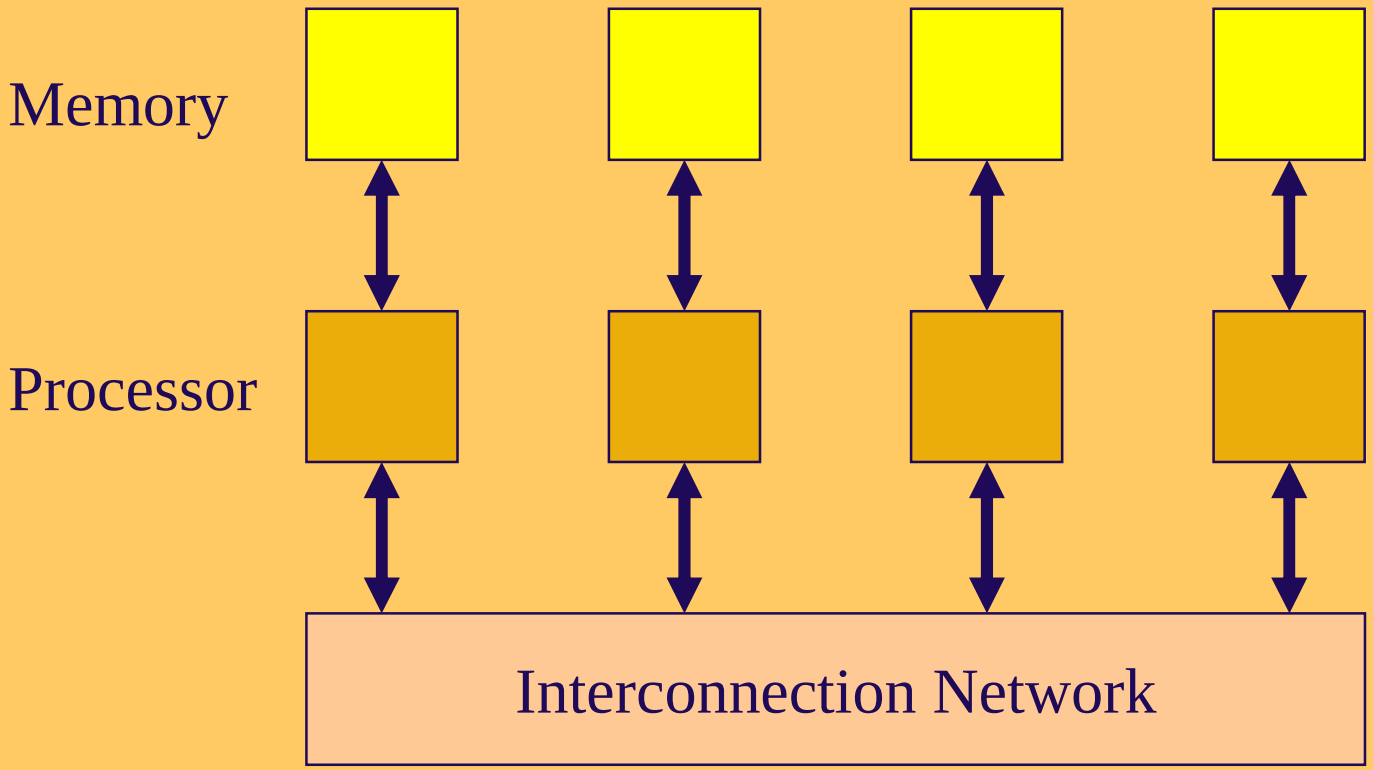
Communication Optimization for Parallel Processing

Lecture 2

Pangfeng Liu, Department of Computer Science and Information Engineering, National Taiwan University.

Distributed Memory Model

- Each processor has its own memory, which cannot be seen by other processors.
- The processors are connected by an interconnection network. For the time being, we assume that the interconnection network provides point-to-point data transmission.


Memory
Processor
Interconnection Network

Cluster Computing

- Cluster computing is a cost-effective way to provide high performance with limited costs.
- A cluster can be viewed as “poor man’s parallel machines”.
- If the processors has different communication and computation capability, the cluster is *heterogeneous;* otherwise it is *homogeneous*.
- Heterogeneity introduces additional complication into distributed memory computation.

Heterogeneity

- A heterogeneous cluster imposes challenge in the following issues.
  - Workload balancing
  - Collective Communication

Data Partitioning

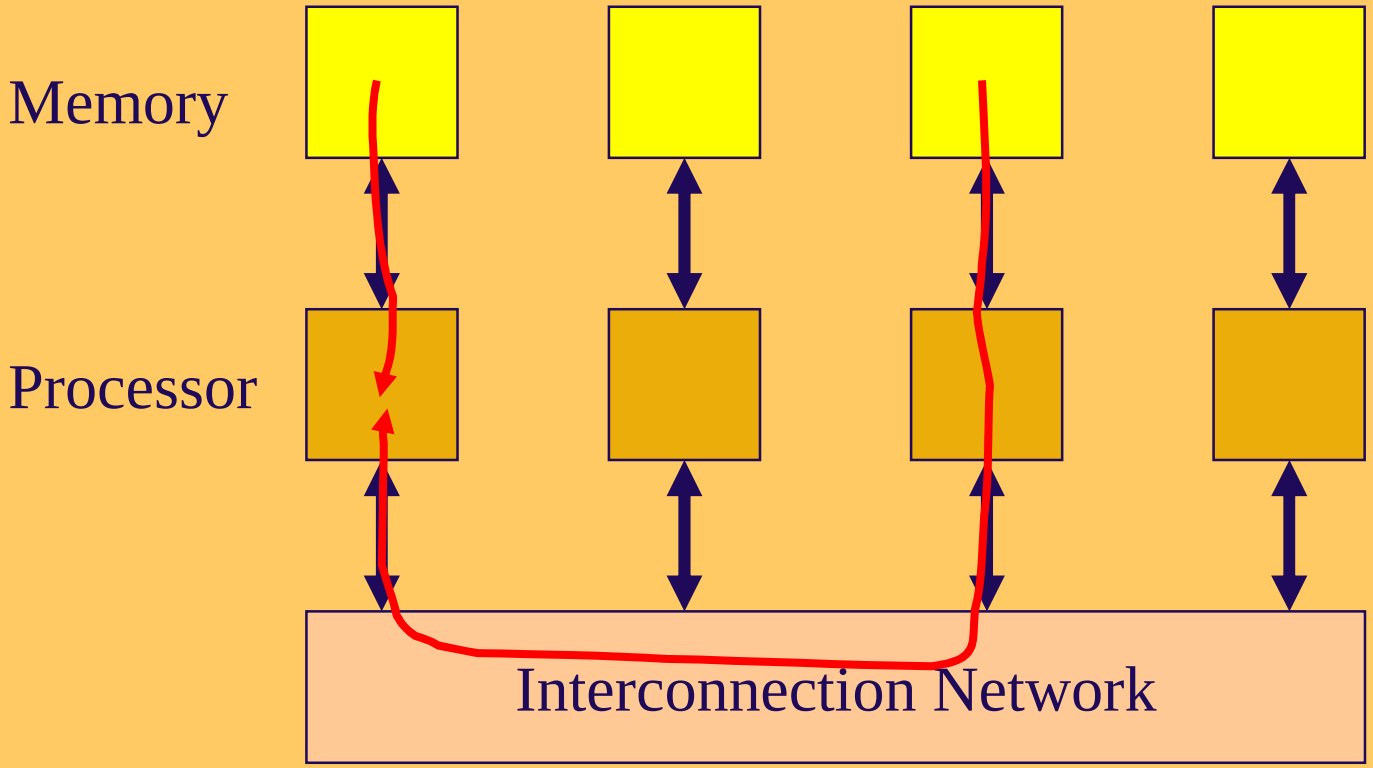
- One of the most important issues in distributed memory parallel computing is the partition of data.
- Two issues in data partitioning
  - Data locality
  - Load balancing
- Data locality indicates the easy of access for a processor to its data. Since now there is no shared memory, data must be distributed among processors.

Local and Remote Data

- The local data are with the processor's local memory.
- The remote data are located in the memory of some other processors.
- The latency of accessing remote data is tremendous.

Load Balancing

- To distribute the data/work evenly among processors.
- There are two measurements of this criterion.
  - The number of data assigned to each processor is roughly the same.
  - The amount of computation of each processor is roughly the same.

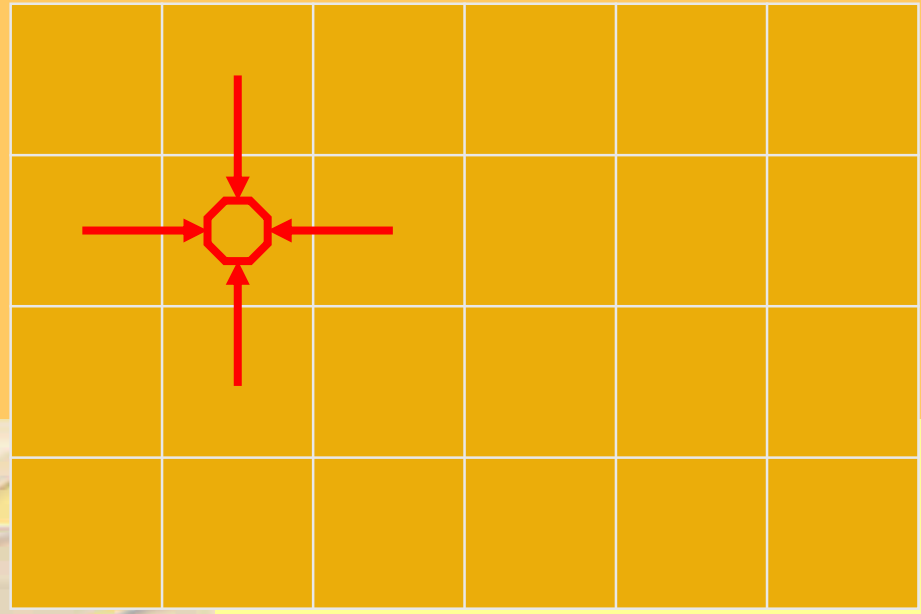

Memory
Processor
Interconnection Network

Data Locality and Load Balancing

- To put all data in one processor has the highest data locality, but load is not balanced.
- To randomly distribute the data to processors produces balanced load, but loses all data locality.

Graph Relaxation

- Compute the attributes of a pixel with the attributes of the neighbors of this pixel.



Data Partitioning for Graph Relaxation

- The data should be distributed to processors in contiguous blocks, not fragments.
- Partitioning methods for arrays.
  - Row partition
  - Column partition
  - Round-robin partition
  - Block partition

Partitioning Issues

- Data locality
  - To have good data locality, data next to each other should be assigned to the same processor as much as possible.
- Load balancing
  - To have balanced load, the processors should be assigned the same number of rows, columns, or blocks, depending on the partitioning method used. The data may be duplicated before the computation starts.

Remote Data Access

- To collect all the data beforehand so that the computation is not slowed down by the computation.
- The simplest form is to duplicate the boundary data before computation.

Remote Data Access

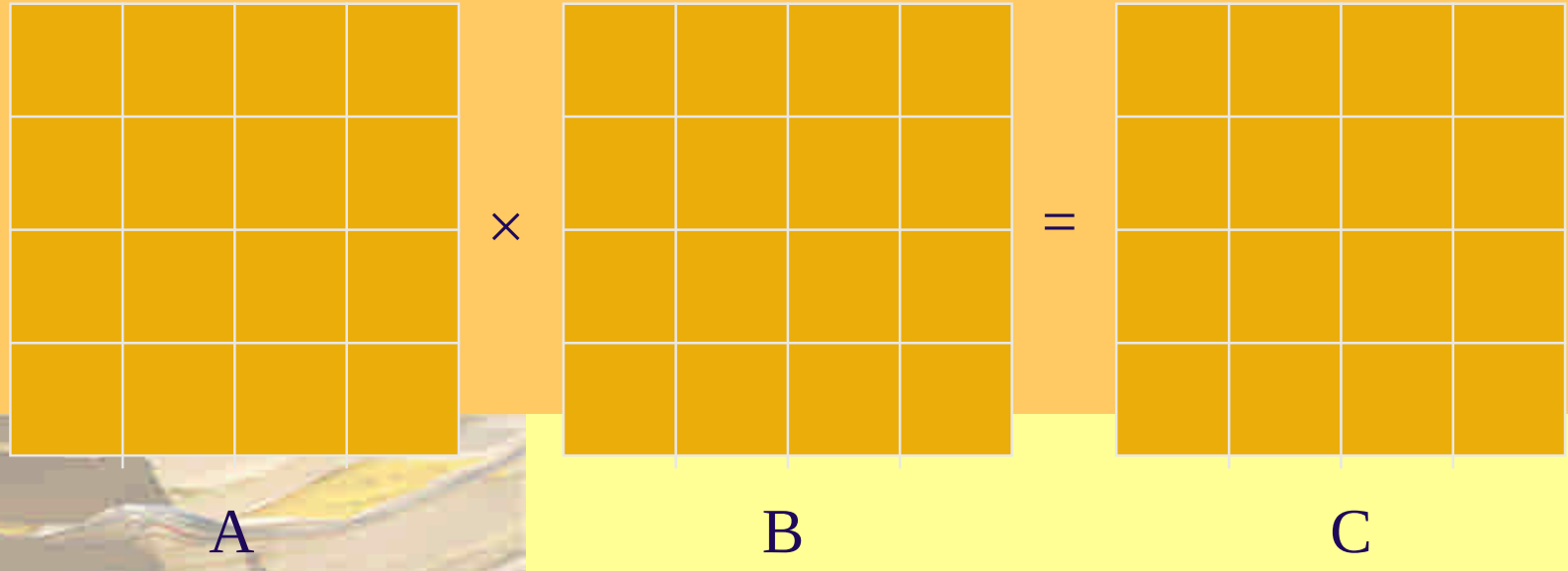
- Inspector and executor approach
  - The inspector “test-run” an application to know the remote data access pattern, and produces a schedule so that the runtime system can retrieve the remote data.
  - The executor, after gathering all the essential data, can proceed to perform computation.

Matrix Multiplication

- Due to the nature of matrix multiplication, it seems natural that in the multiplication of A X B, A should be row partitioned, and B should be column partitioned.

Matrix Decomposition

- How to partition A, B, and C into 4 processors?



Owner-Computes Rule

- The owner of a data compute the new value.
- By assigning the same number of data to each processor we can have the additional benefit that computation is also evenly distributed.
- A more nature approach since additional data transmission is unnecessary.

Message Passing

- Distributed memory machines use message passing for the following purposes.
  - Transmission of data
  - Synchronization

Message Passing Costs

- The latency of message passing incurs much higher costs than shared memory; therefore it is vital to reduce the amount of communication.
  - This is major reason we have this course.
- The goal of reducing remote data access (i.e., communication) coincides with the objective of data locality.

Message Passing Library

- The message-passing library is very similar to network communication. In fact many message-passing libraries are built on top of TCP/IP.
- Message-passing programming is not as intuitive as shared memory, but it has the advantage that no processors will update the same data, so no synchronization on data access is required.
- Due to the phased nature, either due to collecting data or heterogeneity of processors, synchronization is often required.

Parallel Prefix

- The prefix operation is that when given a sequence x_i, compute s_i so that $s_i = x_0 \; op \; x_1 \; ... \; op \; x_{i-1}$
- When the *op* is addition, the prefix operation computes partial sums for the entire sequence of numbers.
- Prefix is easily computed in $O(n)$ sequentially.
- Parallel prefix is to compute prefix in parallel.

Application

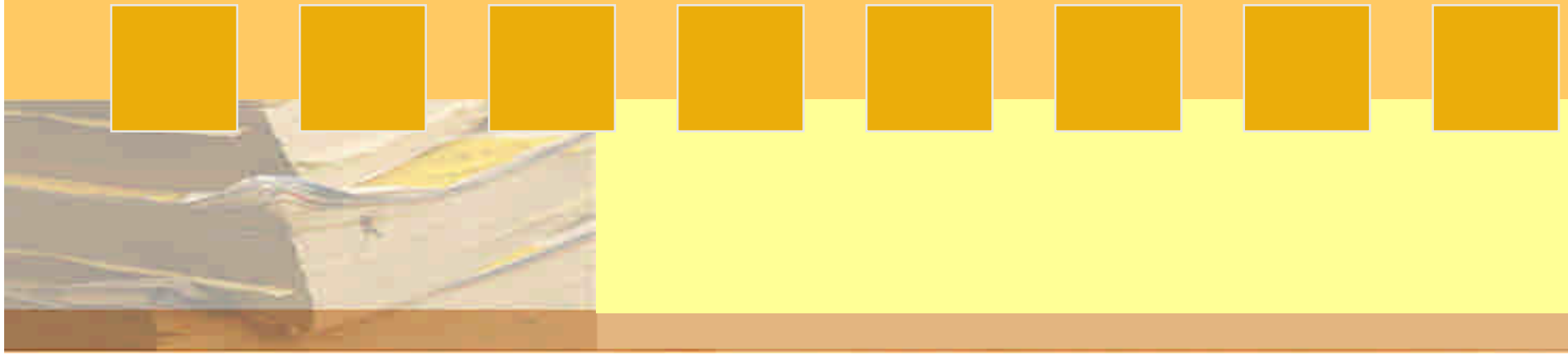
- When a sparsely occupied array wants to be compacted, the new index can be computed from prefix, where each occupied cell has an *x* to be 1, and an empty one 0.

A Naïve Algorithm

- Each processor computes an *s* value has time complexity $O(n)$ and cost $O(n^2)$, clearly not a work optimal solution.

A Better Algorithm

- Receive data from the neighbor that is 2^i to your left, and combine it with your own data.
- Repeat the process for log N times while increasing i by 1 for each iteration.



Caveat

- The operator must be associative.
  - (A op B) op C is equal to A op (B op C).
- All the sends must synchronize with the receives.

A Cost Optimal Algorithm

- Use only n/log n processors, so each processor handles log n data.
- Each processor computes the prefix for the data it has.
- All processor collectively compute the prefix using the last prefix result from the previous step.
- Each processor “patches” the prefix result into the final answers.

An Illustration

Analysis

- The total time is *O(log n)*.
  - Computation
    - *O(log n)*
  - Communication
    - *O(log n)*
- Cost is optimal.

Numerical Integration

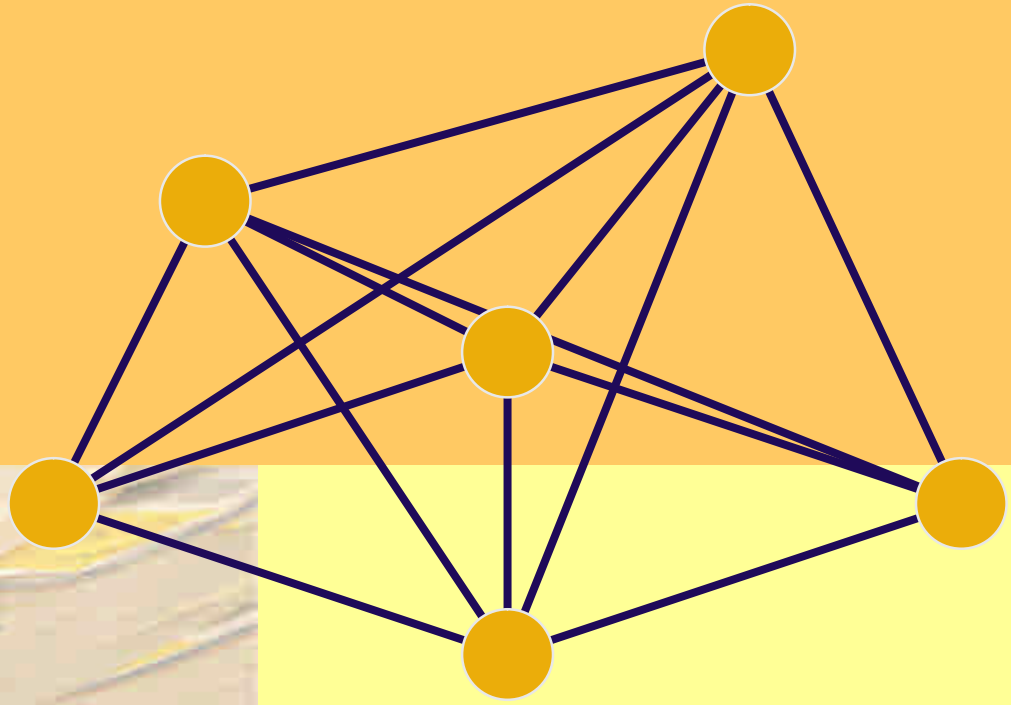
- The domain is partitioned into segment so that each processor is responsible for computing the integral of a segment.
- Finally the answer is sent from each processor to be summed up.

Issues

- Each processor must know the starting and end points of its segment.
- The error can be controlled by a method called adaptive quadrature, which doubles the number of sub-intervals if the results from two successive integrations differ more than a predefined error bound.

N-body Method

- Gravitational N-body problem computes the interaction among N particles in space.



Approximation

- A direct method requires N^2 interaction and is not computational possible for large problem sizes.
- An approximate method uses the center of mass to approximate the effect of a cluster, and can reduce the time complexity to NlogN (for uniform distribution).
- The approximation can be applied only when the cluster is very far way from where the effect is measured.