ADVANCED ALGORITHMS · FALL SEMESTER

Lecture 5

Parallel Algorithms

Faculty: K.R. Chowdhary

Professor of CS

The RAM and PRAM models, multiprocessor architectures, and interconnection networks

From Sequential to Parallel

Most present algorithms are sequential — a single sequence of steps, each a single operation. But hardware has moved steadily toward parallelism, starting with multi-core systems. A parallel algorithm performs multiple operations in a single step.

Worked example — summing n numbers

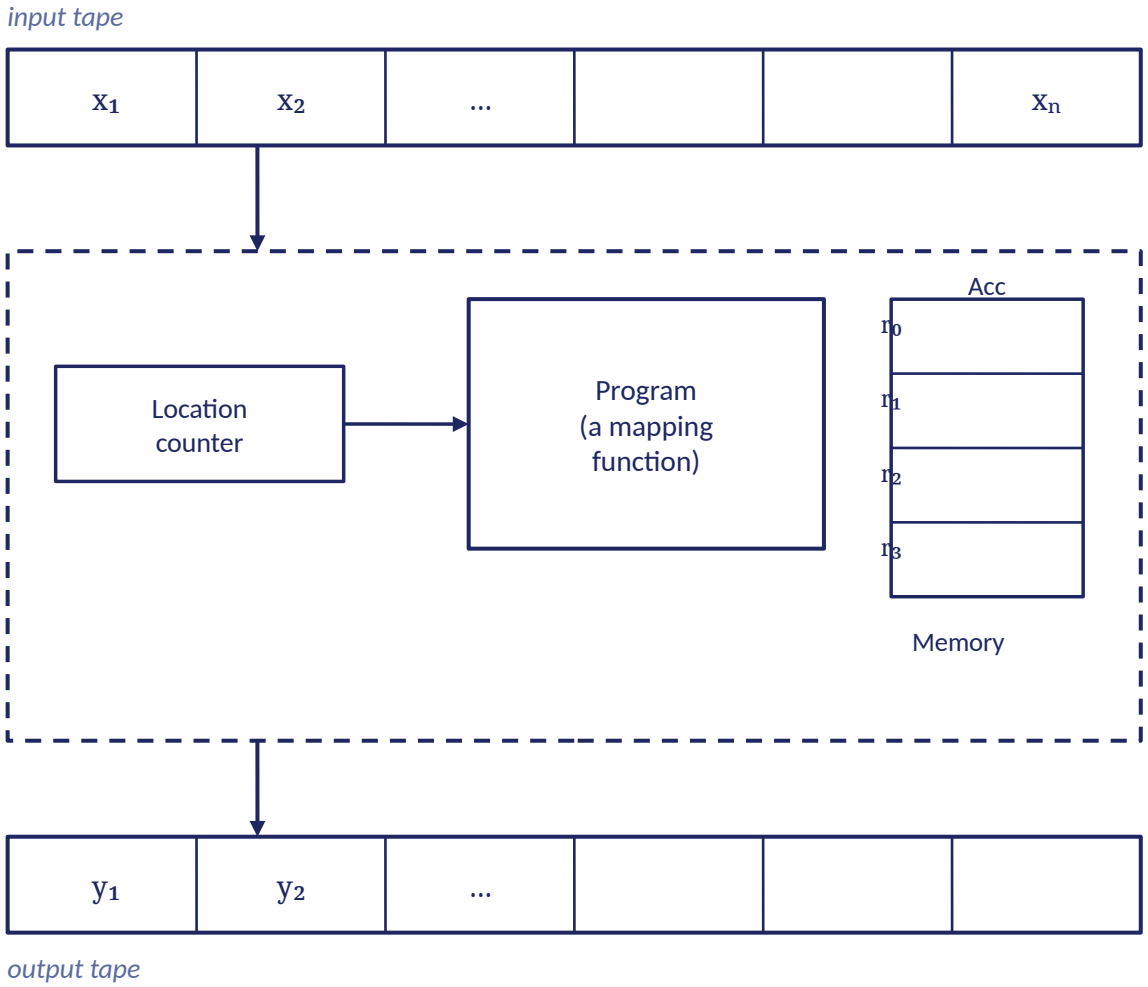
- Sequential: a single pass, keeping a running sum — $n-1$ steps.
- Parallel: pair A[0] with A[1], A[2] with A[3], … — $\lceil n/2 \rceil$ sums computed simultaneously, in one step.
- Repeating the pairing gives $\lceil n/4 \rceil$ sums, then $\lceil n/8 \rceil$, … — the whole sum completes in only $\lceil \log_2 n \rceil$ steps.

A good parallel algorithm should run efficiently on both parallel and sequential architectures — and will run at least as fast as the number of processing units available.

The RAM Model

Random Access Machine

- Model for a sequential algorithm: a single processor connected to a random-access memory.
- Every arithmetic and logical operation takes one time step.
- Models a one-accumulator computer; instructions cannot modify themselves.
- Memory locations r_0, r_1, … each hold an integer of arbitrary size.
- Instruction set: add, sub, mul, div, load, store, read, write, j, jnz, jz, hlt.



Multiprocessor Machine Models

Modeling parallel computation is harder than sequential — real machines vary widely, and there is no consensus on a single right model. Multiprocessor models generalize RAM with more than one processor, in three basic types:

(a) Local memory

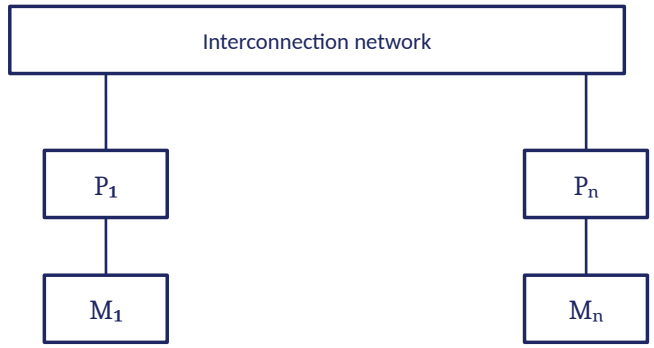
Each processor accesses its own memory directly; others require a network request.

(b) Modular memory

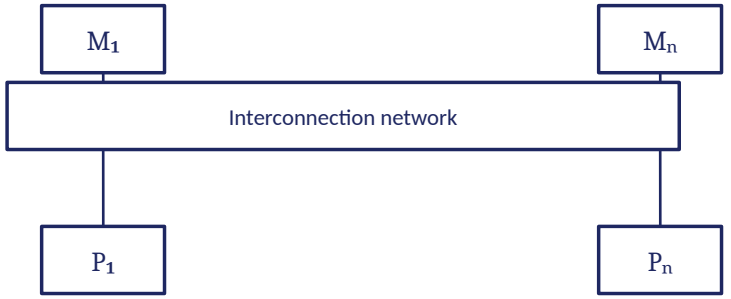
Any processor can access any memory module via the network, with uniform access time.

(c) PRAM

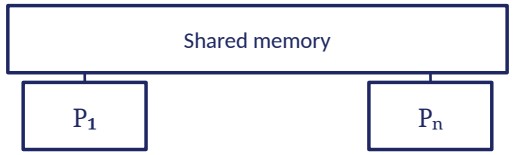
All processors share a single global memory, accessible simultaneously by many.



(a)



(b)



(c)

The PRAM Model

Parallel Random Access Machine

Introduced by Fortune and Wyllie (1978) to model idealized parallel computers where communication cost and synchronization overhead are negligible. A PRAM has a control unit and a global memory shared by p processors P_1, ..., P_p — each also has its own private memory.

THE READ – COMPUTE – WRITE CYCLE

All p processors operate in lockstep. In each step, an active processor may read a value, perform one operation, and write the result back — all active processors execute the same instruction, generally on different data (hence “shared-memory SIMD”).

Atomic access

PRAM guarantees only one memory access at a time is permitted, so algorithms run without interference. An operation is atomic if it completes entirely, or not at all.

Read and Write Modes in a PRAM

PRAM variants are distinguished by how concurrently processors may access the same memory location:

ER

Exclusive Read

Only one processor may read from any given memory location at a time.

EW

Exclusive Write

Only one processor may write to any given memory location at a time.

CR

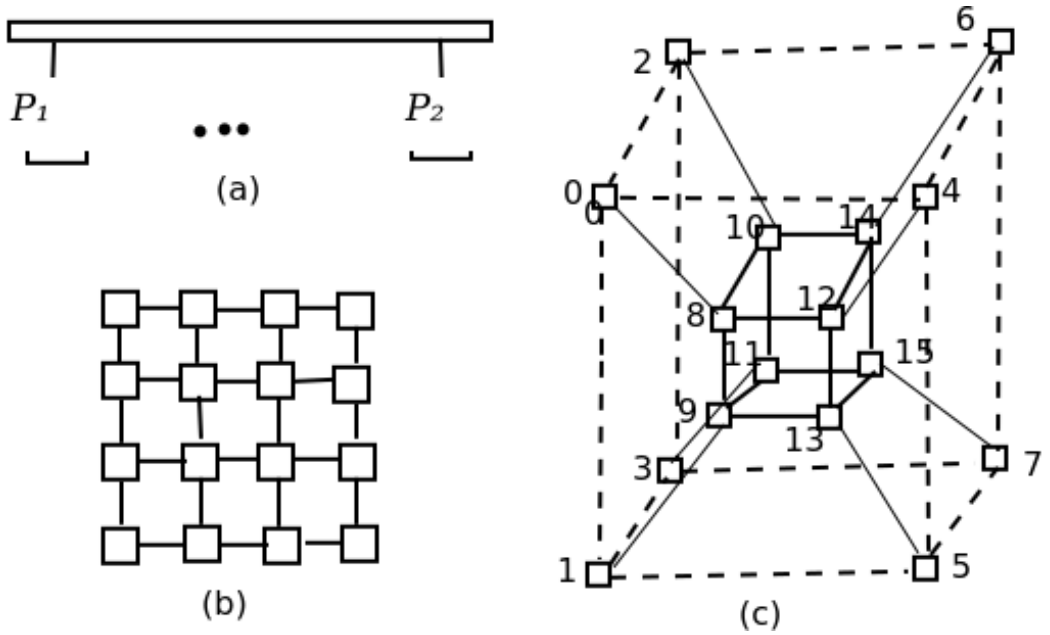
Concurrent Read

Multiple processors may read from the same memory location simultaneously.

Combinations like CREW (concurrent read, exclusive write) and CRCW (concurrent read, concurrent write) define common PRAM sub-models used to classify algorithms by the access discipline they assume.

Bus, Mesh, and Hypercube

A network is a collection of switches connected by communication channels; processors and memory attach to these switches. Topologies include bus, mesh, hypercube, multistage, and fat-tree networks.



(a) Bus (b) 2-D mesh (c) Hypercube (16 switches, 4-bit labels)

Bus

Simplest topology; only one piece of data per step. Simple to build, but processors must take turns — average access time grows with processor count.

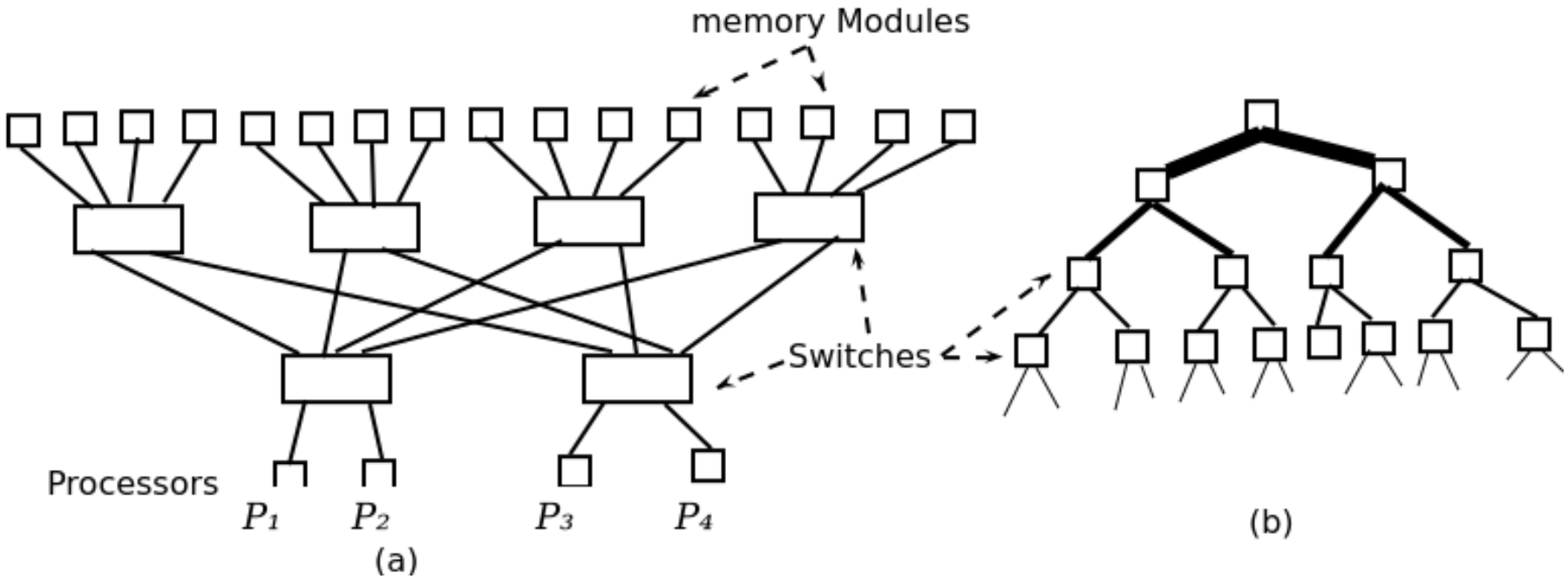
Mesh

Switches labeled (x,y); interior switches connect to 4 neighbors. Good fit for local-memory machines.

Hypercube

2^n switches, each with a distinct n-bit label; two switches connect iff their labels differ in exactly one bit.

Multistage and Fat-Tree Networks



Connects input switches (stage 1) to output switches (stage L) through L stages. Common in modular-memory computers, where processors attach to input switches and memory modules to output switches; a request travels stage-by-stage to its module and the response returns the same way.

Binary Trees, and the Fat-Tree Fix

A complete binary tree of height k has $n = 2^{k+1}-1$ nodes: the root at level 0, and 2^k leaves at level k. Each non-root, non-leaf node has one parent and two children.

3

degree of the network at internal nodes

$2\lceil \log_2 n \rceil$

communication diameter (2k)

1

bisection bandwidth of a plain binary tree — the bottleneck

The problem: when extensive communication occurs, all cross-tree messages must pass through the root — a severe bottleneck. Fat trees fix this by widening the communication links near the root, e.g. using parallel links in place of single ones.

References

[1] **Thomas H. Cormen, et al.,**
"Introduction to Algorithms, 3rd ed." — PHI, 2009

[2] **Mikhail J. Atallah & Marina Blanton,**
"Algorithms and Theory of Computation Handbook, Special Topics and Techniques, 2nd ed." — CRC Press, 2010

[3] **Allen B. Tucker, Jr.,**
"The Computer Science and Engineering Handbook" — CRC Press, 1997

Disclaimer: these notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the faculty.