Parallel Computation Models

- PRAM (parallel RAM)
- Fixed Interconnection Network
  - bus, ring, mesh, hypercube, shuffle-exchange
- Boolean Circuits
- Combinatorial Circuits
- BSP
- LOGP

Parallel and Distributed Computation

- Many interconnected processors working concurrently
- Connection Machine
- Internet

Types of Multiprocessing Frameworks

Parallel

- Technical Aspects
  - Parallel computers (usually) work in tight synchrony, share memory to a large extent and have a very fast and reliable communication mechanism between them.
  - Fixed Interconnection Network
    - bus, ring, mesh, hypercube, shuffle-exchange
  - Boolean Circuits
  - Combinatorial Circuits
  - BSP
  - LOGP

Distributed

- Connection Machine
- Internet
  
  Connects all the computers of the world

Purpose

- Parallel computers cooperate to solve more efficiently (possibly) difficult problems
- Distributed computers have individual goals and private activities. Sometimes communications with others are needed. (E.g., distributed data base operations).

Parallel Computers: Cooperation in a positive sense

Distributed Computers: Cooperation in a negative sense, only when it is necessary

Parallel Algorithms

- Which model of computation is the better to use?
- How much time we expect to save using a parallel algorithm?
- How to construct efficient algorithms?
- Are there problems not admitting an efficient parallel solution, that is inherently sequential problems?

Many concepts of the complexity theory must be revisited
We need a model of computation
- NETWORK (VLSI) MODEL
  - The processors are connected by a network of bounded degree.
  - No shared memory is available.
  - Several interconnection topologies.
  - Synchronous way of operating.

MESH CONNECTED ARRAY
degree = 4 (N) diameter = 2N

Other important topologies
- binary trees
- mesh of trees

In the network model a PARALLEL MACHINE is a very complex ensemble of small interconnected units, performing elementary operations.
- Each processor has its own memory.
- Processors work synchronously.

LIMITS OF THE MODEL
- different topologies require different algorithms to solve the same problem
- it is difficult to describe and analyse algorithms (the migration of data have to be described)
- A shared-memory model is more suitable by an algorithmic point of view

Model Equivalence
- given two models $M_1$ and $M_2$, and a problem $\Pi$ of size $n$
- if $M_1$ and $M_2$ are equivalent then solving $\Pi$ requires:
  - $T(n)$ time and $P(n)$ processors on $M_1$
  - $T(n^{1/\log n})$ time and $P(n)^{O(1)}$ processors on $M_2$

PRAM
- Parallel Random Access Machine
- Shared-memory multiprocessor
- unlimited number of processors, each
  - has unlimited local memory
  - knows its ID
  - able to access the shared memory
- unlimited shared memory

PRAM n RAM processors connected to a common memory of m cells
ASSUMPTION: at each time unit each $P_i$ can read a memory cell, make an internal computation and write another memory cell.
CONSEQUENCE: any pair of processors $P_i, P_j$ can communicate in constant time!
- $P_i$ writes the message in cell c at time t
- $P_j$ reads the message in cell c at time t + 1
PRAM

• Inputs/Outputs are placed in the shared memory (designated address)
• Memory cell stores an arbitrarily large integer
• Each instruction takes unit time
• Instructions are synchronized across the processors

PRAM Instruction Set

• accumulator architecture
  – memory cell $R_0$ accumulates results
• multiply/divide instructions take only constant operands
  – prevents generating exponentially large numbers in polynomial time

PRAM Complexity Measures

• for each individual processor
  – time: number of instructions executed
  – space: number of memory cells accessed
• PRAM machine
  – time: time taken by the longest running processor
  – hardware: maximum number of active processors

Two Technical Issues for PRAM

• How processors are activated
• How shared memory is accessed

Processor Activation

• $P_0$ places the number of processors ($p$) in the designated shared-memory cell
  – each active $P_i$ where $i < p$, starts executing
  – $O(1)$ time to activate
  – all processors halt when $P_0$ halts
• Active processors explicitly activate additional processors via FORK instructions
  – tree-like activation
  – $O(\log p)$ time to activate

THE PRAM IS A THEORETICAL (UNFEASIBLE) MODEL

• The interconnection network between processors and memory would require a very large amount of area.
• The message-routing on the interconnection network would require time proportional to network size (i.e., the assumption of a constant access time to the memory is not realistic).

WHY THE PRAM IS A REFERENCE MODEL?

• Algorithm’s designers can forget the communication problems and focus their attention on the parallel computation only.
• There exist algorithms simulating any PRAM algorithm on bounded degree networks.
  E. G. A PRAM algorithm requiring time $T(n)$, can be simulated in a mesh of $n \times n$ in $T(\log_2 n \log \log n)$, that is each step can be simulated with a slowdown of $\log_2 n \log \log n$.
• Instead of designing ad hoc algorithms for bounded degree networks, design more general algorithms for the PRAM model and simulate them on a feasible network.
Slide 19

- For the PRAM model there exists a well developed body of techniques and methods to handle different classes of computational problems.
- The discussion on parallel model of computation is still HOT.
- The actual trend: COARSE-GRAINED MODELS
  - The degree of parallelism allowed is independent from the number of processors.
  - The computation is divided in supersteps, each one includes:
    - local computation
    - communication phase
    - synchronization phase

Slide 20

Metrics

A measure of relative performance between a multiprocessor system and a single processor system is the speed-up $S(p)$, defined as follows:

$$ S(p) = \frac{T_1}{T_p} $$

Efficiency = $\frac{S}{p}$

Cost = $p \times T_p$

Slide 21

Metrics

- Parallel algorithm is cost-optimal:
  - parallel cost = sequential time
  - $C_p = T_1$
  - $E_p = 100%$

- Critical when down-scaling:
  - parallel implementation may become slower than sequential
  - $T_1 = n^3$
  - $T_p = n^{2.5}$ when $p = n^2$
  - $C_p = n^{4.5}$

Slide 22

Amdahl’s Law

- $f =$ fraction of the problem that’s inherently sequential
  - $(1 - f) =$ fraction that’s parallel
- Parallel time $T_p$: $T_p = f + (1 - f)/p$
- Speedup with $p$ processors: $S = \frac{1}{f + \frac{1-f}{p}}$

Slide 23

Amdahl’s Law

- Upper bound on speedup ($p = \infty$)
  - $S = \lim_{p \to \infty} S = \frac{1}{f}$

- Example:
  - $f = 2\%$
  - $S = 1 / 0.02 = 50$

Slide 24

PRAM

- Too many interconnections gives problems with synchronization
- **However it is the best conceptual model** for designing efficient parallel algorithms
  - due to simplicity and possibility of simulating efficiently PRAM algorithms on more realistic parallel architectures

```
[parallel for] n in X do in parallel
   instruction(n)
```

Shared-Memory Access

Concurrent (C) means, many processors can do the operation simultaneously in the same memory
Exclusive (E) not concurrent

- EREW (Exclusive Read Exclusive Write)
- CREW (Concurrent Read Exclusive Write)
  - Many processors can read simultaneously the same location, but only one can attempt to write to a given location
- ERCW
- CRCW
  - Many processors can write/read at/from the same memory location

Example CRCW-PRAM

- Initially
  - table A contains values 0 and 1
  - output contains value 0

\[
\text{for each } 1 \leq i \leq 5 \text{ do in parallel if } A[i] = 1 \text{ then output}=1;
\]


Example CREW-PRAM

- Assume initially table A contains [0,0,0,0,0,1] and we have the parallel program

\[
\text{for each } 1 \leq i \leq 5 \text{ do in parallel }
\]
\[
\]

then the consecutive values of the tables A (in parallel step 0, 1, 2, 3, 4, 5) correspond to the Pascal triangle, the numbers in the n-th row are:

\[
\binom{n}{0}, \binom{n}{1}, \binom{n}{2}, \ldots, \binom{n}{n}
\]

for n = 0, 1, 2, 3, 4, 5, 6.

Pascal triangle

\[
\begin{align*}
\binom{0}{0} & = 1 \\
\binom{1}{0} & = 1 \\
\binom{2}{0} & = 1 \\
\binom{2}{1} & = 1 \\
\binom{3}{0} & = 1 \\
\binom{3}{1} & = 1 \\
\binom{3}{2} & = 1 \\
\binom{4}{0} & = 1 \\
\binom{4}{1} & = 1 \\
\binom{4}{2} & = 1 \\
\binom{4}{3} & = 1 \\
\binom{5}{0} & = 1 \\
\binom{5}{1} & = 1 \\
\binom{5}{2} & = 1 \\
\binom{5}{3} & = 1 \\
\binom{5}{4} & = 1 \\
\end{align*}
\]

PRAM CREW

\[
\text{for each } 1 \leq i \leq 5 \text{ do in parallel for } n = 1, 2, 3, 4, 5, 6
\]
\[
\]