15-750: Graduate Algorithms April 7, 2017

Lecture 29: Parallel Algorithms

Lecturer: Gary Miller *Scribe: Nimrah Shakeel*

1 Motivation

So far we have assumed a RAM(Random Access machine) model in our study of algorithms, with unit time operations $(+, *, /)$ and read/write into memory. It is a central model for describing graph algorithms. Other models include [1]:

1. Agents or ants
2. Pointer machines

For large amounts of data (as n goes to infinity), the RAM model is unrealistic [1]:

1. Speed of light (Large machines)
2. Quantum effects (Small machines)

Eventually RAM is an important model because

1. Many important algorithms were discovered using this model
2. Most algorithms are coded in a RAM like language e.g C

2 Parallel Models

We are going to consider fixed connection machines, where machine means 1. Finite State Machine and 2.RAM
We will consider cellular arrays (1D, 2D and 3D). Examples include von Neumann (1940's) , algorithm for CA (60's and 70's) , Alvy Ray Smith (1974), Wolfram, Klaus Sutner, Conway game of Life (80's), Edgar Codd(60's), HT Kung (80's).
Highly Connected Models:

1. Hypercube $= (V, E)$ (1980's)
   $V = (a_1, ..., a_m) | a_i \in 0, 1, m = \log n$
   $((a_1, ..., a_m), (a_1, ..., \bar{a_i}, ..., a_m)) \in E$
2. Shuffle-exchange graph (1980's)
   $V = (a_1, ..., a_m) | a_i \in 0, 1$ $((a_1, ..., a_m), (\bar{a_1}, a_2, ..., a_m)) \in E$ $((a_1...a_m), (a_m a_1...a_{m-1})) \in E$
3. Randomly connected graphs
   Possible models of the brain (Valiant)

3 PRAM Issues

What should be the effect of a CW on an EW machine?

(a) Machine crashes!
(b) Garbage Reads!

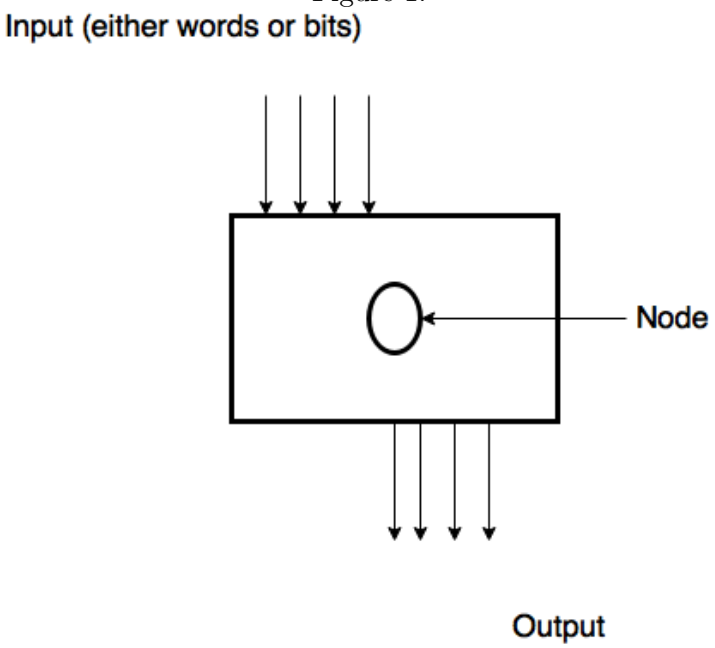
How is synchronization handled?

(a) Synchronize after each unit of time!
(b) Bulk Synchronous Parallel (BSP) Valiant
(c) Not handled

We will mostly use 1.

4. Circuit Model

Figure 1:



Nodes:

(a) $OR, AND, XORgates$
(b) Arithmetic operations

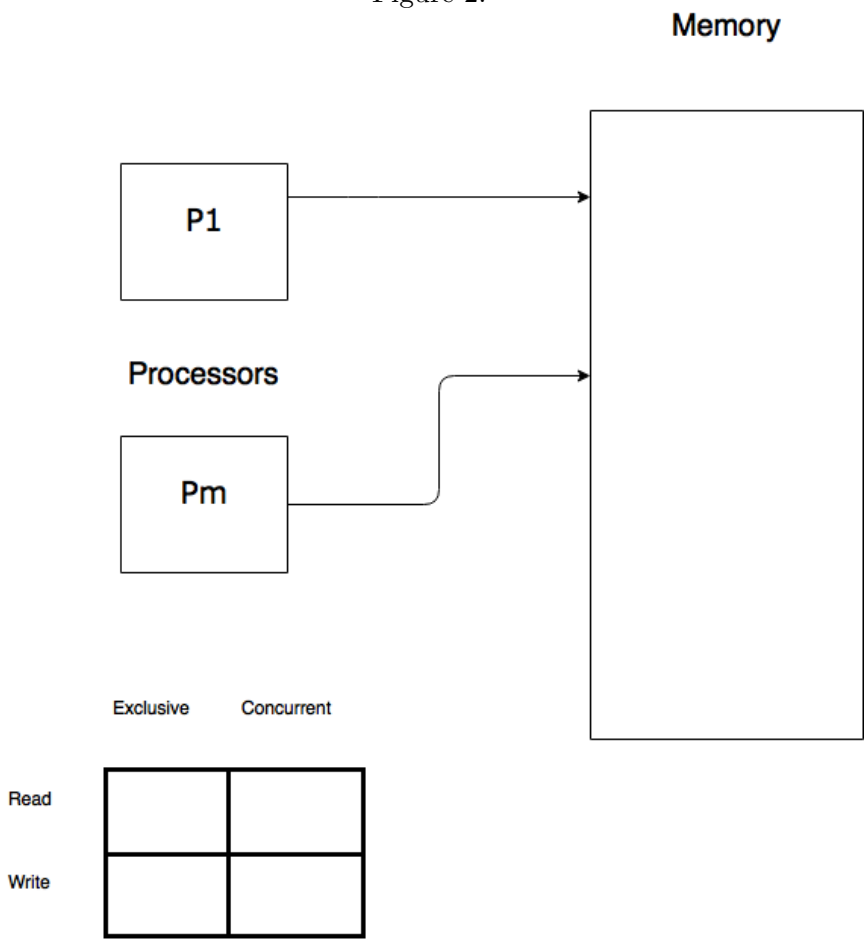
(a) Constant fan in
(b) Arbitrary fan out

Size = no. of nodes = Work
Time = Longest path from input to output (Critical path)

5. Neural Nets (employed in deep learning)

4 Shared memory Models

Figure 2:

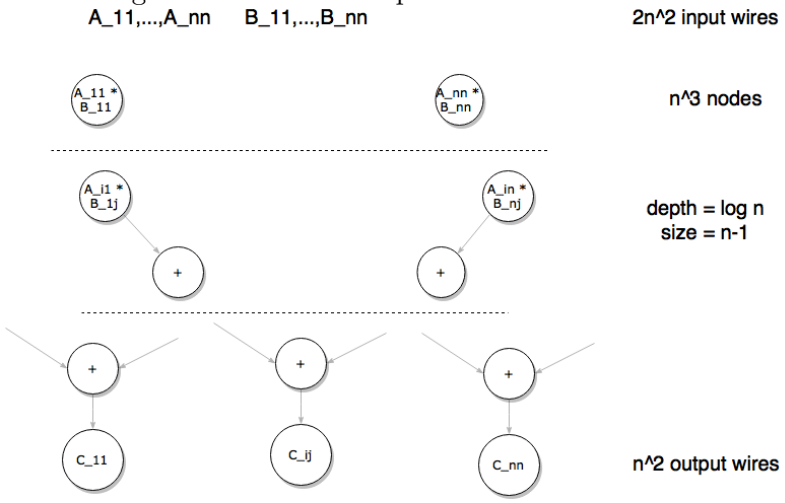


1. PRAM(Parallel Random Access machine)
   Unit time operations $(+, *, /,$ read/write)
   Operations might be ER,EW, CR,CW

5 Naive Matrix Multiply in the circuit model

$C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}, C = A.B$

Figure 3: Matrix multiplication in the circuit model



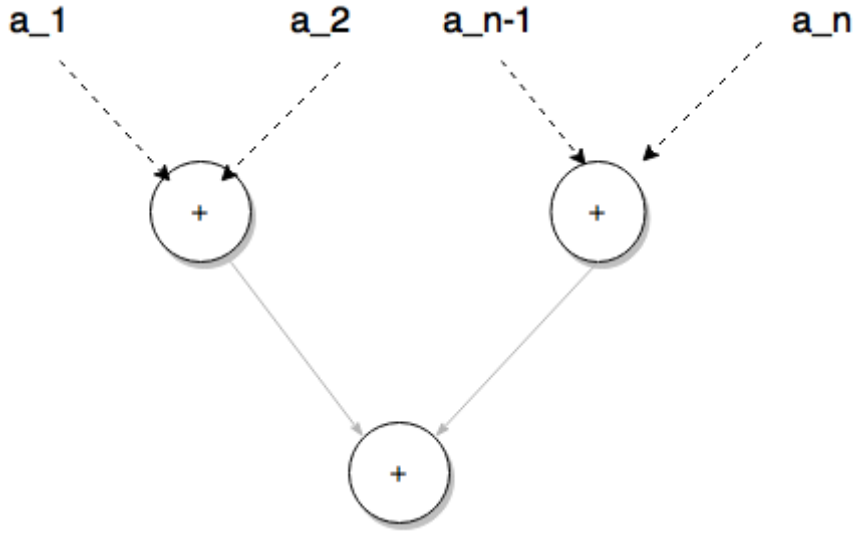
Circuit Totals:
Work : O(n^3)
Time: $\omega(\log n)$
Naive Matrix Multiply on PRAM
P = no. of processors, T = Parallel Time

1. 1-processor per node, CRCW, P = $\omega n^3, T = \omega(\log n)$
2. Fan-out = reads, Fan-in = writes
3. Easy CREW (each processor reads its arguments)
4. EREW (Use binary trees to make copies, this increases depth by $\log n$)

6 PRAM Work

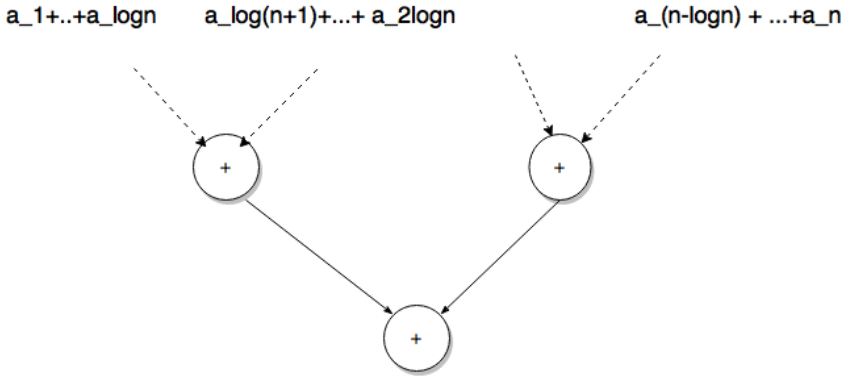
Definition : P = no. of processors used for life of run
T = total time
Example: Matrix Multiplication
So far P = $\omega(n^3)$ and T = $\omega(\log n)$
$P.T = \omega(n^3 \log n)$
Claim $\omega(n^3 \log n)$ processors, $\omega(\log n)$ time for naive matrix multiplication.
Proof Let us start with n^3 multiplications
Each P_i computes $\log n$ multiplications in $\omega(\log n)$ time.
$P = \omega(\frac{n^3}{\log n})$, $T = \omega(\log n)$
Additions: For example:

Figure 4: Example addition



Replace with

Figure 5: Example addition



Add n-numbers in $\omega(fracn \log n)$ processors in $\omega(\log n)$ time.

7 The Slow Down Principle:

Give parallel algorithm processor P and time T
$\forall P' \leq P$, run algorithm in time $\frac{P}{P'}T$ using P' processors.
Each processor simulates $\frac{P}{P'}$ virtual processors
Example: Strassen's Algorithm
The recurrence is:
$MM(n) = 7MM(\frac{n}{2}) + cn^2$
Note: Matrix addition is $\omega(n^2)$ work in $\omega(1)$ time
The recurrence for time is:
$T(n) = T(\frac{n}{2}) + \omega(1)$, which amounts to $\omega(\log n)$ time.
Processors: $P(n) = 7P(\frac{n}{2}) + cn^2$
We must pay for each call, hence this amounts to $\omega(n^{2.81})$ processors.
Total work is $\omega(n^{\log_2 7}) \log n$