Welcome: 15-859HH Parallel + Concurrent algorithms

i.e. What do I do with all those cores?

Guy Blelloch

Goal: Understand 40+ years of work on Parallel + Concurrent algorithms + look at recent cool topics

Mainly theoretical. But will be opportunity for implementations.

They work in practice.

Sorting: C++ quicksort, 101 seconds
1 Billion keys, double precision.
~~doubles~~
72 Core Machine, 1.15 seconds, 87x

BFS, reachability,
400 Million edge social network
9 seconds sequentially) 42 speedup
.215 seconds
.088 seconds → 103 sec code

Grading: Assignments (3-4)
Final Project
A 30 minute presentation
Participation

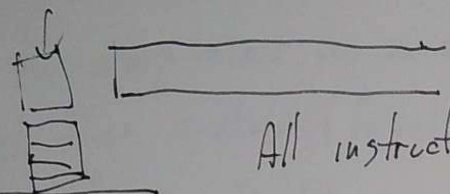
First part, Parallel Algorithms

Brief History of Parallel Algos.

1968 - Batcher Sort, Bitonic Sort
1975 - Valiant Merging, Max
→ loglogn rounds
1978 - PRAM Wyllie
1979 - Circuit Models (P-complete, NC)
1980 - Many ideas covered in first couple week were derived PRAM
1990-95 - Other Models
Asynch PRAM
BSP
Word-depth, Work-span
1995-2010: Dormant (Parallel Comp Winter, AI Winter)
2010 - Present: A lot of interesting work

Models:

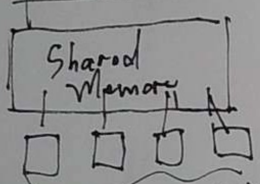
RAM - Random Access Machine



All instructions take unit time

Parallel Models

PRAM - Parallel Random Access Machine



Processors (P)

- synchronous (SIMD)
- Constant time per instruction

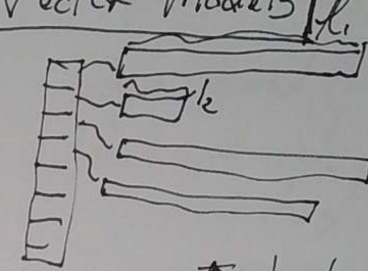
Costs: # of time steps (Time)
Processors

Sorting $O\left(\frac{n \lg n}{P} + \log^2 n\right)$: time

ER - exclusive read
EW - exclusive write
CR - concurrent read
CW - concurrent write (*Arbitrary* Write arthmin)

CREW model

Vector Models



Memory

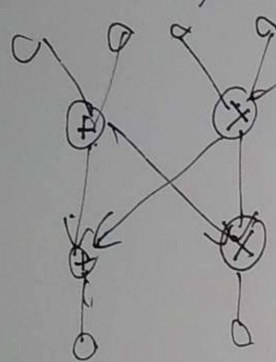
Instructions:
e.g. add two vector

Cost : Step: # of instruction
Work: $\sum_{steps} l_i$ ← length of vectors on step i

Sorting ; Step $O(\log^2 n)$
Work $O(n \log n)$

Circuit Models | 1979

Model computation as a circuit with and + or gates (Acyclic)



Cost Measures

Size: # gates

Depth: longest path

AKS

P-complete, NC

NC^k : Circuits with $O(\log^k n)$ depth

Polynomial size

P-complete problems

Polynomial

Multiprocess Models (MP-RAM)

Single Shared Memory

Start with single process

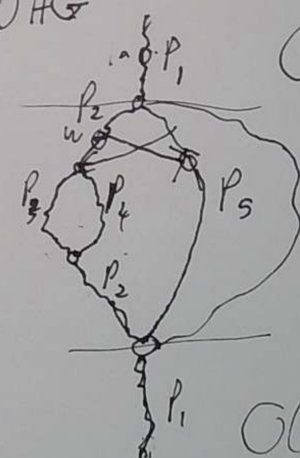
- registers
- program counter

Fork(n): Forks n processes ← child

- same registers, except for an id register
- same prog counter

- When all child processes finish parent continues

DAG



Costs

Work: total # of instructions

Span(Depth): longest path of instructions

$O(n \log n)$ work

$O(\log^2 n)$ span

Races: Assume linearized order