

Parallelism and Cloud Computing

Kai Shen

1

Parallel Computing

- Parallel computing: Process sub-tasks simultaneously so that work can be completed faster. For instances:
 - divide the work of matrix multiplication into subtasks to run in parallel
 - parallelize the modeling of star movements and galaxy formation
 - parallelize the web page ranking (at Google)
 - parallelize the mining of social networks (at Facebook)
 - parallelize the prediction of customer preferences (at Amazon, Netflix, VISA/Mastercard)
 - parallelize the graphics rendering in making animated movies (at Pixar/Disney)
 -

2

Why Parallel Computing?

The speed of uniprocessor will catch up with the need of computing?

- Moore's law (1965, with later revisions) – circuit complexity doubles every one year (18 months, 2 years)
- Some argue that it is a self-fulfilling prophecy
- Moore said in 2005:
 - “In terms of size [of transistors] you can see that we're approaching the size of atoms which is a fundamental barrier, ...”

3

Intel x86 Evolution: Milestones

<i>Name</i>	<i>Date</i>	<i>Transistors</i>	<i>MHz</i>
■ 8086	1978	29K	5-10
■ 386	1985	275K	16-33
■ Pentium 4F	2004	125M	2800-3800
■ Core i7	2008	731M	2667-3333
■ Haswell	today	1.4B	Not faster

- First 16-bit processor. Basis for IBM PC & DOS
- First 32 bit processor , referred to as IA32
- First 64-bit processor, referred to as x86-64
- Multicore
- Improve on core count, power reduction, not frequency increase

4

Why Parallel Computing?

Not always desirable to use the fastest hardware.

Example:

- A machine with eight 3.6GHz CPUs consumes up to 100 Watts
 - A machine with four 1.9GHz CPUs consumes no more than 4 Watts
- It may be more energy-efficient to use multiple low-power machines in parallel

5

Why Parallel Computing?

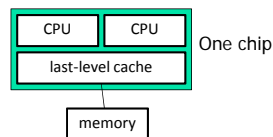
Multiprocessors are ubiquitous

- Commodity processors contain multiple computing cores because they are cost-effective to produce, and power-efficient to run
- Traditionally sequential software (e.g., desktop applications) should take advantage of it

6

Multicore Architecture

- Last-level cache becomes a significant part of the chip
- ⇒ Multicore: add another processor core on the chip (sharing a single last-level cache)
- Low cost (manufacturing, power)



- Additional benefit: faster processor-to-processor sharing

7

Arguments against Parallel Computing?

- CPU performance isn't important
- Parallel programming is too difficult

8

Parallelism and Dependency

- Computation-intensive application contains tasks that can run in parallel
- Dependencies limit available parallelism/concurrency
 - Dependency between two operations – one has to wait for another to complete
 - Examples of dependencies?
- Amdahl's law:

$$\text{Normalized runtime} = 1 - \text{fraction_enhanced} + \frac{\text{fraction_enhanced}}{\text{speedup_enhanced}}$$

9

Load Balance

- Utilizations of parallel processors
 - What happens if most work is assigned to one processor?
- Require load balancing
 - Parallel processors are typically symmetric

10

Simulation of Ocean Currents

- Entity in ocean floor: a unit body of water
- Variables: velocity, direction, ...
- Simulation over time
 - Velocity/direction of water body depends on the velocity/direction of nearby water bodies in the last time unit
- Parallelism/concurrency
- Dependencies
- Load balance

11

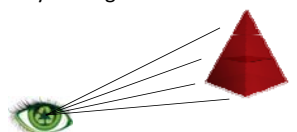
Evolution of Galaxies

- Entity in space: a star
- Variables: velocity, direction, location, ...
- Simulation over time
 - Velocity/direction/location of a star depends on the mass and location of other stars in the last time unit
- N-body simulation
- Parallelism/concurrency
- Dependencies
- Load balance

12

Ray Tracing

- Ray tracing



- Parallelism/concurrency
- Dependencies
- Load balance

13

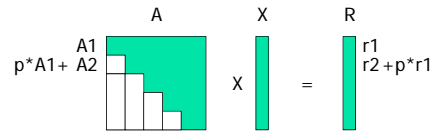
PageRank

- PageRank scores each page
 - the score is the converged probability for a random web surfer to arrive at the page
- Scores transfer through links (random click by the surfer):
 - If pages B, C are the pages that link to A, then $PR(A) = PR(B)/L(B) + PR(C)/L(C)$
- Damping factor (for stability):
 - The surfer has a tendency to stay at where he/she is
 - $PR(A) = (1-d)/N + d(PR(B)/L(B) + PR(C)/L(C))$
- A linear system of equations (N variables, N linear equations)

4/23/2015 CSC 258/458 - Spring 2015 14

Parallel Programming Example: Gaussian Elimination

- Solving a system of linear equations
- Reduce an equation matrix into an equivalent upper-diagonal matrix



- Parallelism/concurrency
- Dependencies
- Load balance

15

More on PageRank

- $PR(A) = (1-d)/N + d(PR(B)/L(B) + PR(C)/L(C))$
- Iterative solver:
 - Start from an initial PR vector (1/N for each page)
 - Compute a new PR vector by the above linear transformation
 - Keep repeating the linear transformation until the PR converges (very small change after further linear transformation)
 - Even with a large, complex web graph, often converge after dozens of iterations
- Each iteration is a matrix-vector multiplication, so the whole computation is a series of matrix-vector multiplication

4/23/2015 CSC 258/458 - Spring 2015 16

Parallel Programming

- Challenges:
 - Control dependencies
 - Data sharing
- Parallel algorithms:
 - Identification of parallelisms in applications
 - Design control flow and data sharing mechanism
- Parallel programming:
 - Shared memory
 - Distributed memory

17

Shared Memory Parallel Systems

- In shared memory parallel computers, multiple processors can access the memory simultaneously
- Problems:
 - One processor's cache may contain a copy of the data that was just modified by another processor
⇒ hardware support for cache coherence
 - Two processes' (semantically) atomic operations may be interleaved
⇒ system support for mutual exclusion and synchronization

18

Distributed Memory Parallel Systems

- Parallel systems that do not share memory
- Less requirement on the system support
 - Little or no hardware support
 - Software system support for communications (point-to-point, group)
- More trouble with writing applications
 - Data must be explicitly partitioned and transferred when needed
 - Hard to do dynamic workload management

19

What is Cloud Computing?

"The interesting thing about Cloud Computing is that we've redefined Cloud Computing to include everything that we already do. . . . I don't understand what we would do differently in the light of Cloud Computing other than change the wording of some of our ads."

- Larry Ellison, quoted in the Wall Street Journal, September 26, 2008

"A lot of people are jumping on the [cloud] bandwagon, but I have not heard two people say the same thing about it."

- Andy Isherwood, quoted in ZDnet News, December 11, 2008

20



Resource Provisioning Problem

- Provisioning of computing/storage resources for Internet applications
- Challenges:
 - Difficulty to predict load of Internet applications
 - Curse of hugely successful Internet services, DoS attacks
 - Cost of dynamic capacity expansion
 - High upfront cost
 - I want to try some idea and see if people are interested
 - Cost of excess capacity (power)

21



Cloud Computing: Computing As A Utility

- Pooling resources together, managed centrally (in a data center); many computing jobs and network services share the resource pool
- Benefits:
 - While one application's resource needs may vary dramatically, the sum of many independent applications' resource needs varies much less
 - More efficient resource utilization through sharing
 - Space sharing as well as time sharing
 - Upfront setup cost is amortized over many applications
 - Cost is proportional to use
 - Commoditizing support for availability, data durability, ...

22