Introduction to Parallel Computing

Pangfeng Liu
National Taiwan University

May 4, 2020

Why Parallel Computing?

- Solve a problem faster.
- Solve a problem better.

Olympic Games

The modern Olympic Games are the leading international sporting event featuring summer and winter sports competitions in which thousands of athletes from around the world participate in a variety of competitions.[1]

[1] http://en.wikipedia.org/wiki/Olympic_Games

Olympic Motto

The Olympic motto, Citius, Altius, Fortius, a Latin expression meaning "Faster, Higher, Stronger" was proposed by Pierre de Coubertin in 1894 and has been official since 1924.

Faster

- The reigning 100 m Olympic champion is often named "the fastest man/woman in the world".
- The world record is 9.58 seconds.

Higher

- The high jump is a track and field event in which competitors must jump over a horizontal bar placed at measured heights without the aid of certain devices.
- The world record is 2.45m.

Stronger

- Olympic weightlifting, also called Olympic-style weightlifting, or weightlifting, is an athletic discipline in the modern Olympic programme in which the athlete attempts a maximum-weight single lift of a barbell loaded with weight plates.
- The world record is 305 kg, the sum of snatch and clean & jerk.

Discussion

- Why faster, higher, and stronger?

Why Faster?

- Many computations are *slow*.
- Many computations are *time critical*.

Slow Computation

- Brute force search
- Computer simulation
- Exponential time complexity
- Grand Challenge problems

Traveling Salesman

Given a set of cities and the distance between two cities, find the shortest route that goes through all cities without visiting any city twice.

- A famous NP-complete problem, i.e., a proven hard-as-hell problem in computer science.
- Easy, you just permute all cities and find the shortest route.
- However, the number of permutations is $(n-1)!$ where n is the number of cities.

Factorial

- Consider a traveling salesman problem with 101 cities.
- 100! is roughly $9.332621544 \times 10^{157}$.
- Assume that the computation of the length of a path can be done in 100 cycles.
- Assume that a computer runs at 10 GHz.
- It takes $9.332621544 \times 10^{146}$ seconds to enumerate all paths.

Computation Time

- A year has 31,556,926 seconds, so the computation takes 2.9574×10^{130} billion years.
- The age of earth is 4.54 billion years[2].
- The age of the universe is 13.798 billion years[3].
- Do you honestly think anyone can live to see the results?

[2] http://en.wikipedia.org/wiki/Age_of_the_Earth

[3] http://en.wikipedia.org/wiki/Age_of_the_universe

Algorithm Optimization

Despite techniques like branch-and-bound and A^* search can reduce the number of cases one needs to examine, the sheer number of permutations is enormous.

Discussion

- Give an example of computation that will take a lot of time.

Time Critical Computation

- Weather forecast
- Stock market
- Radar signal processing

Worried? Me?

- We do not need to do anything because computers become faster *automatically*!
- "Moore's law" is the observation that, over the history of computing hardware, the number of transistors in a dense integrated circuit doubles approximately every two years[4].
- If the speed of a CPU is proportional to the number of transistors, we expect an eight times speed improvement in six years.

[4] http://en.wikipedia.org/wiki/Moore%27s_law

Worried? Yes.

- We assume we have an $O(n^3)$ "efficient" algorithm to solve a problem, e.g., matrix multiplication.
- You can solve problem size twice as large as the original one if you wait six years.
- By the time you wait for the hardware to catch up, your career (e.g., as a Ph.D. student) is over.

How to be Faster?

Hardware improvement cannot help you, OK!! You need better solutions than waiting.

Late Information

- Late is *not* better than nothing.
- Justice delayed is justice denied.
- A weather forecast for tomorrow takes three days.
- A stock recommendation for next week takes three weeks.
- The computation of an early warning radar system takes three times of the time for an enemy missile to destroy the radar.

How to be Faster?

Having more than one CPU to work on the problem seems to be a reasonable choice.

Discussion

- Give an example of time-critical computation.
- Describe Moore's Law.

Why Better?

- People are *greedy*.
- Video quality is an increasing function of time.
  - VCD (NTSC) 352×240.
  - DVD (25 frame rate) 720×576.
  - Blueray (HD) 1920×1080.
- The number of pixels increases 24.5 times.
- An $O(n^2)$ algorithm will have to run 600 times faster.

Why Better?

- Again, remember that people are *greedy*.
- Remember faster, higher, stronger.
- We always seek possibility just beyond our capability.
- We want to live long and prosper (show a Vulcan gesture).

Live Long and Prosper

5

[5] http://vignette4.wikia.nocookie.net/memoryalpha/images/5/52/Spock_performing_Vulcan_salute.jpg/revision/latest?cb=20090320072701&path-prefix=en

The Weather

- The weather forecast resolution is proportional to the number of cells in the simulation.
- Taiwan has a length of 394 km and a width of 144 km.
- The area to simulate is 56736 square km.
- Assume that the simulation model is 10 km in height.
- If the resolution is 1 cubic km, we need 567360 cells.
- If we refine the resolution to 100 m, the number of cells will be 567360000.
- An $O(n^2)$ algorithms will have to run 1000000 times faster.

Discussion

- A naive matrix multiplication algorithm multiply two matrices of size n by n in $O(n^3)$ time. If we increase the size of the matrix n by a factor of 10, the execution time of this naive algorithm will roughly be?

Scientific Process

1. Observation
   - Why does the apple fall on my head?
2. Theory
   - Every matter attracts every matter.
3. Experiment
   - Cavendish's experiment to verify the theory.

Newton and Apple

6

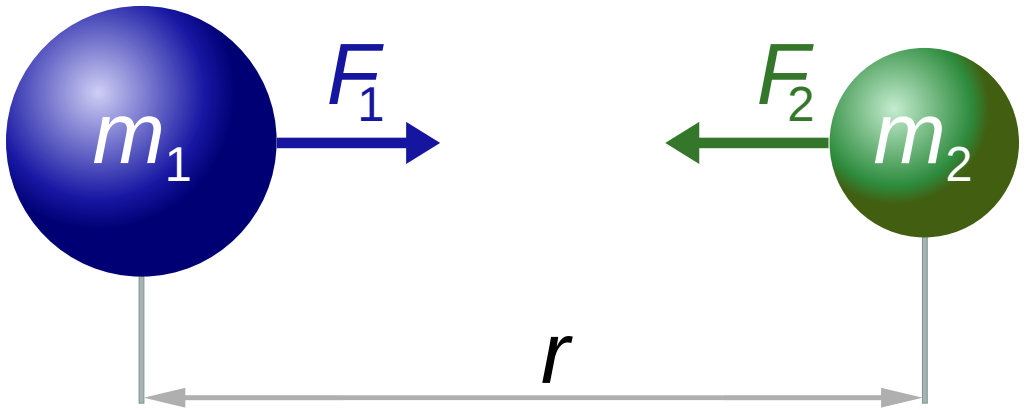
[6] http://www.yalcafruittrees.com.au/wp-content/uploads/Isaac-Newton.png

Newton's Law of Universal Gravitation

Newton's law of universal gravitation states that any two bodies in the universe attract each other with a force that is directly proportional to the product of their masses and inversely proportional to the square of the distance between them.[7]

[7] http://en.wikipedia.org/wiki/Newton%27s_law_of_universal_gravitation

Formulation

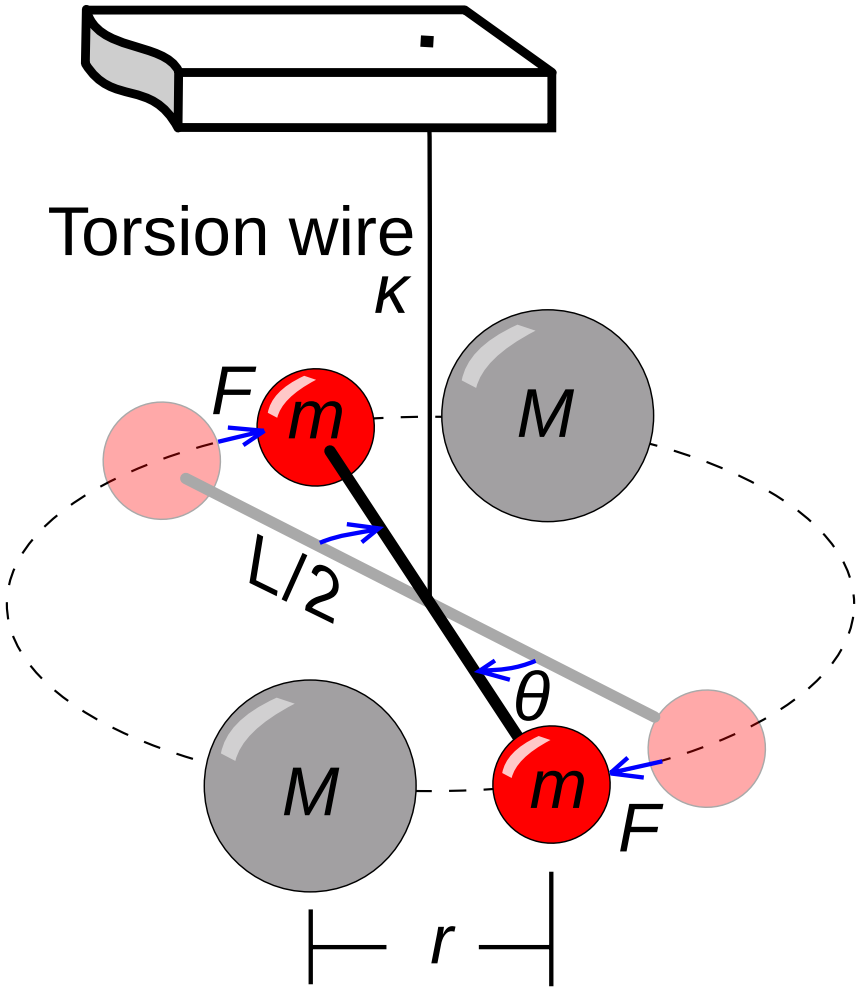


$$F_1 = F_2 = G\frac{m_1 \times m_2}{r^2}$$

8

[8] NewtonsLawOfUniversalGravitation by I, Dennis Nilsson. Licensed under CC BY 3.0 via Wikimedia Commons http://upload.wikimedia.org/wikipedia/commons/thumb/0/0e/NewtonsLawOfUniversalGravitation.svg/200px-NewtonsLawOfUniversalGravitation.svg.png

Cavendish's Experiment



[9]

[9] Cavendish Torsion Balance Diagram by Chris Burks (Chetvorno). Licensed under Public Domain via Wikimedia Commons
http://commons.wikimedia.org/wiki/File:Cavendish_Torsion_Balance_Diagram.svg#mediaviewer/File:

Scientific Process

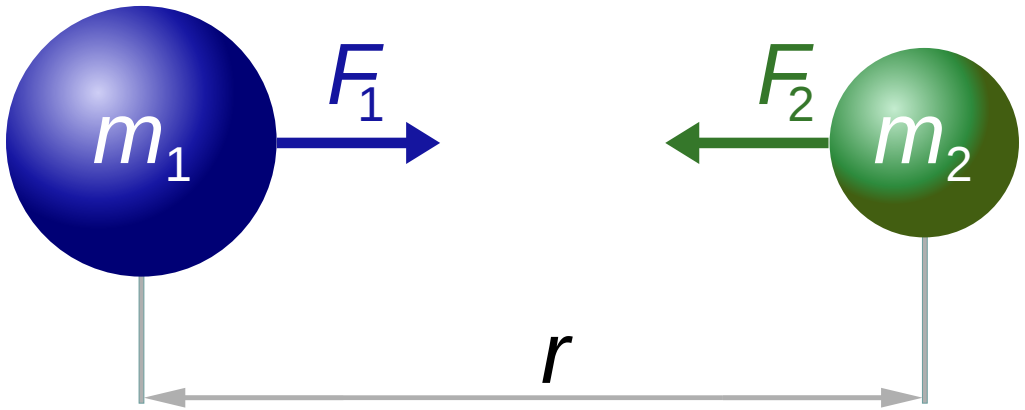
1. Observation
   - Why does the galaxy have spirals?
2. Theory
   - Every matter attracts every matter.
3. Experiment
   - What are you talking about?

Spiral Galaxy

10

[10] http://i.space.com/images/i/000/022/667/wS4/spiral-galaxy-ngc1232-1600.jpg

Formulation



$$F_1 = F_2 = G\frac{m_1 \times m_2}{r^2}$$

11

[11] NewtonsLawOfUniversalGravitation by I, Dennis Nilsson. Licensed under CC BY 3.0 via Wikimedia Commons http://upload.wikimedia.org/wikipedia/commons/thumb/0/0e/NewtonsLawOfUniversalGravitation.svg/200px-NewtonsLawOfUniversalGravitation.svg.png

Experiment

There are no ways we can put stars in a laboratory and observe the effects of gravity to incur spirals.

Discussion

- Give an example of scientific process, including observation, theory, and experiment.
- Give an example of experiment that cannot be conducted in a laboratory.

Difficulties

- Experiments are *expensive* – the stars may be expensive to buy.
- Experiments are *dangerous* – you do not want to have a black-hole in your laboratory.
- Experiments are *unfeasible* – my lab is not large enough.
- Experiments are *time consuming* – I do not have billions of years for observation.

Computer Simulation

- Simulation is cheap.
- Simulation is safe.
- Simulation is feasible.
- However, simulation is *slow*.

How to be Better?

Having more than one CPU to work on the problem seems to be a reasonable choice.

Discussion

- Give an example of scientific simulation.

Grand Challenge

A grand challenge is a fundamental problem in science or engineering, with broad applications, whose solution would be enabled by the application of high performance computing resources that could become available in the near future.[12]

[12] http://en.wikipedia.org/wiki/Grand_Challenge

Grand Challenge Examples

- Computational fluid dynamics
- Electronic structure calculations
- Plasma dynamics for fusion energy technology and for safe and efficient military technology
- Calculations to understand the fundamental nature of matter, including quantum chromodynamics and condensed matter theory
- Symbolic computations

How to be Better?

Having more than one CPU to work on the problem seems to be a reasonable choice.

Discussion

- Give an example of grand challenge problem.

Use multiple CPU's to solve a problem faster and/or better.

Why Parallel Computing?

- We want computation to be faster and better.
- The performance of a single computer is limited, so the only way is to have *more* computers.
- We cannot find single core CPU anymore[13], so it is essential to know how to get performance from a parallel computer.

[13] http://www.intel.com/pressroom/kits/quickreffam.htm

Limits

There are technological and physical limits to uni-processor performance that cannot be overcome. For example, clock times cannot be smaller than the response time of electronic circuits, which in turn are limited by physical laws.[14]

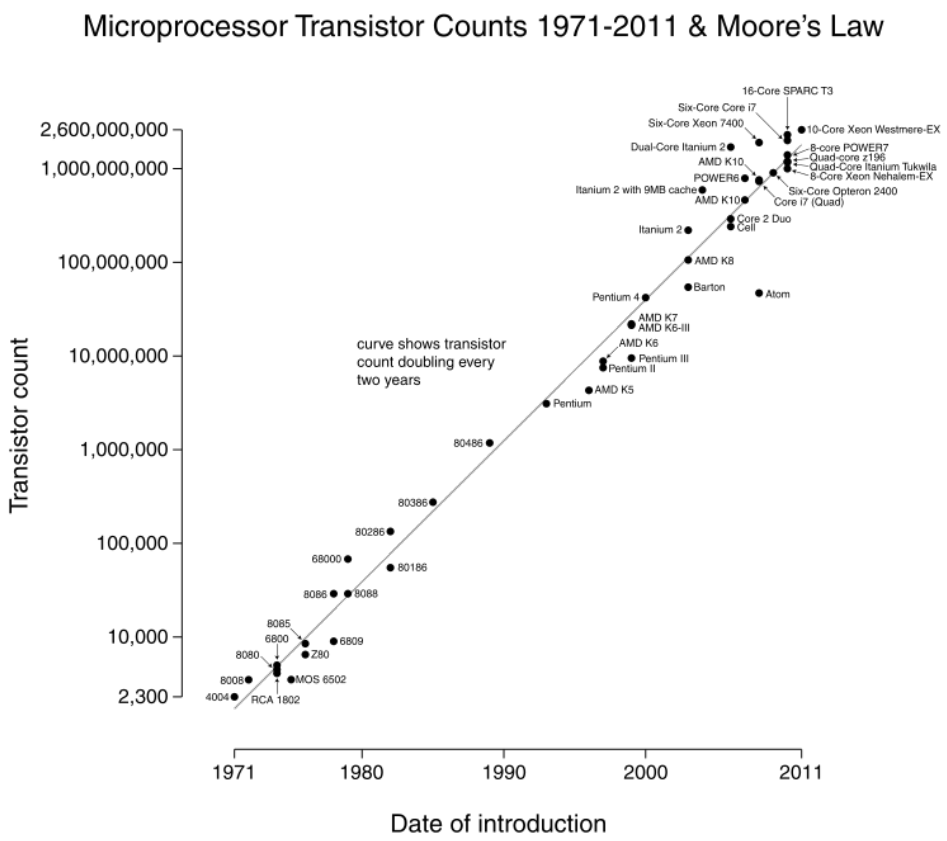
[14] Pangfeng Liu, The Parallel Implementation of N-body Algorithm, Ph.D. dissertation, 1994.

Moore's Law

"Moore's law" is the observation that, over the history of computing hardware, the number of transistors in a dense integrated circuit doubles approximately every two years.[15]

[15] http://en.wikipedia.org/wiki/Moore%27s_law

An Illustration



16

[16] Transistor Count and Moore's Law - 2011 by Wgsimon - Own work. Licensed under CC BY-SA 3.0 via Wikimedia Commons

Fifty Years of Moore's Law

- Moore's Law is a direct consequence of the incredible and unique scaling heuristics of semiconductor manufacturing: by holding the cost per unit area of manufacturing constant[17]
- The economic benefits of Moore's Law come from the shrinking of the transistor.

[17] Chris A. Mack, IEEE Transactions on Semiconductor Manufacturing, Vol. 24, NO. 2, MAY 2011.
http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5696765

- Moore's Law can be formulated as a learning curve by plotting minimum feature size as a function of cumulative revenue or area of silicon produced by the industry on a log-log scale.
- Moore's Law has kept on a relatively constant learning curve until about 1998—2000. The acceleration of this Moore's learning curve over the last decade is likely an unsustainable, momentum-driven attempt to recapture past revenue growth rates.
- The industry, and the world, has enjoyed 50 remarkable years of Moore's Law. *There are unlikely to be many more years left*.

Discussion

- Give at least three data points to support the Moore's Law.

Parallel Computer

- A parallel computer is a system that has multiple processing units and supports parallel computing by *parallel programming*.
  - Multicore
  - Multiprocessor
  - Multicomputer

Multicore CPU

- A CPU that has multiple cores as processing units.
- The cores share the memory, and have usually have their own cache.
- A memory arbitrator guarantees the consistency of shared memory and cache.

Intel Ivy Bridge Xeon Processors

- Ivy Bridge is the codename for a line of processors based on the 22 nm manufacturing process developed by Intel.
- Ivy Bridge Xeon has up to 15 cores and 37.5 MB L3 cache, released on February 2014.

Intel Xeon Phi

- Intel Many Integrated Core Architecture (MIC) is a coprocessor computer architecture developed by Intel incorporating the Larrabee many core architecture, the Teraflops Research Chip multicore chip research project, and the Intel Single-chip Cloud Computer multicore microprocessor[18].
- At the International Supercomputing Conference (2012, Hamburg), Intel announced the branding of the processor product family as Intel Xeon Phi.

[18] http://en.wikipedia.org/wiki/Xeon_Phi

Design

- The cores of Intel MIC are based on a modified version of P54C design, used in the original Pentium.
- The basis of the Intel MIC architecture is to leverage x86 legacy by creating a x86-compatible multiprocessor architecture that can utilize existing parallelization software tools.
- Having a large number of cores – for example, 51110P has 60 cores running at 1.053 GHz.[19]

[19] http://ark.intel.com/zh-tw/products/71992/Intel-Xeon-Phi-Coprocessor-5110P-8GB-1_053-GHz-60-core

Discussion

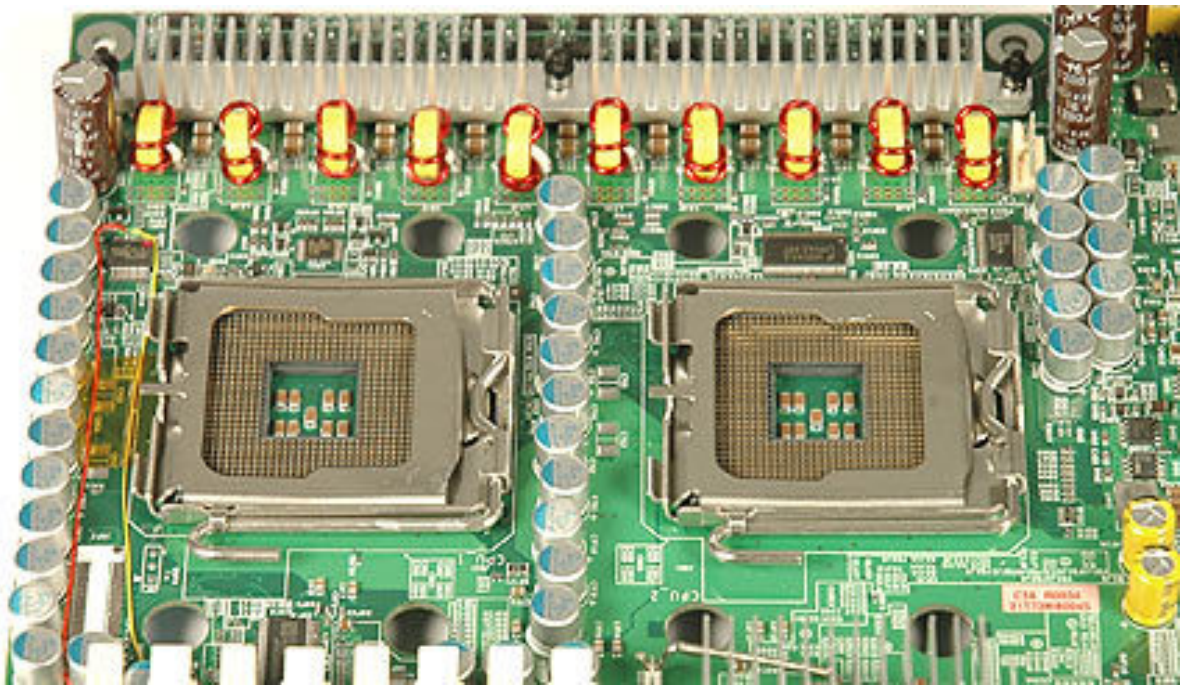
- Describe the number of cores in at least three current CPU's.

Multiprocessor

- A parallel system consists of *multiple* processors.
- Note that in this course we usually do not distinguish "processor" and "cores", so we do not make a clear distinction between "multiprocessor" and "multicore".
- The processors are connected by a *shared memory*.
- Since the processors are connected together by a shared memory, they communicate with each other by writing and reading the shared memory, like writing on bulletin board.

Dual Socket Server

- Intel Server Board S5000PSL with two Xeon sockets.



Tianhe Cluster Node

- Tianhe is a huge cluster consisting of 16,000 computers (more details later).
- Each computer (node) has two Intel Ivy Bridge Xeon processors and three Xeon Phi chips.
- A single node has several processors. The two Xeon and three Xeon Phi share a memory on the same printed circuit board.

Tianhe 2 Node

- The two Xeon CPU share a memory, and the three MIC (Xeon Phi) have their own memory.

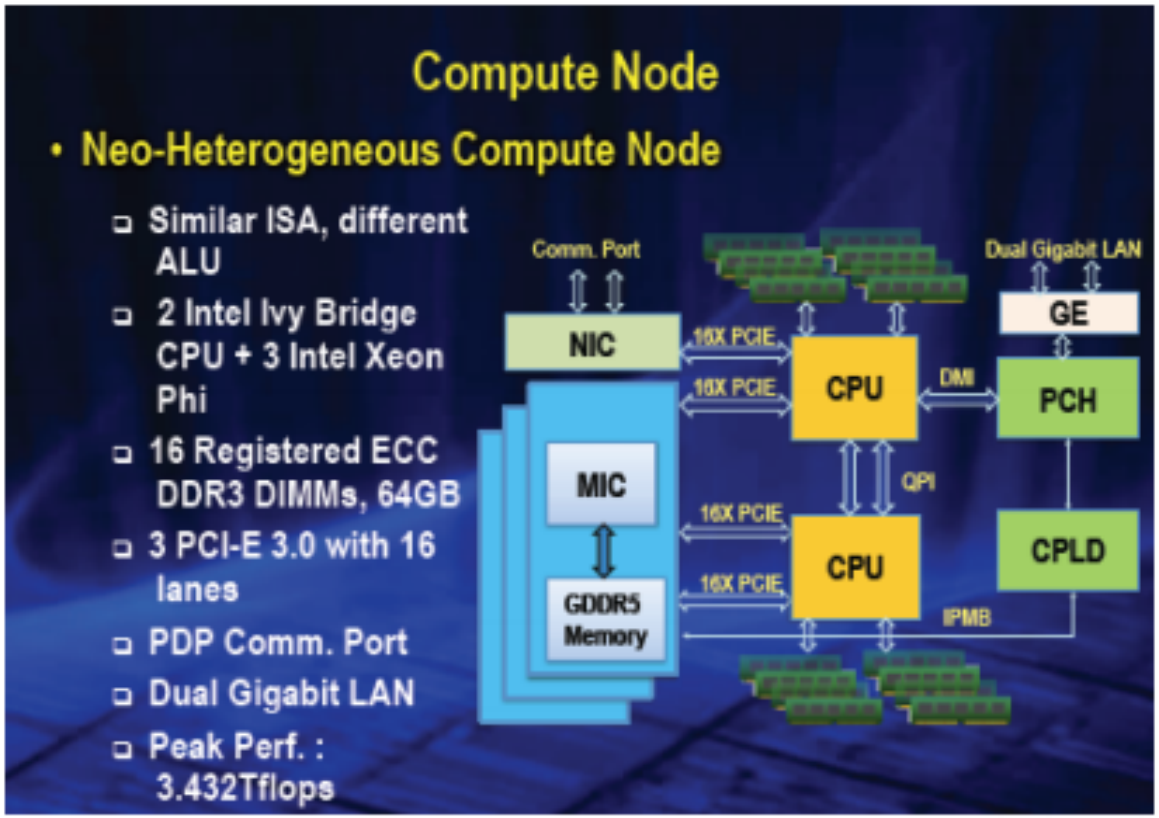


Figure 1: Compute Node

Discussion

- Describe the number of processors within a node from any system in the top 10 of the top 500 list.

Multicomputer

- A parallel system consists of *multiple* computers.
- The computers are connected by a *communication network*.
- Since the computers are independent and do not share memory, they communicate with each other by *messages*, like making phone calls.

Multicomputer Examples

- Cluster computing
- Massively parallel computing
- Grid computing
- Cloud computing

Cluster

- A computer cluster consists of a set of loosely or tightly connected computers that work together so that, in many respects, they can be viewed as a single system.[20]
- Usually the computers are *loosely connected*. i.e., they are *not* connected by fast and expensive network.
- An economical alternative to those who cannot afford expensive parallel computers.

[20] `http://en.wikipedia.org/wiki/Computer_cluster`

My Definition

- In my humble opinion any computer system that are connected by a network, but not a shared memory, can be considered as a cluster.
- The point is that they do not have shared memory so they can only communicate with the network.
- The network is not necessarily slow – some networks are extremely fast, so the term "loosely coupled" may not be true in all cases.

Tianhe Cluster

- Currently the fastest computer on Earth – 33.86 petaflops/second.
- Built by China's National University of Defense Technology (NUDT) in collaboration with the Chinese IT firm Inspur.[21]
- A huge cluster consisting of 16,000 computers. Each computer (node) has two Intel Ivy Bridge Xeon processors and three Xeon Phi chips. The total number of cores is 3,120,000.
- The TH Express-2 interconnect, designed by NUDT, utilizes a fat tree topology with 13 switches each of 576 ports[22].
- For me it is still a cluster.

[21] http://www.top500.org/featured/top-systems/tianhe-2-milkyway-2-national-university-of-defense/

[22] http://en.wikipedia.org/wiki/Tianhe-2

Hierarchy

- A Xeon is a multicore CPU.
- A Tianhe node is a multiprocessor.
- A Tianhe cluster is a multicomputer.
- Again we do not intend to have a clear distinction between a multicore CPU and a multiprocessor.
- More details in the "Architecture" lecture.

Discussion

- Describe the total number of cores within a node from any system in the top 10 of the top 500 list.

Parallelism

- Instruction-level parallelism
- Data parallelism
- Task parallelism

Instruction-level Parallelism

- Instructions can be re-ordered and combined into groups which are then executed in parallel without changing the result of the program.

Data Parallelism

- Data parallelism is a form of parallelization of computing across multiple processors in parallel computing environments by distributing the *tasks* across different parallel computing nodes.
- Often in the form of loops.

Data Parallelism Example

Example 1: Data Parallelism

```
1 for (i = 0; i < 1000000; i++)
2   array[i] = 0;
```

- All the assignment can be done in parallel.

Task Parallelism

- Task parallelism is a form of parallelization of computing across multiple processors in parallel computing environments by distributing the *tasks* across different parallel computing nodes.
- Also called functional parallelism. Often in the form of function calls.

Data Parallelism Example

Example 2: Task Parallelism

```
1  parallel do {
2    {
3      cook_dinner();
4      eat_dinner();
5    }
6    {
7      turn_on_radio();
8      listen_to_music();
9    }
10 } /* end of parallel do */
11 go_to_bed();
```

Discussion

- Give an example of data parallelism and task parallelism.

Dependency Graph

- Every node is task.
- Every edge is a dependency. This dependency is related to data or synchronization.
- A task can only starts when all tasks that precede it finish, i.e., the tasks must be done in a topological sort order.

Dependency Graph

- We can eat dinner only after we cook it.
- We can listen to music only after we turn on the radio.
- We can go to sleep only after we finish dinner and music.
- However, we can listen to the music while having dinner.

Wavefront

- Task parallelism must respect the dependency in dependency graph.
- One can image that task parallelism is a series of wave fronts in the dependency graph.
- We want to *increase* the number of tasks per wavefront in order to *reduce* the number of wavefront. That is, we want to increase the *parallelism* of our algorithm so that it takes less time to complete.

Discussion

- Draw the dependency graph of the previous task parallel example. You may add a *start* node and an *end* node to indicate the beginning and the end of execution. Also point out the wavefront and dependency in your drawing.

Quantities

- P The number of processors
- N The number of tasks
- L The longest path in the dependency graph
- W The maximum number of tasks in a wavefront
- T The execution time, assuming that it takes one unit of time to finish a task.

Execution Time

$$N \geq T \geq \max(L, \frac{N}{\min(P, W)}) \tag{1}$$

- Increase P.
- Increase W, i.e., parallelism.
- Decrease L.

Discussion

- Explain the inequality in the previous page.

Parallel Programming

- Program a parallel system to perform parallel computing.
- That is why we are here.
- more about this in the "Programming Model" lecture.

Automatic Parallel Programming?

- Can a program convert our sequential programs into parallel programs automatically?
- Apparently not, otherwise we will not be here.
- Computer scientists have been trying to invent "smart" compiler that automatically does it, but only have limited success.
- More details in the "Programming Model" lecture.

- It is very often that people talk about parallel and distributed computing.
- For example, the name of my laboratory is "Laboratory of Parallel and Distributed Computing".
- Many conferences and journals have both in their titles.
  - IEEE International Parallel and Distributed Processing Symposium
  - IEEE International Conference on Parallel and Distributed Systems
  - Journal of Parallel and Distributed Computing
  - IEEE Transactions on Parallel and Distributed Systems

Difference

Parallel Computing A set of processing unit to work on a job *at the same time*, i.e., the focus is that the computations is *temporarily* in parallel.

Distributed Computing A set of processing unit to work on a job *at different locations*, i.e., the focus is that the computations is *geographically* distributed.

Difference

- The focus of parallel processing is *performance*. We care very much about the speed we can finish a job.
- The focus of distributed processing is *reliability*. We care very much about can we finish a job no matter what happens – network down, hardware failure, earthquake, Godzilla attacks, etc.

Race-car

23

[23] http://upload.wikimedia.org/wikipedia/commons/4/4a/Formel3_racing_car_amk.jpg

Tank

24

[24] By Boevaya mashina - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=43925342

In this Course

- This course will focus on performance, so we mostly discuss parallel processing, and will briefly discuss distributed processing when necessary.
- Distributed computing has become increasingly popular because of cloud computing and big data processing.
- Nevertheless, the role of parallel processing is still crucial since the processing speed is still essential.
- The focus is to combine the speed of parallel processing and the reliability of distributed computing into a "parallel and distributed system".

Armored Vehicle

25

[25] By Dino246 at English Wikipedia - Transferred from en.wikipedia to Commons., Public Domain, https://commons.wikimedia.org/w/index.php?curid=3423910

Discussion

- Explain and describe the difference between parallel and distributed computing. Give examples to illustrate your points.