Parallel Computing
平行計算

Tien-Hsiung Weng
翁添雄
大葉大學資工系
thweng@pu.edu.tw

Lecture 1

Contact Information

- Instructor: Tien-Hsiung Weng 翁添雄
- Email: thweng@pu.edu.tw
- Office:靜宜大學管理學院二研**507**
- Phone: (04) 26318001~18107
- Office hours: by appointment

Course Objectives

- Learn fundamentals of parallel computing
  - Principles of parallel algorithm design
  - Parallel computer architectures
  - Programming models and methods
- Develop skill writing parallel programs
  - Programming assignments
- Develop skill analyzing parallel computing problems
  - Homework problems
  - Paper presentation

Topics

- Introduction
- Parallel computer architectures
  - Interconnection networks
  - Shared-memory systems and distributed-memory systems
- Principles of parallel algorithm design
- Programming tightly-coupled shared-memory systems: OpenMP and Cilk
- Programming distributed-memory systems
  - Message passing: MPI

Prerequisites

- Programming in C or Fortran
- Data structures
- Foundation of computer architecture

Essential Reading

- Text book:
  - Introduction to Parallel Computing, Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar, Addison-Wesley, 2003.

- Additional reading:
  - Parallel Programming in C with MPI and OpenMP, Michael J. Quinn, McGraw Hill,2004
  - Patterns for Parallel Programming, Timothy G. Mattson, Beverly A. Sanders, Berna L. Massingill, Addison-Wesley,2004
  - Parallel Programming with MPI, Peter S. Pacheco, Morgan Kaufmann Publishers, 1997
  - OpenMP Specification, www.openmp.org

Grading

– Midterm	20%
– Assignments	30%
– Papers presentation	50%

Why do we need Parallel Computing?

- The need for more computing power
- The limitation of serial computer
- The current trend in multi-core computer

Why do we need powerful computers?

- The need for more computational power (the need for speed)
  - Data mining
  - Earthquake simulation
  - Global climate modeling
  - Semiconductor design
  - Nuclear weapons test by simulation
  - Cryptography
  - Financial and economic modeling
  - Transaction processing, web services and search engines
  - Computation fluid dynamics (airplane design)
  - Crash simulation
  - Ocean simulation
  - Computational Chemistry
  - Computational Material Sciences and Nanosciences

Why do we need for more computational power?

- Suppose our computer performs one billion (10^9) calculations/second
- We want to predict the weather over US and Canada for the next 2 days:
  - Area of US and Canada is 20 million km^2
  - model the atmosphere from sea level to an altitude of 20 km
  - make prediction of the weather at each vertex of the grid (cubical grid), with each cube measuring 0.1 km on each side.
  - To predict weather one hour from now, each grid point takes about 100 calculations.

- So, we need total grid points:
  - $2.0 * 10^7$ km^2 * 20 km * 10^3 cubes/km^3 = $4*10^{11}$ grid points
  - in order to predict weather one hour from now, we need $4*10^{13}$ cals
  - To predict the weather at each hour for 2 days: $4*10^{13}$ cals/hour * 48 hours = $2 * 10^{15}$ cals
  - Using this computer it will take about $2 * 10^{15}$ calculations / 10^9 cals/secs = $2 * 10^6$ secs (23 days)

The limitation of a serial computer

- Can a serial computer execute the following code in one second?

```
for (i=0; I < ONE_TRILLION; i++)
        z[i] = x[i] + u[i]
```

- On a conventional computer, we successively fetch x[i] and u[i], store the result in z[i]. So, in order to execute this code in one second, it needs to carry out at least $2*10^{12}$ copies between memory and register / second

- If data travels from memory to CPU at the speed at light ($3*10^{8}$ meters/sec)

The limitation of a serial computer

- Let r be the average distance of a word of memory from the CPU, then r must satisfy

- $3*10^{12}$ * r meters = $3*10^{8}$ meters/sec * 1 sec
- r = 10^{-4} meters
- We need memory hardware layout in a regular rectangular grid.
- If we use a square grid with side length s and connect the CPU to the center of the square, then the average distance from a memory location to the CPU is about s/2. so we want s/2=r=10^{-4} meters, or s=$2*10^{-4}$ meters.
- If our memory words form a square grid, a typical row of memory words will contain

$$\sqrt{3*10^{12}} = \sqrt{3}*10^{6}\,\text{words}$$

- Thus, to fit a single word of memory into a square with side length measuring

$$\frac{2*10^{-4}\,meters}{\sqrt{3}*10^{6}} \approx 10^{-10}\,meters$$

- This is the size of a relatively small atom.
- We need a parallel computer

Unit of measure in HPC

- **High Performance Computing (HPC) units are:**
  - **Flops: floating point operations**
  - **Flop/s: floating point operations per second**
  - **Bytes: size of data (double precision floating point number is 8)**
- **Typical sizes are millions, billions, trillions…**

Mega	**Mflop/s = 10^6 flop/sec**	**Mbyte = 10^6 byte (also 2^{20} = 1048576)**
Giga	**Gflop/s = 10^9 flop/sec**	**Gbyte = 10^9 byte (also 2^{30} = 1073741824)**
Tera	**Tflop/s = 10^{12} flop/sec**	**Tbyte = 10^{12} byte (also 2^{40} = 10995211627776)**
Peta	**Pflop/s = 10^{15} flop/sec**	**Pbyte = 10^{15} byte (also 2^{50} = 1125899906842624)**
Exa	**Eflop/s = 10^{18} flop/sec**	**Ebyte = 10^{18} byte**

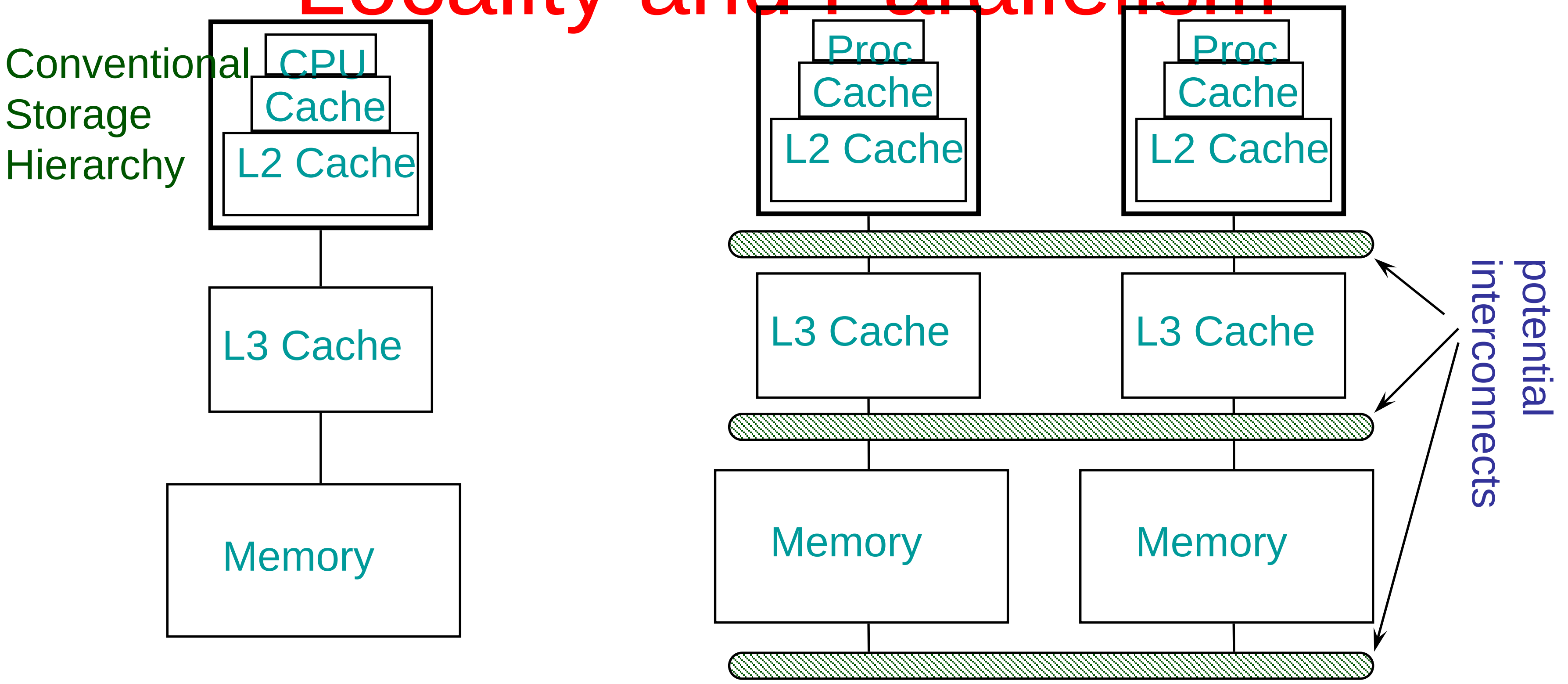
Impediments to Parallel Computing

- Software development is harder
  - Lack of standardized and effective development tools, programming models and environment
- Algorithm development is harder
  - Complexity of coordinating concurrent activities
- Rapid change in computer system architecture
  - Today’s parallel algorithm may not be suitable for tomorrow’s parallel computer

Impediments to Parallel Computing

- The complexity of how the processors will work together
- Having a collection of processors and memory, we must
  - Decide on and implement an interconnection network for the processors and memory modules
  - Design and implement system software for the hardware
  - Devise algorithms and data structures for solving our problem
  - Divide the algorithms and data structures up into subproblems
  - Identify the communications that will be needed among the subproblems
  - Assign subproblems to processors and memory modules

Issues in Parallel Performance: Locality and Parallelism



- Large memories are slow, fast memories are small
- Storage hierarchies are large and fast <u>on average</u>
- Parallel processors, collectively, have large, fast cache
  - the slow accesses to "remote" data we call "communication"
- Algorithm should do most work on local data

Finding Enough Parallelism: Amdahl's Law

- Suppose only part of an application seems parallel
- Amdahl's law
  - Let s be the fraction of work done sequentially, so (1-s) is the fraction parallelizable
  - Let P = number of processors

$$\text{Speedup}(P) = \text{Time}(1)/\text{Time}(P)$$

$$<= 1/(s + (1-s)/P)$$

$$<= 1/s$$

- Even if the parallel part speeds up perfectly, the sequential part limits overall performance.

Load Imbalance

- Load imbalance is the time that some processors in the system are idle due to
  - insufficient parallelism (during that phase)
  - unequal size tasks

- Examples of the latter
  - adapting to “interesting parts of a domain”
  - tree-structured computations
  - fundamentally unstructured problems

- Algorithm needs to balance load

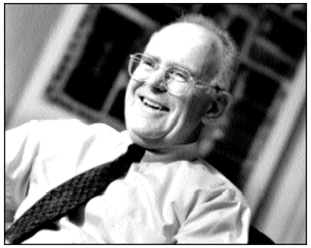
Today HPC Systems

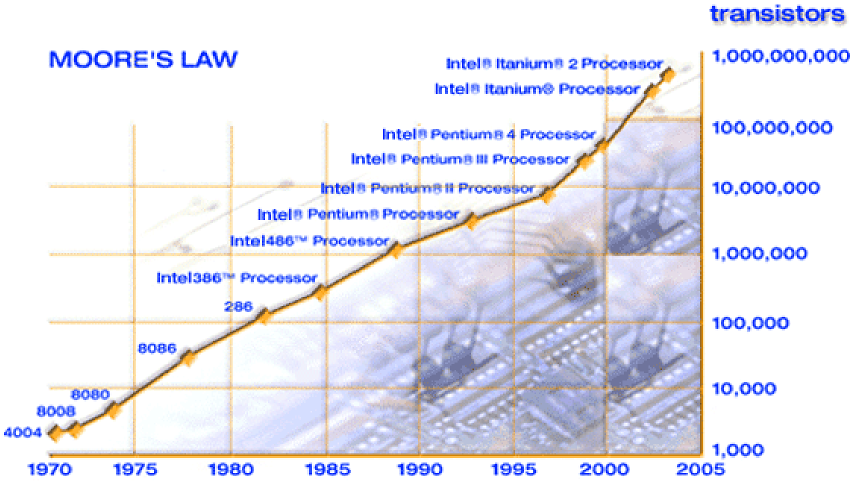
Rmax TF/s

Rank	Site	Manufactu	Computer	Country	Year	Processors	Rmax	Processor	Interconnect
1	DOE/NNSA/LLNL	IBM	eServer Blue Gene Solution	United States	2005	131072	280600	PowerPC 440	Proprietary
2	IBM Thomas Watson	IBM	eServer Blue Gene Solution	United States	2005	40960	91290	PowerPC 440	Proprietary
3	DOE/NNSA/LLNL	IBM	eServer pSeries p5 575 1.9 GHz	United States	2006	12208	75760	POWER5	Federation
4	NASA Ames	SGI	SGI Altix 1.5 GHz, Voltaire Infiniband	United States	2004	10160	51870	Intel IA-64 Itanium 2	Numalink/Infiniband
5	CEA	Bull SA	NovaScale 5160, Itanium2 1.6 GHz, Quadrics	France	2006	8704	42900	Intel IA-64 Itanium 2	Quadrics
6	Sandia National Laboratories	Dell	PowerEdge 1850, 3.6 GHz, Infiniband	United States	2006	9024	38270	Intel EM64T Xeon EM64T	Infiniband
7	GSIC Center, Tokyo Institute of Tech	NEC/Sun	Sun Fire X4600 Cluster, Opteron 2.4/2.6 GHz, Infiniband	Japan	2006	10368	38180	AMD x86_64 Opteron Dual Core	Infiniband
8	FZJ	IBM	eServer Blue Gene Solution	Germany	2006	16384	37330	PowerPC 440	Proprietary
9	Sandia National Laboratories	Cray Inc.	Red Storm Cray XT3, 2.0 GHz	United States	2005	10880	36190	AMD x86_64 Opteron	XT3 Internal Interconnect
10	The Earth Simulator Center	NEC	Earth-Simulator	Japan	2002	5120	35860	NEC	Multi-stage crossbar

Figure from top 500 list: http://www.top500.org June 2006

Advance of Technology: "Moore's Law"





Gordon Moore, Founder of Intel

1965: since the IC was invented, the number of transistors/inch^2 in these circuits roughly doubled every year. This trend would continue for the foreseeable future

1975: revised – circuit complexity doubles every 18 months

http://www.intel.com/technology/silicon/mooreslaw/pix/mooreslaw_chart.gif

Parallelism

- Definition: capable to execute parts of a program concurrently
- Goal: Shorter execution time
- Grain of parallelism: how bi are the parts?
  - Bit, instruction, statement, procedure, jobs
- Our focus: coarse-grain parallelism

Hierarchical Parallelism in Systems

- Multi-processor nodes
- SMP (Symmetric shared-memory) systems
- All processors share a bus to memory
- Direct-connect systems, i.e. Opteron
- Part of memory directly-connected to each processor
- Hardware accelerators
- Field programmable gate arrays
- Clusters
- Multiprocessor nodes
- Interconnection network
- Crossbar switch: all to all
- Mesh: nearest neighbor
- Multi-stage interconncetion networks