Parallel Processing

염헌영

Seoul National University
수업개요

- [dcslab.snu.ac.kr/courses/pp2010/](dcslab.snu.ac.kr/courses/pp2010/)

- 교재 1: Designing and Building Parallel Programs
  by Ian Foster

- 교재 2: Programming MPPs (draft)
  by David Kirk, Wen-mei Hwu

- 교재 3: GPU programming guide
Programming Languages/Systems

- MPI
- OpenMP
- CUDA
- Cilk
Motivation and History
Tunnel Vision by Experts

❖ “I think there is a world market for maybe five computers.”
  • Thomas Watson, chairman of IBM, 1943.

❖ “There is no reason for any individual to have a computer in their home”
  • Ken Olson, president and founder of Digital Equipment Corporation, 1977.

Slide source: Warfield et al.
“640K [of memory] ought to be enough for anybody.”

- Bill Gates, chairman of Microsoft, 1981.

“On several recent occasions, I have been asked whether parallel computing will soon be relegated to the trash heap reserved for promising technologies that never quite make it.”

- Ken Kennedy, CRPC Directory, 1994
Why Faster Computers?

- **Solve compute-intensive problems faster**
  - Make infeasible problems feasible
  - Reduce design time

- **Solve larger problems in same amount of time**
  - Improve answer’s precision
  - Reduce design time

- **Gain competitive advantage**
Definitions

- **Parallel computing**
  - Using parallel computer to solve single problems faster

- **Parallel computer**
  - Multiple-processor system supporting parallel programming

- **Parallel programming**
  - Programming in a language that supports concurrency explicitly
Why MPI?

- MPI = “Message Passing Interface”
- Standard specification for message-passing libraries
- Libraries available on virtually all parallel computers
- Free libraries also available for networks of workstations or commodity clusters

- MPI, the complete reference

Why OpenMP?

- OpenMP an application programming interface (API) for shared-memory systems
- Supports higher performance parallel programming of symmetrical multiprocessors

- OpenMP.org
Why Cilk?

- Cilk is a C/C++ extension which makes it easy to program multithreaded programs
- With the wide availability of multi-core CPUs, it’s the most natural way of programming

- www.cilk.com
Why CUDA?

- A quiet revolution!
Three Laws of Computing:

- **Moore’s Law.**
  - Transistors on a single chip doubles ~ every 18–24 months.

- **Gilder’s Law.**
  - Aggregate bandwidth triples ~ every year.

- **Metcalfe’s Law.**
  - The value of a network may grow exponentially with the number of participants.
Still, it is true in terms of the “# of transistors”.

http://en.wikipedia.org/wiki/Moore's_law
But, Things have changed. 
# of transistors ≠ clock speed

❖ Moore’s Law
  - For a long time, it meant that *clock speeds* would also *double* along the same timeline.
  - This trend has ground to a halt!

❖ Intel made an announcement in 2003
  - that they were moving away
    • from trying to increase the *clock speed*
    • to adding *more cores*
Is there speed limit?

Little progress have been made since the curve flattens out around 2007!
Moore’s Law Extrapolation:
Power Density for Leading Edge Microprocessors

10000
1000
100
10
1

Power Density (Watts/cm²)


Rocket Nozzle
Nuclear Reactor
Hot Plate

Power Density Becomes Too High to Cool Chips Inexpensively

Source: Shekhar Borkar, Intel Corp
Is the Speed of light fast enough?

- 299 792 458 m/s
- Modern computer clock speed
  - 3 GHz
  - 3,000,000,000 cycles per second
- How far can the light travel during 1 cycle?
  - About 10 cm

- Electronic circuits are small, but......
Consider the 1 Tflop/s sequential machine:

- Data must travel some distance, r, to get from memory to CPU.
- To get 1 data element per cycle, this means $10^{12}$ times per second at the speed of light, $c = 3 \times 10^8$ m/s. Thus $r < \frac{c}{10^{12}} = 0.3$ mm.

Now put 1 Tbyte of storage in a 0.3 mm x 0.3 mm area:

- Each bit occupies about 1 square Angstrom, or the size of a small atom.

No choice but parallelism
Revolution is Happening Now

- Chip density is continuing increase \( \sim 2x \) every 2 years
  - Clock speed is not
  - Number of processor cores may double instead
- There is little or no hidden parallelism (ILP) to be found
- Parallelism must be exposed to and managed by software

Source: Intel, Microsoft (Sutter) and Stanford (Olukotun, Hammond)
Multicore

- **What is it?**
  - a processing system composed of two or more independent cores.

- **Modification to hardwares**
  - Integration of cores
  - Interconnect between components
    - AMD: Direct Connect Architecture
    - Intel: QuickPath
  - Shared cache between cores
    - AMD: shared L3
    - Intel: shared L2
Traditional Scaling Process

Speedup
1.8x
3.6x
7x

User code

Traditional Uniprocessor

Time: Moore’s law

Moore’s law
Multicore Scaling Process

Unfortunately, not so simple...
Real-World Scaling Process

Parallelization and Synchronization require great care...
What is the problem here?

- Linear speedup is only possible when processes run totally independent of each other.
- Even then, OS interferes.
  - Shared data structures for
    - Scheduling
    - Interrupt handling

- If processes have to communicate (most cases), it gets worse......
No more free speedup, unless ...

- Redesign the Software to
  - exploit parallelism
  - provide scalability

Data Explosion comes to rescue!
  - Plenty of PARALLELISM
Multi-Core Products

**AMD**
- **Turion X2**
  - 2.1 ~ 2.4 GHz
  - L1: 128KB
  - L2/core: 512KB
- **Athlon X2**
  - 2.7 GHz
  - L1: 128KB
  - L2/core: 512KB
- **Phenom II X3**
  - 2.5 ~ 3.2 GHz
  - L1/core: 128KB
  - L2/core: 512KB
  - L3: 6MB
- **Phenom II X4**
  - 2.6 ~ 2.8 GHz
  - L1/core: 128KB
  - L2/core: 512KB
  - L3: 6MB
- **Opteron (Codename: Istanbul)**
  - L1/core: 128KB
  - L2/core: 512KB
  - L3: 6MB
  - 12, 16 cores will be ready on 2011.

**Intel**
- **Core2Duo**
  - 1.8 ~ 3.3 GHz
  - L1: 64KB
  - L2: 2 ~ 6MB
- **Core2Quad**
  - 2 ~ 3 GHz
  - L1/core: 128KB
  - L2: 4 ~ 12MB
- **Core i7**
  - 2.93 GHz
  - L1/core: 128KB
  - L2: 8MB

Each core implements “HyperThreading” ➔ Total 8 processing threads
Berkeley NOW Project (1995)
Beowulf Concept

- NASA (Sterling and Becker)
- Commodity processors
- Commodity interconnect
- Linux operating system
- Message Passing Interface (MPI) library
- High performance/$ for certain applications
**MANUFACTURER/COMPUTER** | **LOCATION** | **COUNTRY** | **CORES** | **R\text{max}**
---|---|---|---|---
1. IBM BladeCenter QS22/LS21, PowerXCell 3.2 GHz / Opteron 1.8 GHz, Voltaire Iband | DOE/NNSA/LANL | USA | 129600 | 110500
2. Cray XT5 QC 2.3 GHz | DOE/OS/BNL | USA | 150152 | 1059000
3. IBM Blue Gene/P Solution | Forschungszentrum Juelich | Germany | 29412 | 825500
4. SGI Altix ICE 8200EX, Xeon QC 3.0/2.66 GHz | NASA/Ames Research Center/NAS | USA | 51200 | 487005
5. IBM BlueGene/L - eServer Blue Gene Solution | DOE/NNSA/LLNL | USA | 212992 | 478200

**PERFORMANCE DEVELOPMENT**

**PROJECTED PERFORMANCE DEVELOPMENT**
Programming Parallel Computers

- Extend compilers: translate sequential programs into parallel programs
- Extend languages: add parallel operations
- Add parallel language layer on top of sequential language
- Define totally new parallel language and compiler system
Current Status

- **Low-level approach is most popular**
  - Augment existing language with low-level parallel constructs
  - MPI and OpenMP are examples
- **Advantages of low-level approach**
  - Efficiency
  - Portability
- **Disadvantage: More difficult to program and debug**
Thank You!

Q & A