Sackler Course BMSC-GA 4448

High Performance Computing in Biomedical Informatics

Class 2: Friday February 14^{th}, 2014
2:30PM – 5:30PM

AGENDA

- Recap 1^{st} class & Homework discussion.
- Fundamentals of Parallel Computing: Multi-Processing
- Unix Hands-On:
    - Data sharing and File Permissions
- Student Projects

Single-Processor Performance Growth

- Moore's Law: Processor performance doubles every 18 months

(Gordon E. Moore, Intel co-founder, 1965)

(1) Increased transistor density

(2) Higher clock rates

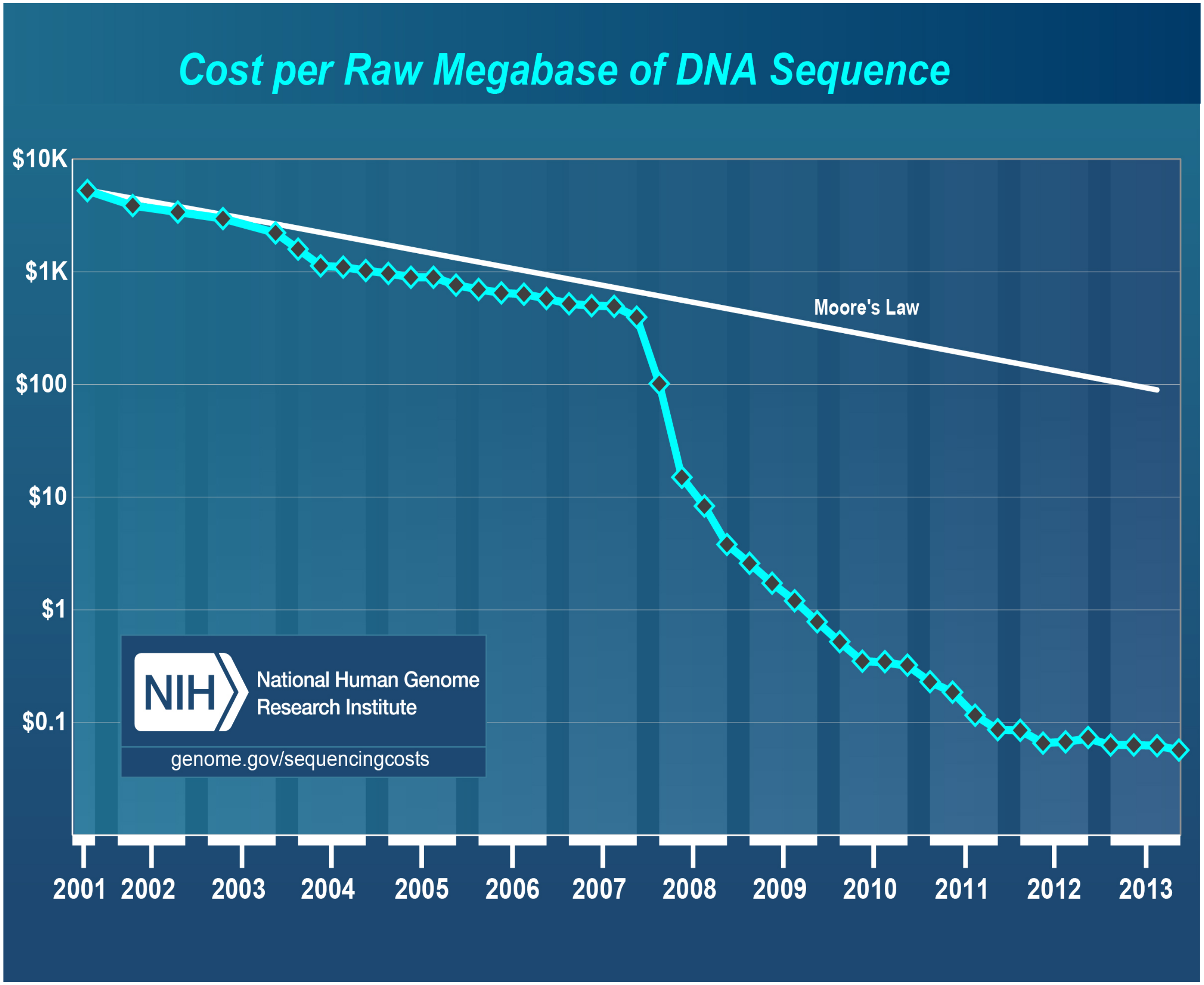
- Exponential growth of Single-Processor performance for over 40 years

1971: 0.7 MHz (Intel 4004)

2001: 1,600. 0 MHz

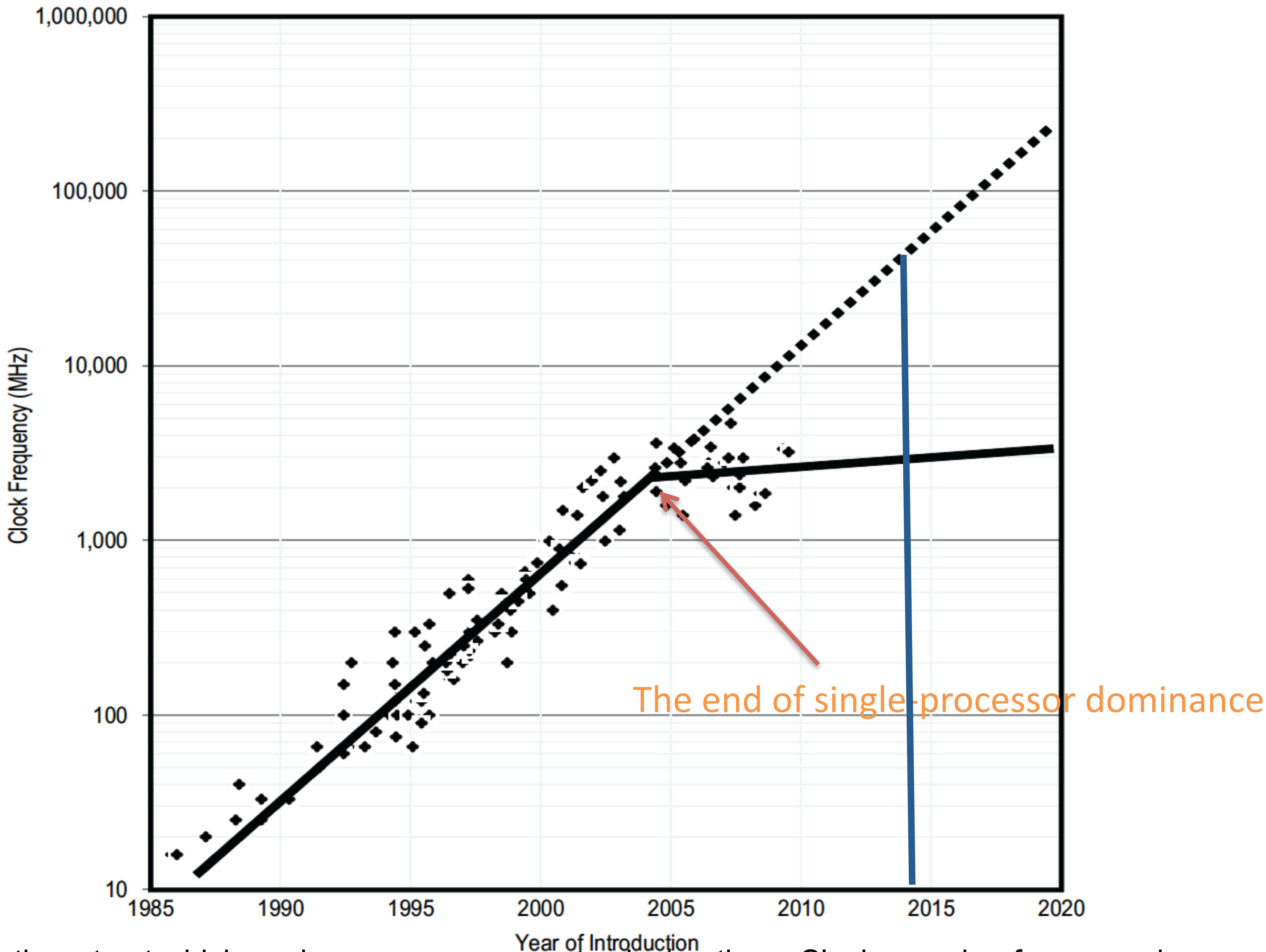
- Maintain compatibility of specific instruction sets.

- Exponential growth in computing power transformed the way we do business, interact and socialize.



Advances in DNA sequencing technologies vs. Moore's Law

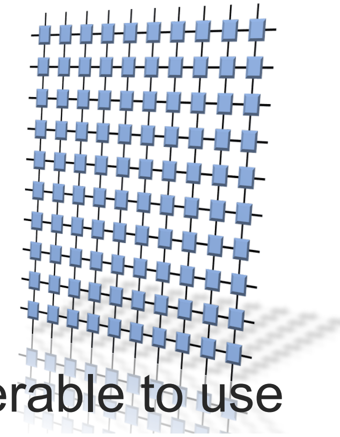
Single-Processor Clock Frequency Growth



Clock speed: the rate at which a microprocessor executes instructions. Clock speed or frequency is expressed in MHz or GHz.

National Academies Press: 2011-The Future of Computing Performance: Game Over or Next Level?

Multi-core Processors



Power ~ Frequency^3

- Horizontal growth: More processing cores per chip. It is preferable to use many slower devices than a single ultrafast device.

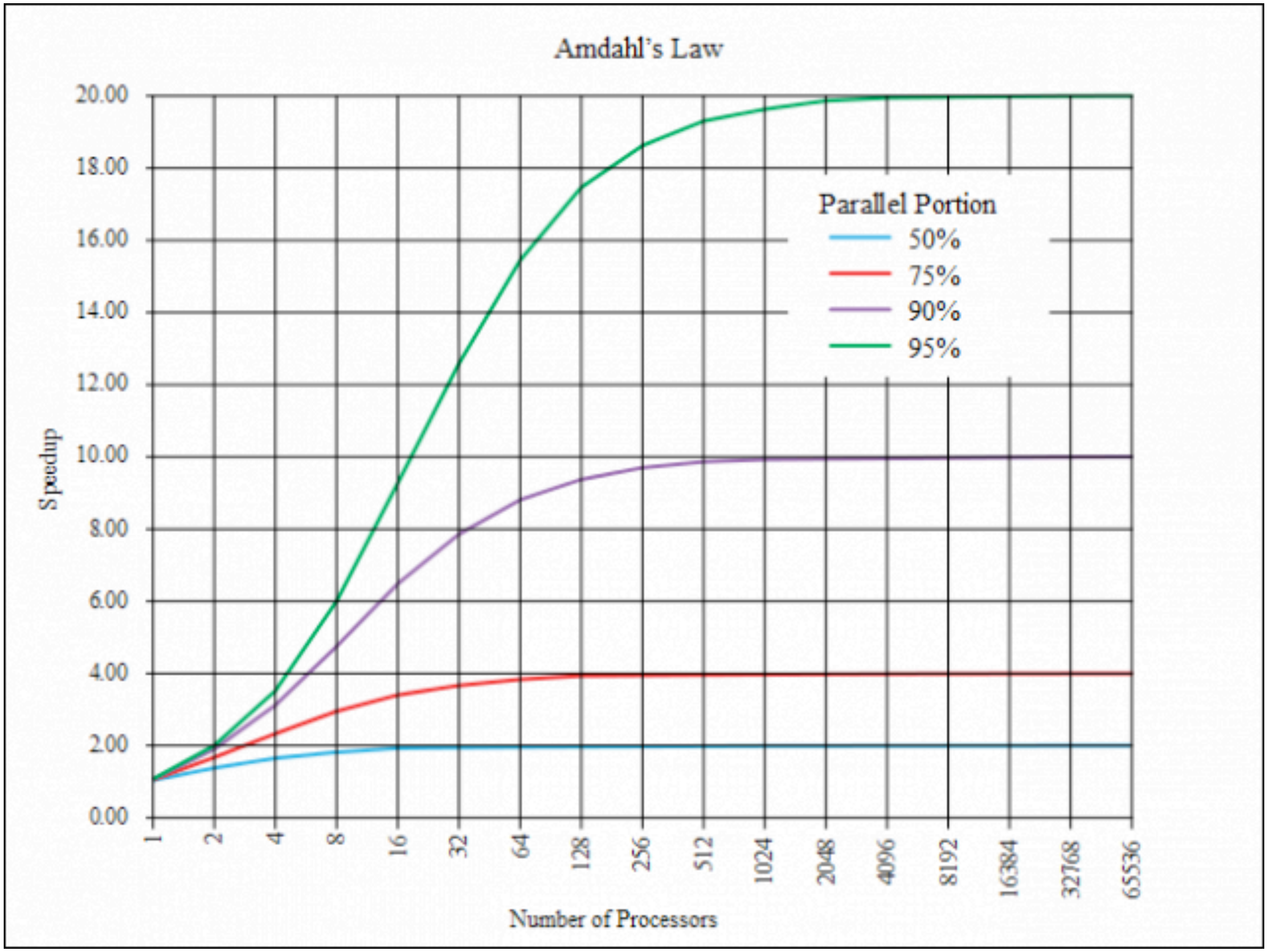
- Energy efficient cores to turn more transistors into more performance.

- New challenges for software developers:
  Sequential (serial) programs do not automatically benefit from the use of multiple cores. Examples.

- Parallel computing enters mainstream computing.

- Can innovation in software and algorithms enable ongoing performance growth?

3 GHz + 3 GHz < 6 GHz



In a parallelized program, performance is limited by the amount of serial code

$$SpeedUp = \frac{1}{(1-P)+\frac{P}{N}}$$

P: Parallel Portion

N: Processing Elements

Cache: memory set aside as a specialized buffer storage memory that is continually updated ; Used to optimize data transfers between system elements with different characteristics.

Cache memory bridges the speed gap between the processor and the memory.

Figure 2a. Multi-core processor separate L2

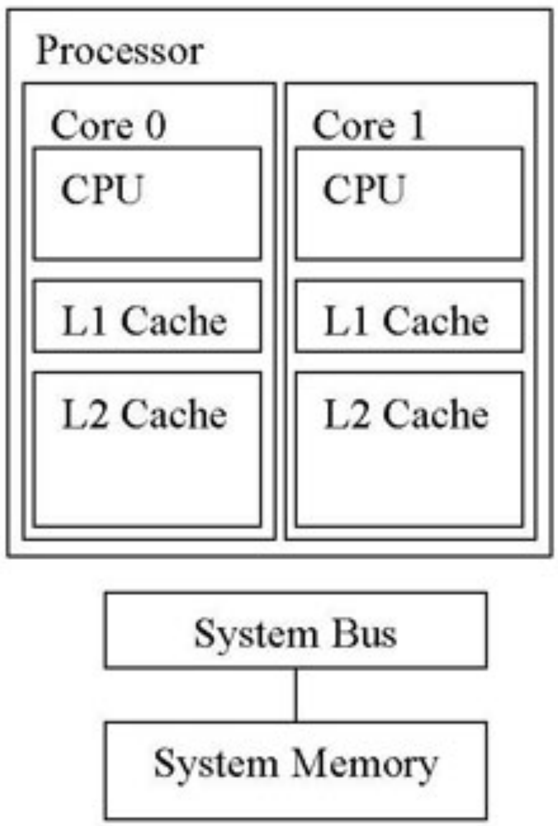
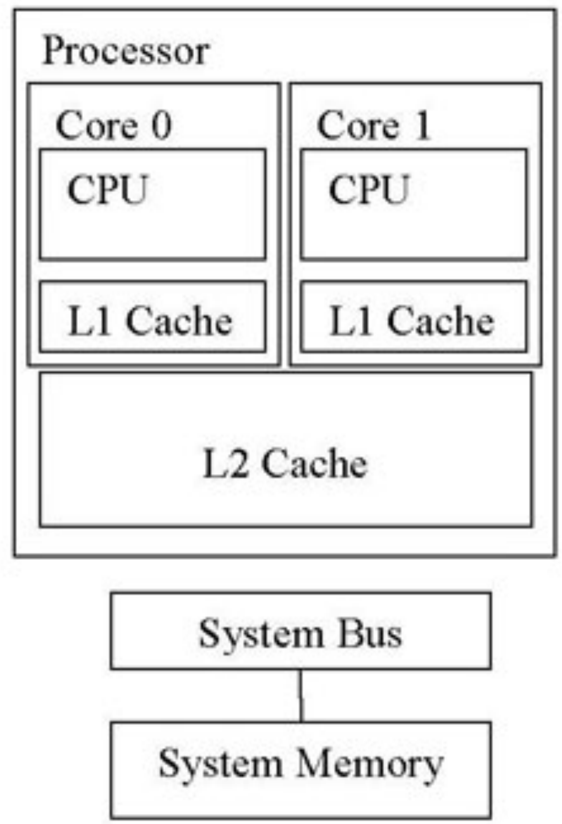
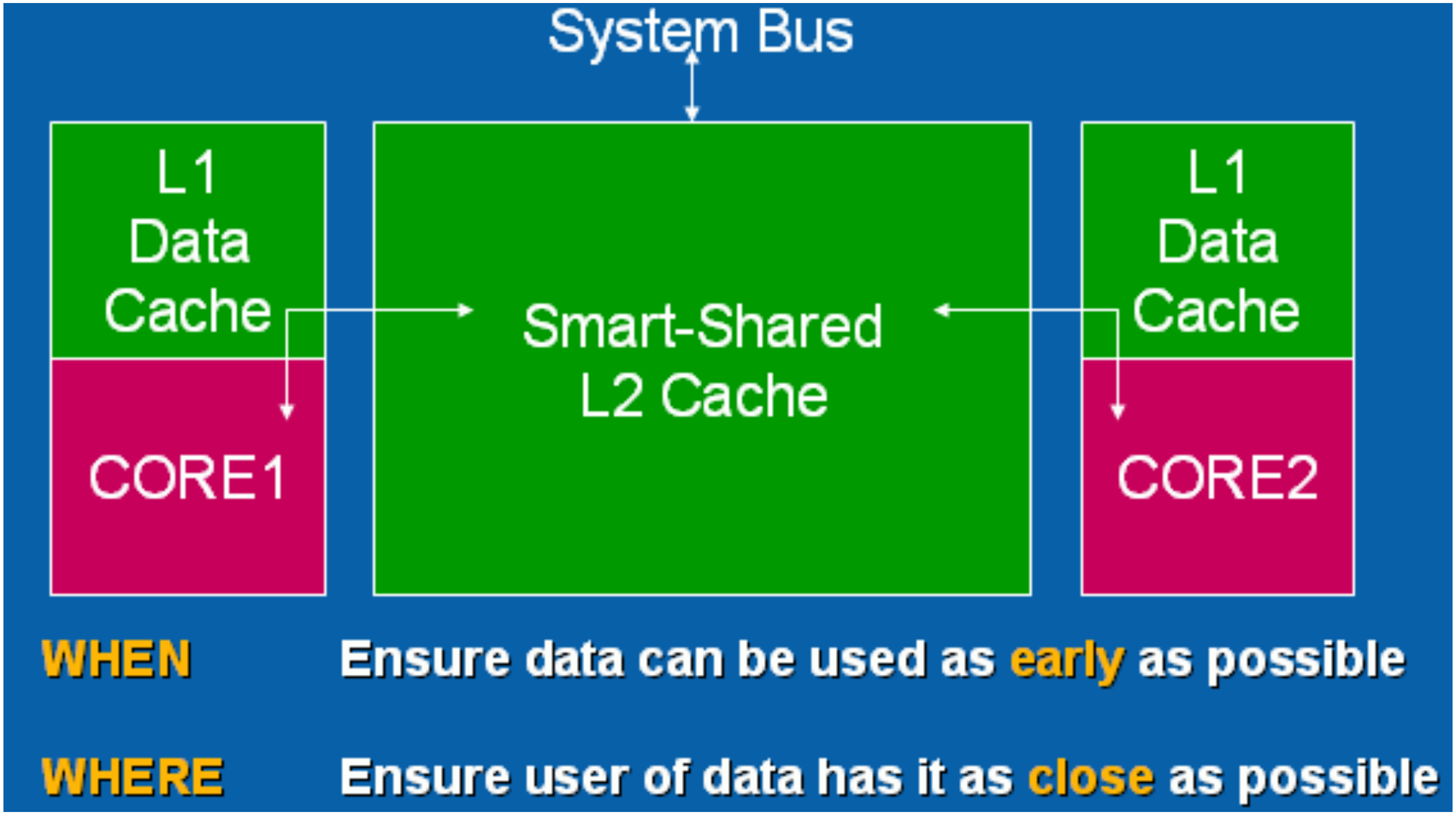


Figure 2b. Multi-core processor shared L2



http://www.drdobbs.com/development-and-optimization-techniques/212600040



Advantages of smart/shared L2 Cache:

- Size allocated per core can be dynamically adjusted with the potential of the total level 2 cache being available to applications that require it.

- Sharing of items between cores. Threads on separate cores can synchronize through the faster level 2 cache as opposed to main memory or the next level of cache.

http://www.drdobbs.com/development-and-optimization-techniques/212600040

Questions/Discussion

BREAK

- How many CPUs, cores are in your laptop? A cluster compute node?
- What is the clock speed of the processor on your laptop? A cluster compute node?
- How much system RAM is in your laptop? A cluster compute node?
- How much L2 cache is on the processors of your laptop? A cluster compute node

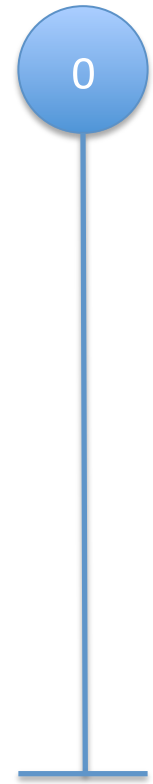
Hint: On the cluster, use:

```
cat /proc/cpuinfo
cat /proc/meminfo
```

Computer Processes

Problem: We need to find out how many words are in a book.

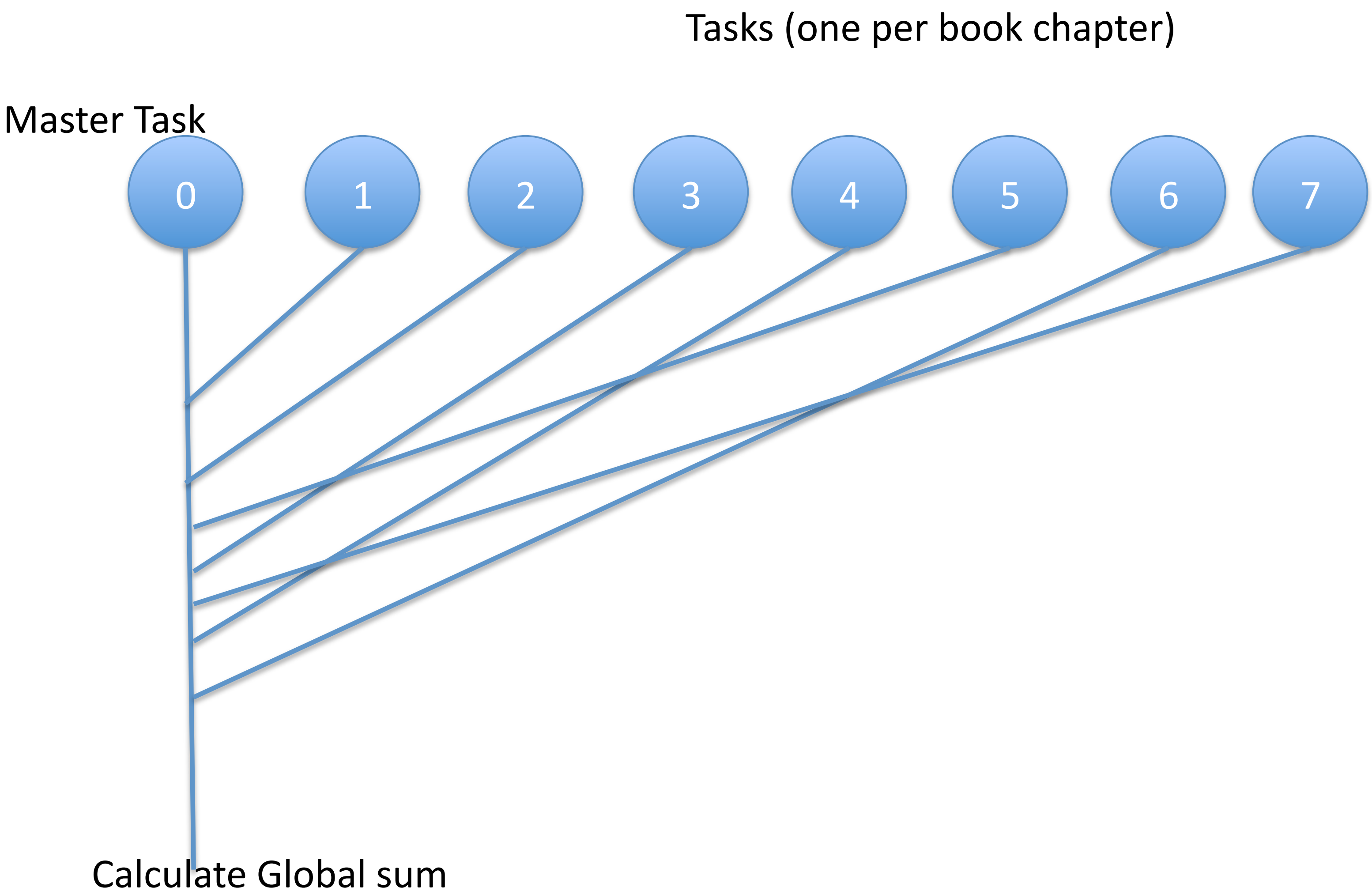
Serial Job : A *single* task (*a process*)



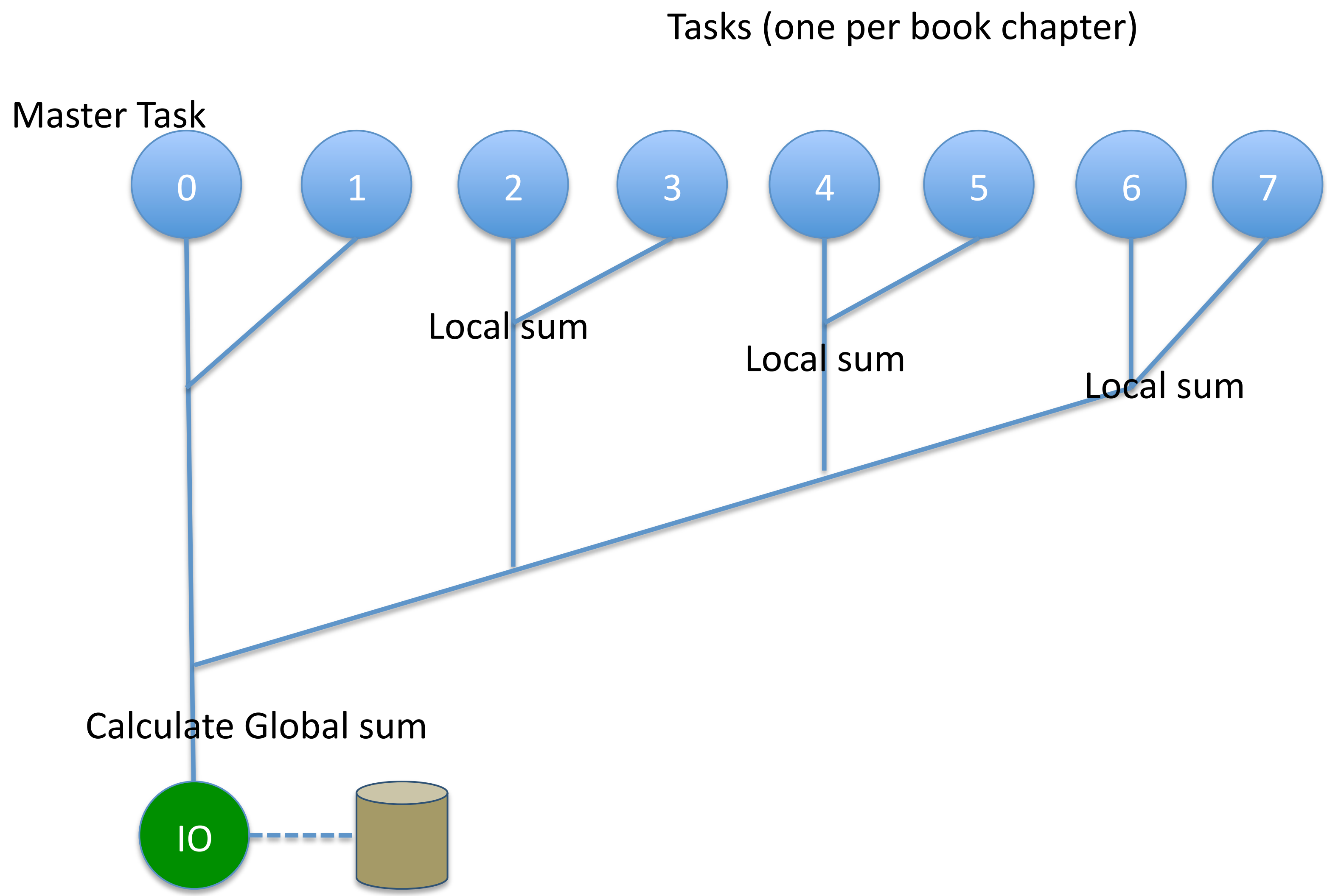
Serial pseudo-code:

```
sum = 0;

Open (BOOK);

while (in BOOK)  {
    Read( something, BOOK);
     if (something.is.word) sum = sum + 1;
}

Close (BOOK);

print (sum);
```

Problem: We need to count how many words are in a book



Problem: We need to count how many words are in a book



Parallelism Considerations: **Communications**

Splitting a large problem into a large number of tasks:

decreases the execution time attributable to computation, but
increases the execution time attributable to communication.

(1) Communication Latency
(2) Communication Bandwidth
(3) Communication Pattern (nearest-neighbor, broadcast, one-to-all, etc.)

NetPerf, *Iperf* (Open Source tools) provide accurate measurements of network latency and bandwidth.

The Unix *ping* command:

Used commonly to check network connectivity, some basic network characteristics, find out the IP address of a remote server, etc.

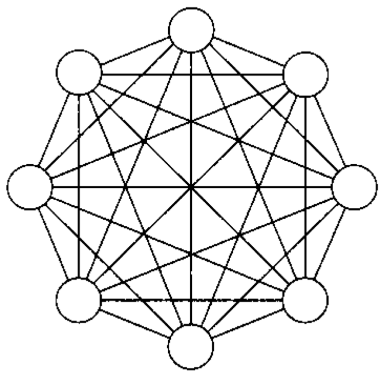
Try on your laptop's terminal:
- *ping phoenix.med.myu.edu*

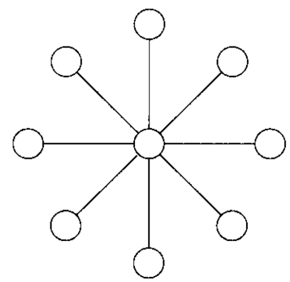
To Exit, press Control-C

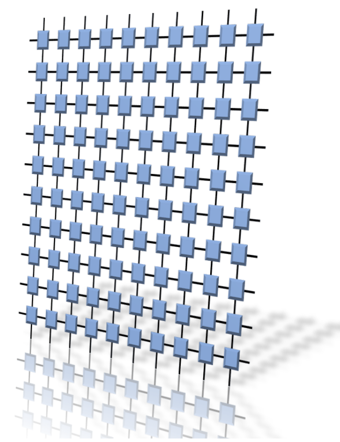
- *ping –c 5 google.com*

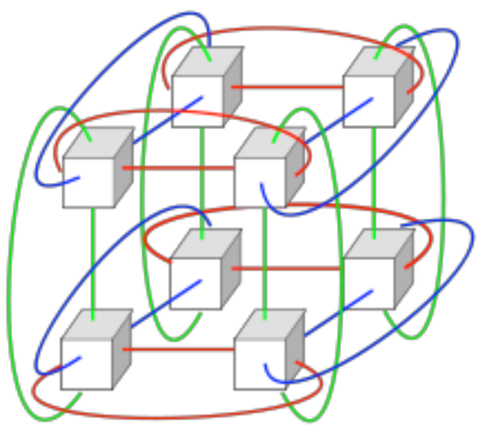
- *man ping*

Parallelism Considerations: **Communications/ Node Interconnects**

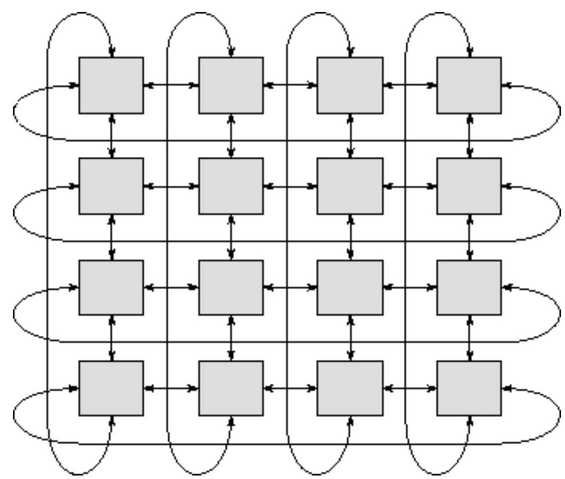








2 x 2 x 2 Torus



2 x 2 Torus

Parallelism Considerations: ***I/O Operations***

Many tasks making up a parallel job may need to read/write to the same file system, at the same time, over a conventional network. This can become a serious bottleneck.

(1) Reduce I/O when possible.
(2) Parallel file systems (GPFS, Lustre, etc.) can be helpful.
(3) Stage-in, Stage-out. On a cluster environment, consider writing/reading from local (on the node) file system, rather than a shared file system over the network.
(4) Create unique filenames using filename extensions that correspond to a task Id is a good practice.
(5) Separate IO from message-passing: The IO network is usually separate from the inter-node network.
(6) IO-Nodes: Only a small number of nodes have access to the external file system. The rest of the nodes (compute nodes) will need to send IO operations to an IO node. P-set ratio.
What is the p-set ratio of the HPC cluster?

Sackler Course BMSC-GA 4448

High Performance Computing in Biomedical Informatics

Class2 Homework

(1) Read cloud-related docs at :
https://genome.nyumc.org/hpcf/wiki/HPC_course/spring_2014#Session_3

(2) Create a directory, under your home directory on phoenix, called *HPCCache*

- Put a test file, named *dataFile ,* in the HPCCache directory so that hpcclass group members can view the contents of the *dataFile* and also copy the file.
- Allow members of the hpcclass group to write files in the HPCCache directory.
- Create a directory under your home directory, called HPCtest where members of the hpcci group can write files.

(3) Write a Shell script that runs the *uptime* command on the first 10 nodes of the HPC cluster and sorts the output based on the 1-min average load on each node.

Sackler Course BMSC-GA 4448

High Performance Computing in Biomedical Informatics

Class 3: Thursday February 20th, 2014, 2:30PM

AGENDA

- Cloud Computing (presented by Dr C. Krampis, JCVI)