What is Concurrent Programming?

Concurrent Program: When two or more ‘processes’ cooperate to achieve a common goal.

‘Processes’ are also called ‘threads’ or ‘tasks’. In this course we use the terms interchangibly.

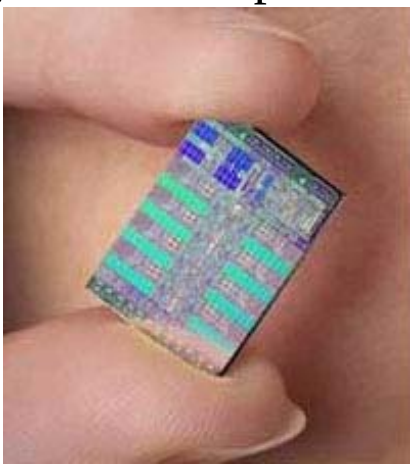
- Multiple *threads of control*
  - A number of sequential programs (i.e. ‘ordinary’ programs work together to achieve a goal
- Inter-process Communication
  - Shared variables
  - Message passing
- Synchronization
  - Mutual exclusion — processes must execute their *critical sections* one at a time.
  - Conditional synchronization — processes wait until a condition is true.

Note: Concurrent programming does not require multi-processors.

Computing Today

Parallel hardware

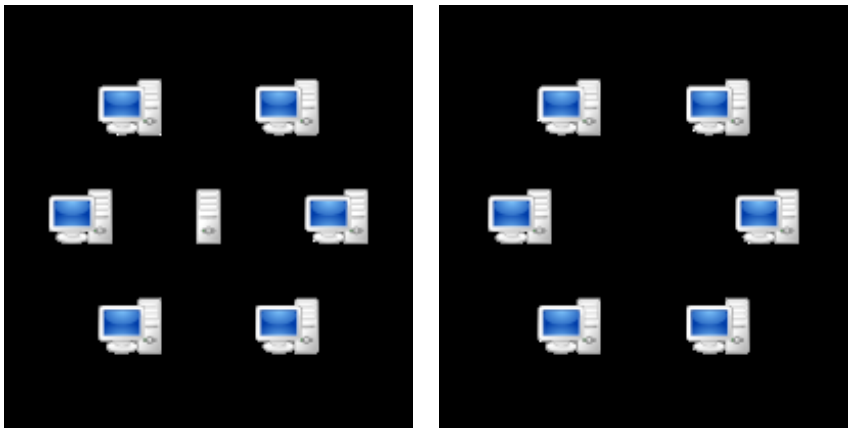
- Moore's law meets the law of diminishing returns
- It's hard to use 1 billion transistors effectively for a single CPU
- Instead manufacturers are putting 2 or 4 CPUs on a chip (multi-core)
- How many will it be in 10 years?
- And they are executing multiple threads with each CPU
  - Either by context switching or by
  - simultaineous multithreading (Intel calls it hyperthreading)
- Even the PS3 game system has 9 processors on one chip.



- Programs won't go any faster on parallel hardware unless they are parallel programs.

Distributed Computing

- The internet and other networks mean that applications are distributed across a network.
- The different parts of such applications must communicate with each other.

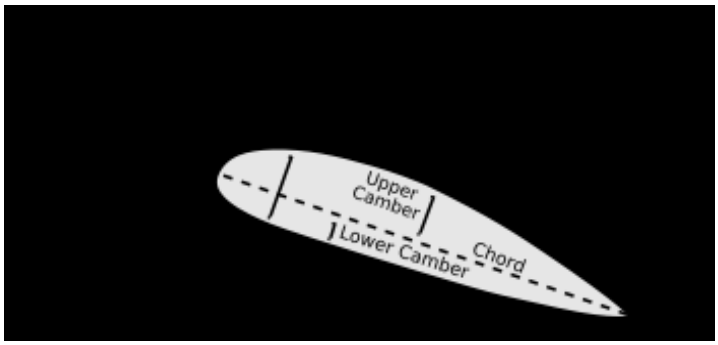


Big problems

Advances in computing only fuel the need for "big iron" computing.

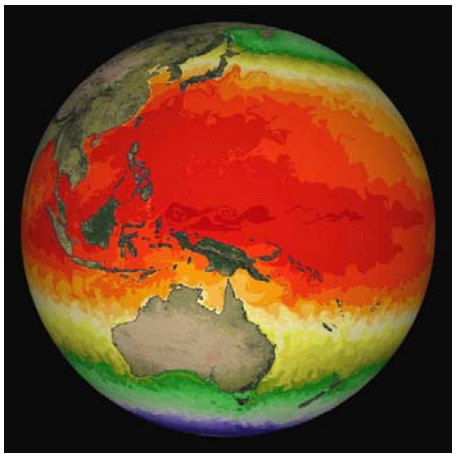
- Such problems require parallel computation for timely solution
- Example: airplane wings used to be designed in 2-D cross-

section

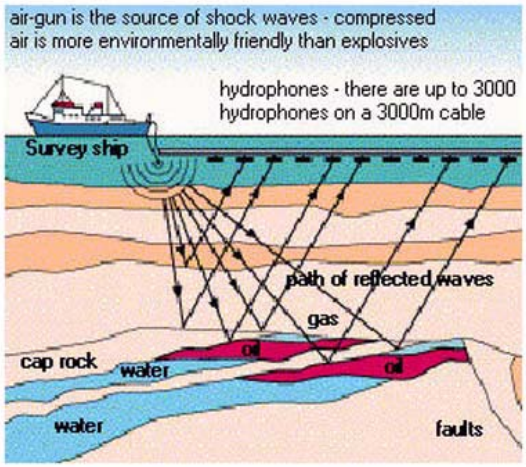


Computers allow 3-D modelling of the whole plane, leading to novel designs

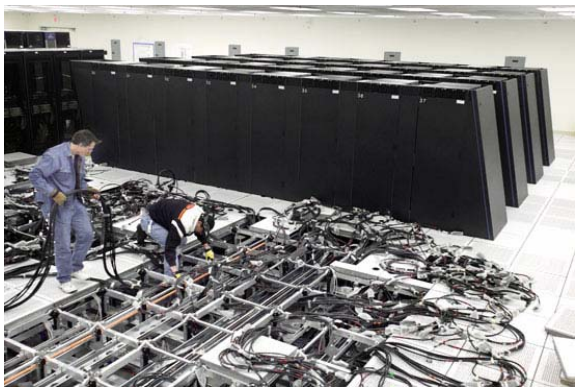
- Example: Modelling climate change.



- Example: Sesmic oil exploration.



- Example: Bioinformatics. The Blue Gene super computer has 32000 processors



Hardware is inherently parallel

As we move to behavioural hardware design, we find that hardware design *is* parallel programming.

Why Use Concurrent Programs?

- Faster processing (when ≥ 1 processor)
- More effective use of resources (e.g. disks)
- Faster response to user
- System is conceptually concurrent
- Fault tolerance
- System is distributed

What's different about it?

- Program steps from different processes may be inter-leaved or concurrent.
- Need to consider other processes.
- Usual reasoning rules don't apply.
- Programs may fail partially.
- Testing is never sufficient.

Course Outline (Approximate)

- Following text reasonably closely
- Programming will be done in Java

Topic	Lectures (Very Approx)
Architectures & Applications	2
Processes & Synchronization	3
Locks & Barriers	3
Semaphores	2
Monitors	3
Message passing	3
Software Transactional Memory	2
RPC & Rendezvous	2
Interaction Paradigms	4
Scientific Computing	4
Real-time systems (?)	4
Model checking (?)	3
Transaction processing (?)	2