SFWRENG 4F03
Parallel Computing

Winter 2020

02 Parallel Computing

Dr Asghar Bokhari

Faculty of Engineering, McMaster University

January 6, 2020

How to handle a large complex task?

- Two ways:
  - Sequential: One person works on different parts of the task one after the other
  - Parallel: A number of persons work on differnt parts (in parallel) that are combined to complete the task.

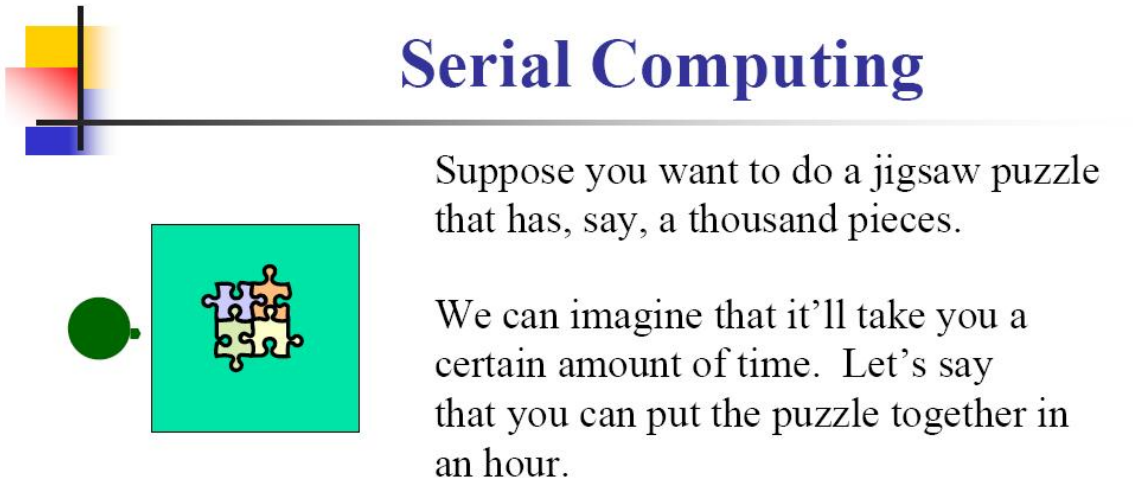
  Examples:
  - Move all chairs of this room to another room
  - Construction - different tasks by specialised workers
  - Making cars - different sub-assemblies by different companies

How to handle a large complex task?

- Advantage of second approach:
  - Same job in less time
  - Much bigger jobs can be handled in reasonable time

  Steps involved:
  - Decomposition of task into smaller independent parts
  - Employing many workers each working on the smaller part
  - Coordination

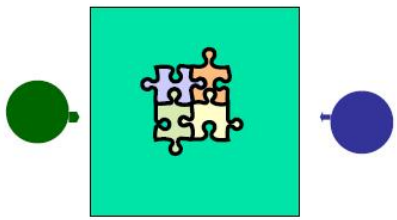
Jig saw puzzle example [1]



[1] Taken from OU Supercomputing Center for Education and Research, University of Oklahoma

Jig saw puzzle example

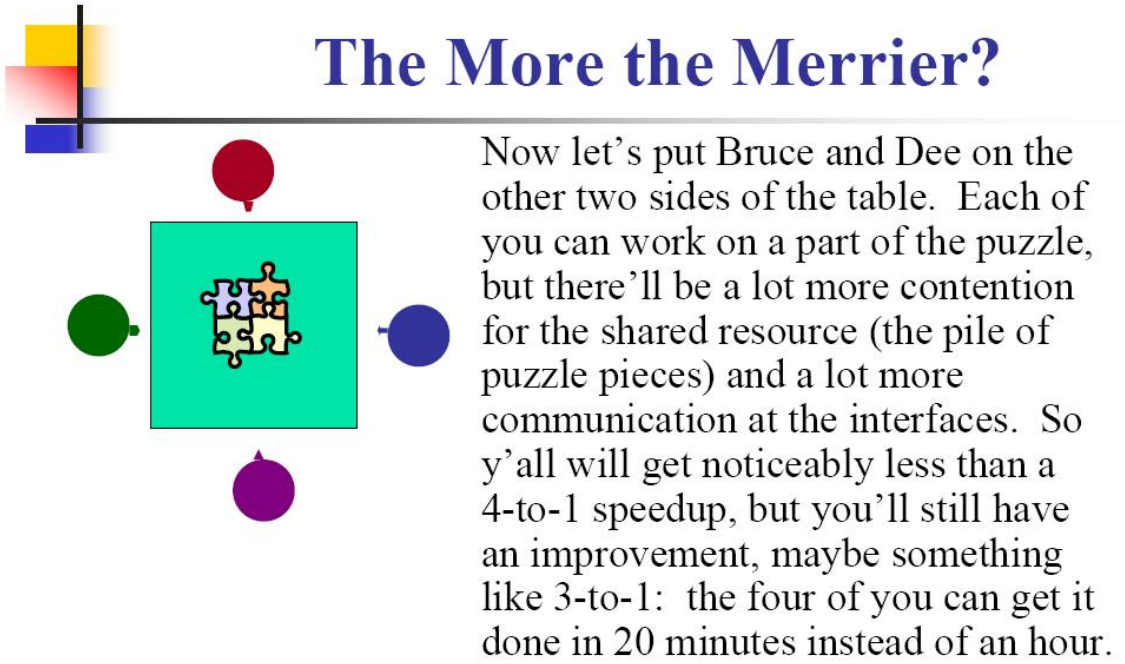
Shared Memory Parallelism



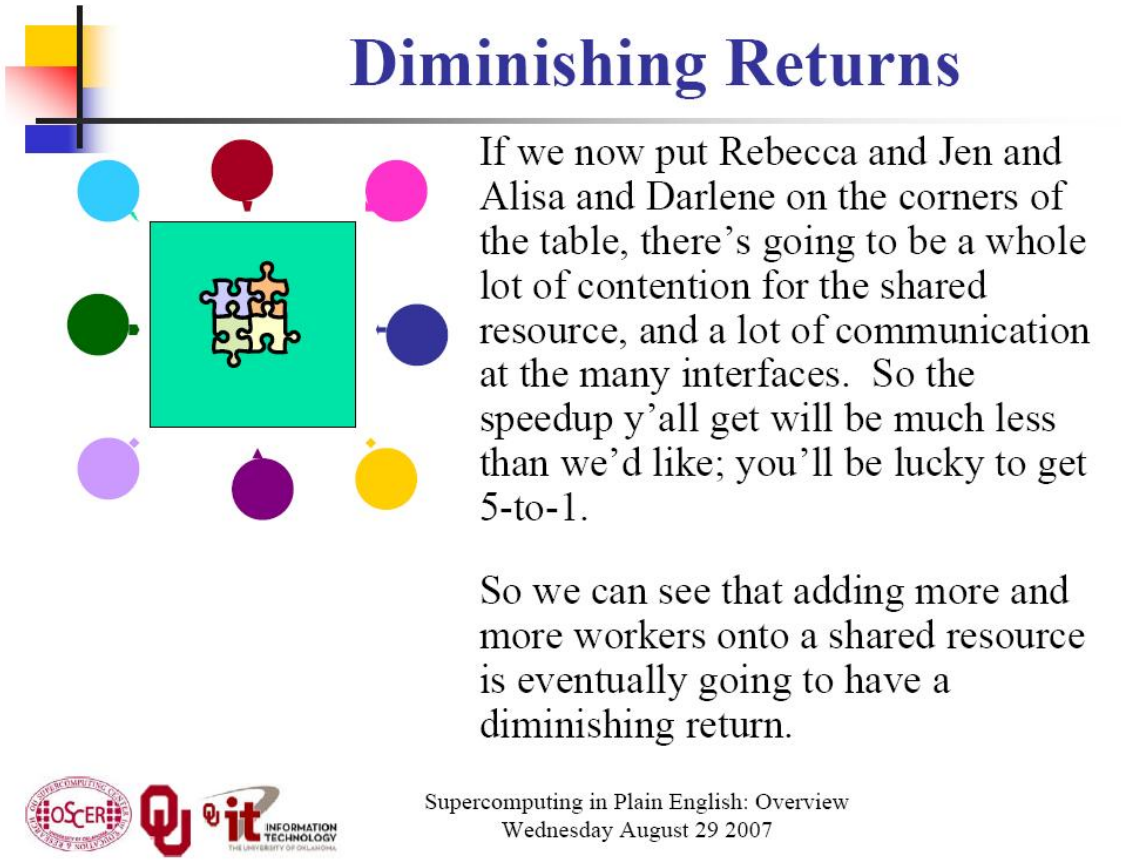
If Horst sits across the table from you, then he can work on his half of the puzzle and you can work on yours. Once in a while, you'll both reach into the pile of pieces at the same time (you'll ***contend*** for the same resource), which will cause a little bit of slowdown. And from time to time you'll have to work together (***communicate***) at the interface between his half and yours. The speedup will be nearly 2-to-1: y'all might take 35 minutes instead of 30.

Supercomputing in Plain English: Overview
Wednesday August 29 2007

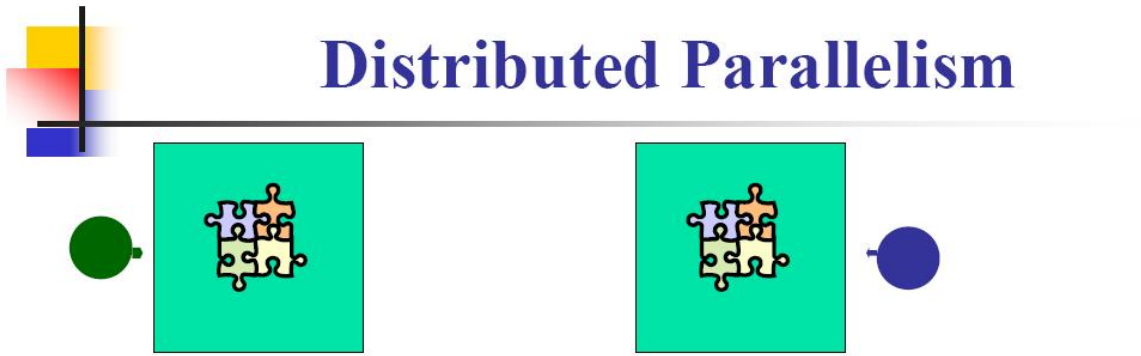
Jig saw puzzle example



Jig saw puzzle example



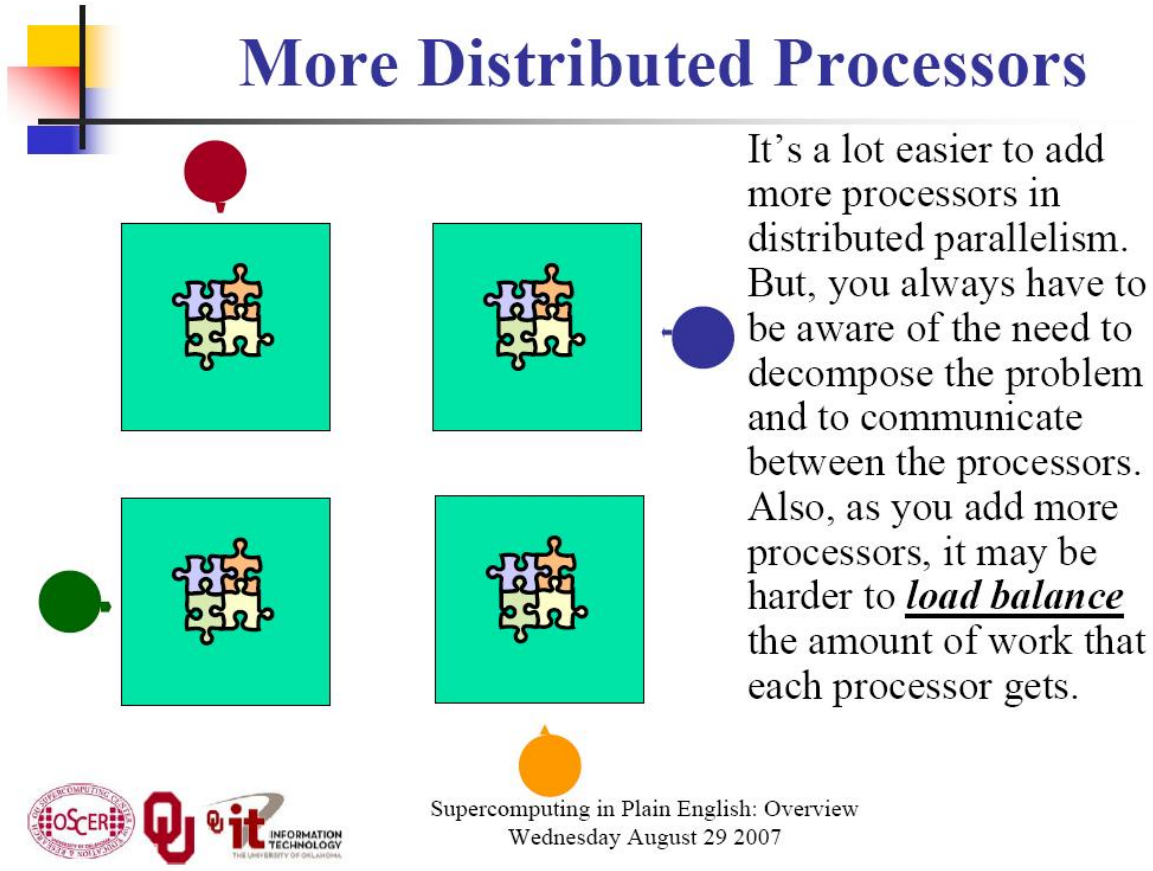
Jig saw puzzle example



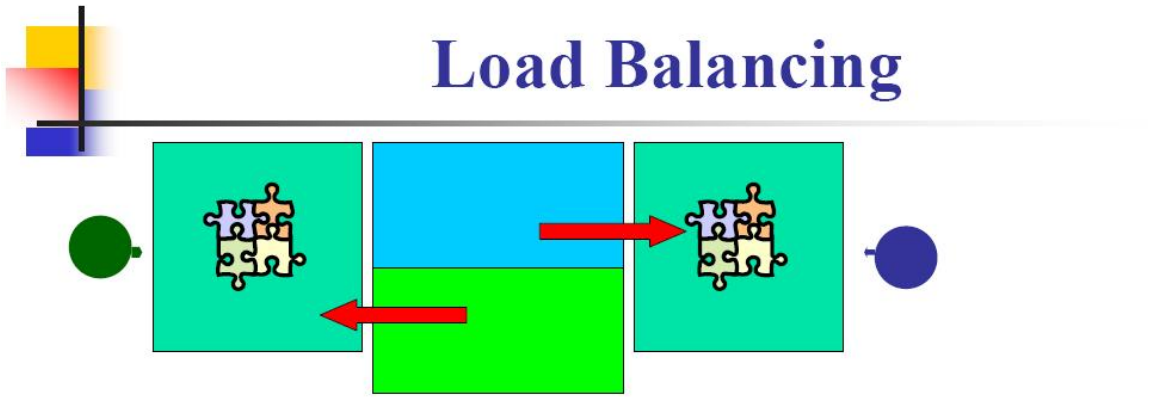
Now let's try something a little different. Let's set up two tables, and let's put you at one of them and Horst at the other. Let's put half of the puzzle pieces on your table and the other half of the pieces on Horst's. Now y'all can work completely independently, without any contention for a shared resource. **BUT**, the cost of communicating is **MUCH** higher (you have to scootch your tables together), and you need the ability to split up (***decompose***) the puzzle pieces reasonably evenly, which may be tricky to do for some puzzles.

Supercomputing in Plain English: Overview
Wednesday August 29 2007

Jig saw puzzle example



Jig saw puzzle example

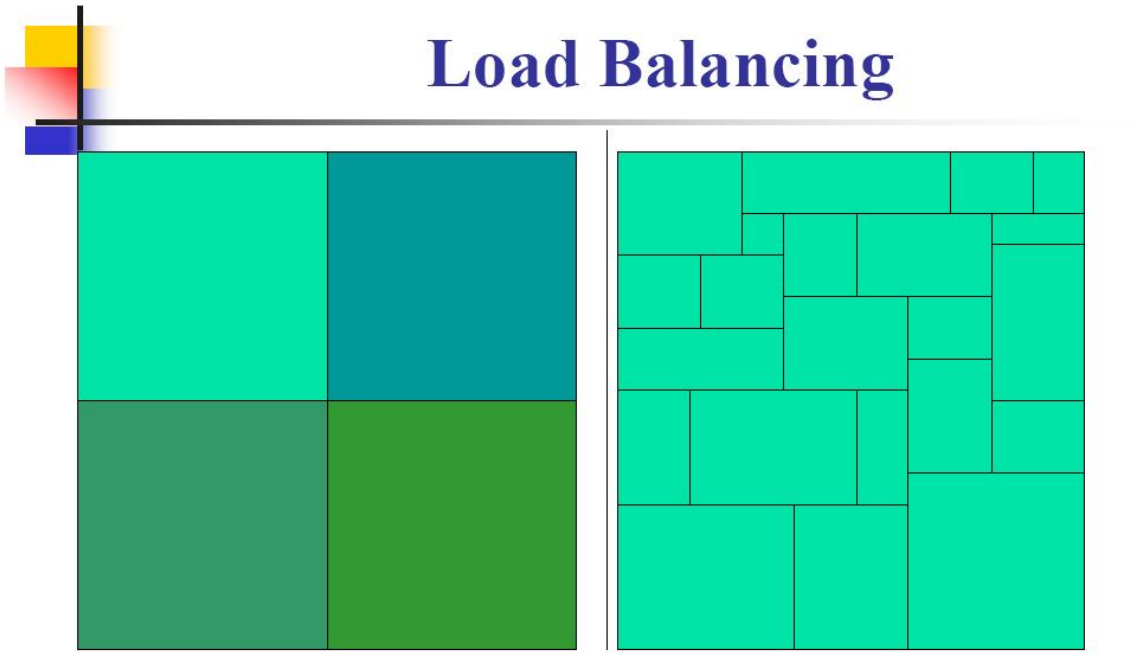


Load balancing means giving everyone roughly the same amount of work to do.

For example, if the jigsaw puzzle is half grass and half sky, then you can do the grass and Julie can do the sky, and then y'all only have to communicate at the horizon – and the amount of work that each of you does on your own is roughly equal. So you'll get pretty good speedup.

Supercomputing in Plain English: Overview
Wednesday August 29 2007

Jig saw puzzle example



Load balancing can be easy, if the problem splits up into chunks of roughly equal size, with one chunk per processor. Or load balancing can be very hard.

Supercomputing in Plain English: Overview
Wednesday August 29 2007

Parallel Programming

- Traditionally computers carry out tasks by executing one instruction at a time sequentially
- We can solve bigger problems in less time by using more than one computers in parallel.
- But
  - Processing power of CPUs have increased tremendously recently
  - Some of these CPUs can execute one instruction in 9 to 12 billionth of a second
  - Is that not fast enough?
- Several problems require much more speed
- Many problems interesting to scinetists and engineers will take months, even years to run on a PC but may take a few hours on a supercomputer.

Parallel Programming

- There are probelms that may not fit on a PC because they need more RAM or disk space. Some examples of these problems:
  - Simulation
  - Climate modeling
  - Fluid Turbulence
  - Pollution Dispersion
  - Human Genome
  - Ocean Circulation
  - Glaxy information
  - Data Mining
  - Visualization - turning large amounts of data into pictures that scientists can understand.

Parallel Programming

- But, I am not into all these things!!
  - Learning how to run your code faster, empowers you to handle more complex problems in less time
  - Historically, what happens in supercomputing today, will be in your destop 10 to 15 years from now
  - Learning it today puts you ahead of the curve.
  - Applications such as www, web services, on line banking and on line retailers, oil exploration, data mining, advanced graphics and virtual reality, computer aided diagnostics in medicine already use clusters to handle complexity and traffic.
  - parallelism is not restricted to supercomputers alone but it is getting popular in workstations and personal computers also. Future programs will be required to exploit multiple processors inside each computers in addition to those available over the network

Alternatives

- Traditional Supercomputers
  - Very fast single computers with large memory bandwidth
  - Some successful applications
  - Use sequential programming that is most commonly understood by programmers
  - Supported by a lot of experience in compilerss and other tools
  - Interconnects not required
  - very expensive
  - generates a lot of heat requiring special cooling
  - performance of a single CPU reaching its limit

Alternatives

- **Use more than one computers working in parallel**
  - Take advantage of the advances made in processor, networks and memory fields
  - Use off-the-shelf components
  - Possibilites to quickly integrate new elements into a system for scalability or for taking advantage of improvements in a partcular technology
  - Usually much cheaper compared to single processor supercomputers
  - Sequential programs won't run
  - Not many programmers know parallel programming techniques
  - Mature tools and compilers not available

Alternatives

- A number of different arrangements have been tried
- Location of processors
  - Many processors (CPUs) located inside a single enclosure
  - A number of computers interconnected together to act as a supercomputing platform - multicomputer systems, clustering
- Interconnection
  - Different types of direct connections
  - Connection via Ethernet
- Access to memory
  - Shared memory
  - Distributed memory
  - Distributed shared memory
  - Use of cache memory

Alternatives

- Software considerations
  - How to decompose a program into seperate independent parts that can run in parallel on a single or multiple processes?
  - Parallel algorithms
  - Operating systems to handle parallel programs

Alternatives

- Different parallel programming models
  - Shared memory: tasks share a common address space to read from and write to, asynchronously.
    - Access to shared memory controlled by locks, semaphores and other similar mechanisms
    - Program development may be simplified
    - Compilers translate user program variables to actual global memory addresses
    - Management and understanding of data locality difficult
  - Message passing
  - Instruction level parallism
  - Multithreading
  - Data parallel systems
  - Hybrid systems

Road Map

- Review of single processor arhitecture
- CISC, RISC
- Memory and cache organization
- Details of different hardware arrangements/classifications for parallism
- Software considerations
- Message Passing Computing
- Message Passing Interface (MPI)
- Parallel Virtual Machine (PVM)
- Pthreads
- OpenMP
- Strategies and algorithms
- Applications

Title

-
-
- Template!