VIT®
BHOPAL
www.vitbhopal.ac.in

CSE3009 - Parallel and Distributed Computing

Course Type: LTP Credits: 4

Prepared by
Dr Komarasamy G
Senior Associate Professor
School of Computing Science and Engineering
VIT Bhopal University

INTRODUCTION TO PARALLEL COMPUTING

OBJECTIVES:

To introduce you to the basic concepts and ideas in parallel computing

To familiarize you with the major programming models in parallel computing

To provide you with with guidance for designing efficient parallel programs

OUTLINE:

- Introduction to Parallel Computing / High Performance Computing (HPC)
- Concepts and terminology
- Parallel programming models
- Parallelizing your programs
- Parallel examples

What is High Performance Computing?

What is High Performance Computing?

What is High Performance Computing?

Odyssey supercomputer is the major computational resource of FAS RC:

- 2,140 nodes / 60,000 cores
- 14 petabytes of storage

What is High Performance Computing?

Odyssey supercomputer is the major computational resource of FAS RC:

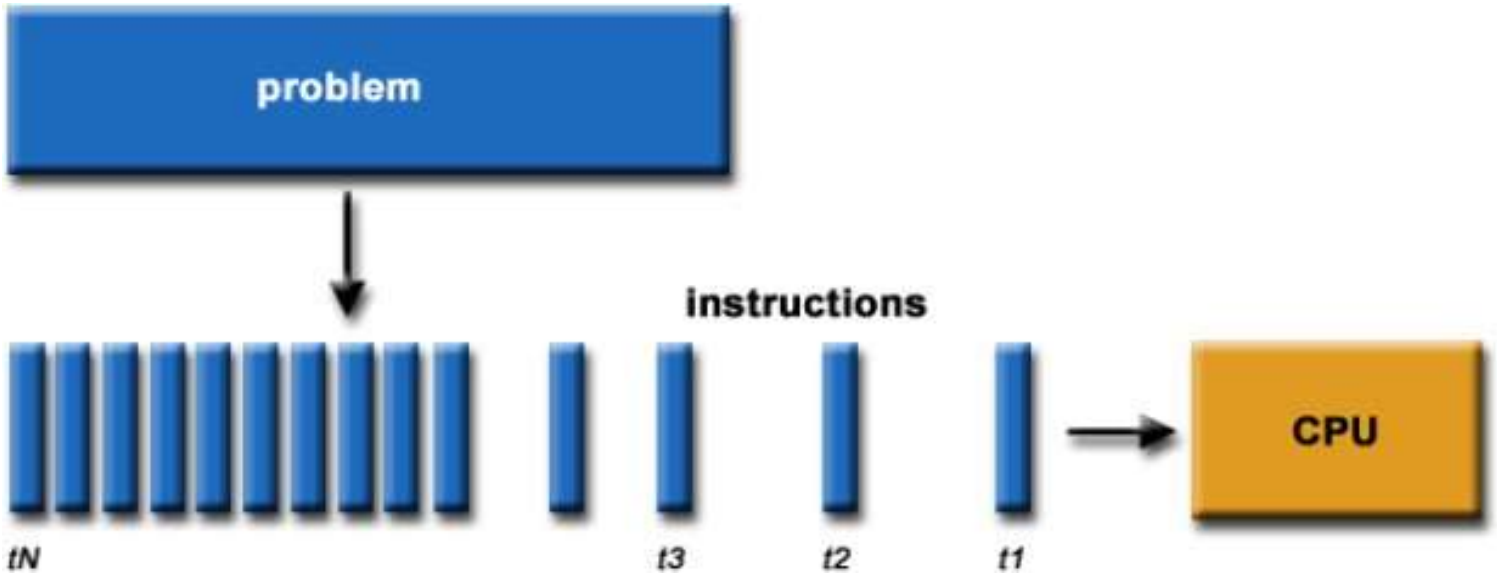
- 2,140 nodes / 60,000 cores
- 14 petabytes of storage

Using the world's fastest and largest computers to solve large and complex problems.

Serial Computation:

Traditionally software has been written for serial computations:

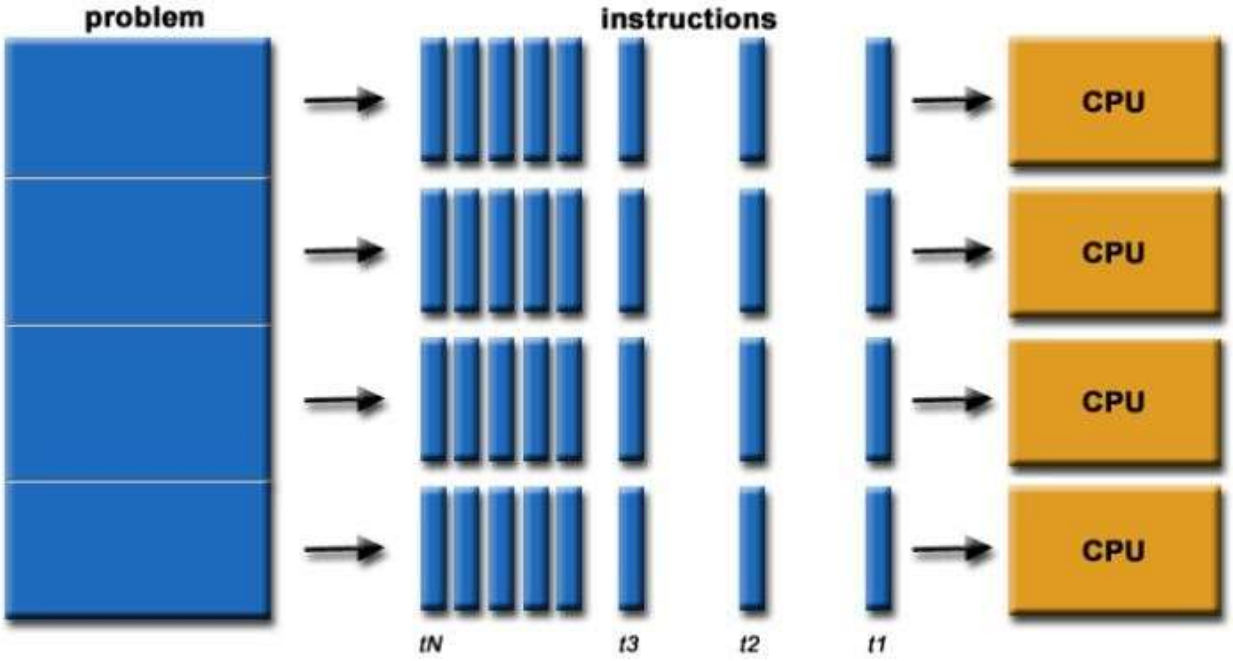
- To be run on a single computer having a single Central Processing Unit (CPU)
- A problem is broken into a discrete set of instructions
- Instructions are executed one after another
- Only one instruction can be executed at any moment in time



Parallel Computing:

In the simplest sense, parallel computing is the simultaneous use of multiple compute resources to solve a computational problem:

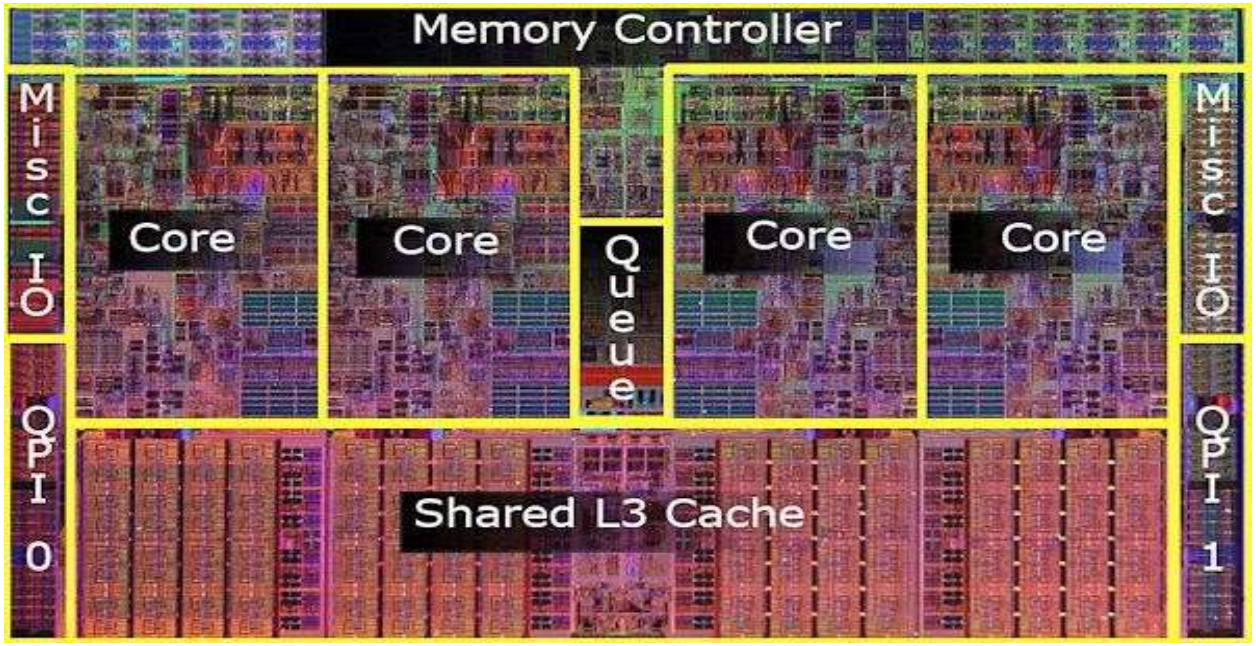
- To be run using multiple CPUs
- A problem is broken into discrete parts that can be solved concurrently
- Each part is further broken down to a series of instructions
- Instructions from each part execute simultaneously on different CPUs



Parallel Computers:

Virtually all stand-alone computers today are parallel from a hardware perspective:

- Multiple functional units (floating point, integer, GPU, etc.)
- Multiple execution units / cores
- Multiple hardware threads



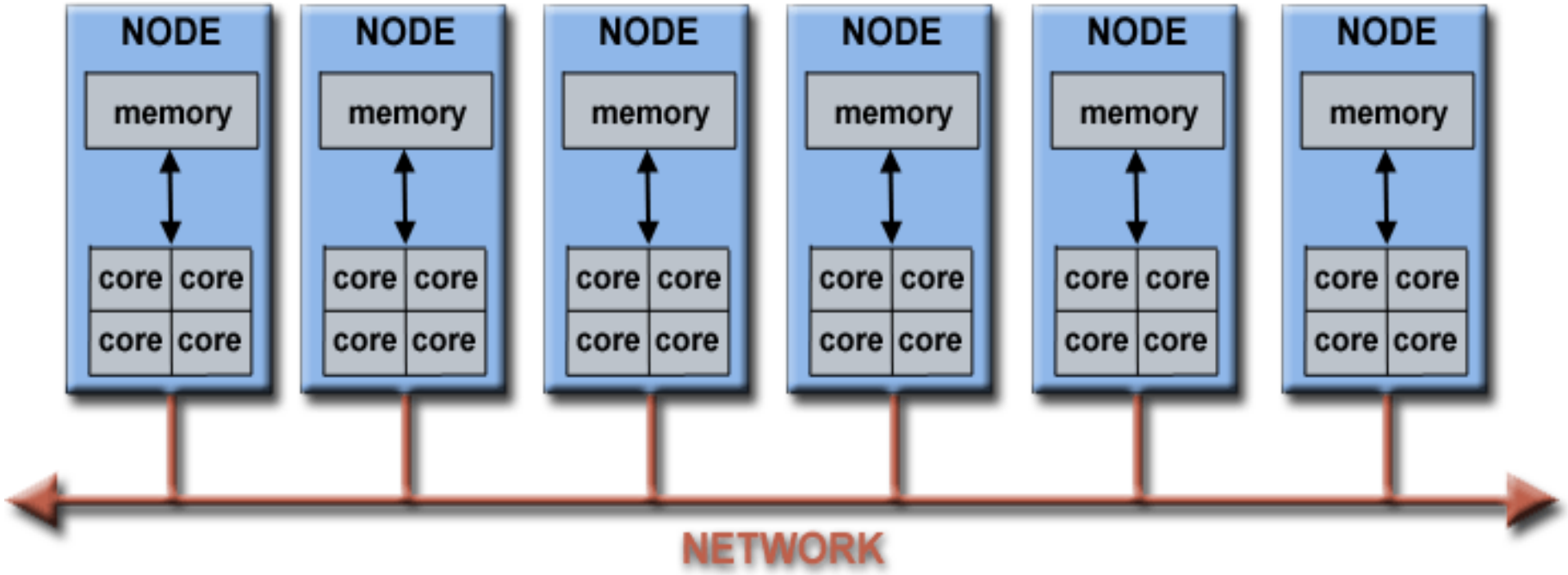
Intel Core i7 CPU and its major components

Image Credit: Intel

Parallel Computers:

Networks connect multiple stand-alone computers (nodes) to create larger parallel computer clusters

- Each compute node is a multi-processor parallel computer in itself
- Multiple compute nodes are networked together with an InfiniBand network
- Special purpose nodes, also multi-processor, are used for other purposes



Why Use HPC?

Major reasons:

Save time and/or money: In theory, throwing more resources at a task will shorten its time to completion, with potential cost savings. Parallel clusters can be built from cheap, commodity components.

Solve larger problems: Many problems are so large and/or complex that it is impractical or impossible to solve them on a single computer, especially given limited computer memory.

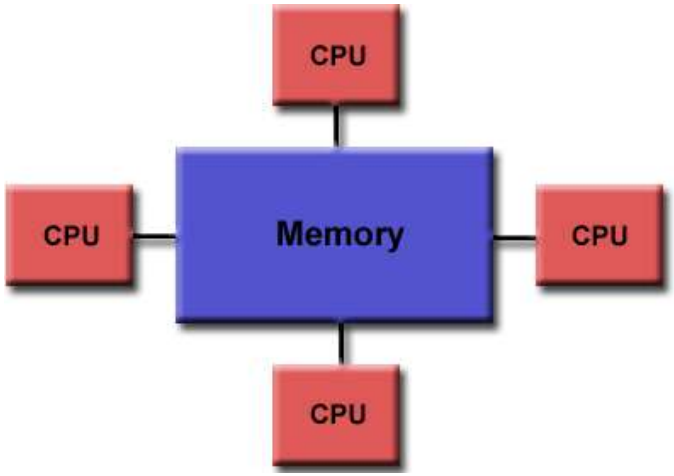
Provide concurrency: A single compute resource can only do one thing at a time. Multiple computing resources can be doing many things simultaneously.

Use of non-local resources: Using compute resources on a wide area network, or even the Internet when local compute resources are scarce.

HPC Terminology:

- **Supercomputing / High-Performance Computing (HPC)**
- **Flop(s)** – Floating point operation(s)
- **Node** – a stand alone computer
- **CPU / Core** – a modern CPU usually has several cores (individual processing units)
- **Task** – a logically discrete section from the computational work
- **Communication** – data exchange between parallel tasks
- **Speedup** – time of serial execution / time of parallel execution
- **Massively Parallel** – refer to hardware of parallel systems with many processors ("many" = hundreds of thousands)
- **Pleasantly Parallel** – solving many similar but independent tasks simultaneously. Requires very little communication
- **Scalability** - a proportionate increase in parallel speedup with the addition of more processors

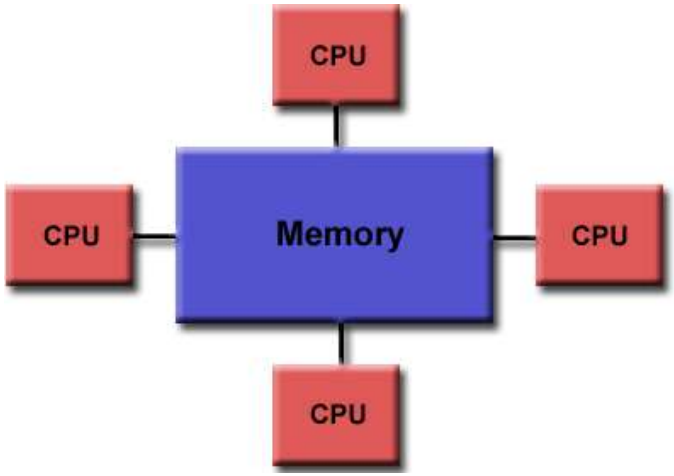
Parallel Computer Memory Architectures:



Shared Memory:

- Multiple processors can operate independently, but share the same memory resources
- Changes in a memory location caused by one CPU are visible to all processors

Parallel Computer Memory Architectures:



Shared Memory:

- Multiple processors can operate independently, but share the same memory resources
- Changes in a memory location caused by one CPU are visible to all processors

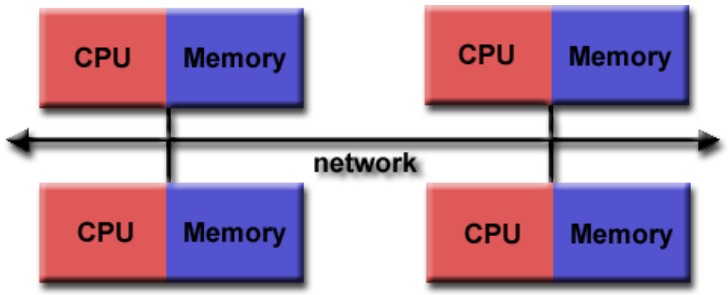
Advantages:

- Global address space provides a user-friendly programming perspective to memory
- Fast and uniform data sharing due to proximity of memory to CPUs

Disadvantages:

- Lack of scalability between memory and CPUs. Adding more CPUs increases traffic on the shared memory-CPU path
- Programmer responsibility for "correct" access to global memory

Parallel Computer Memory Architectures:

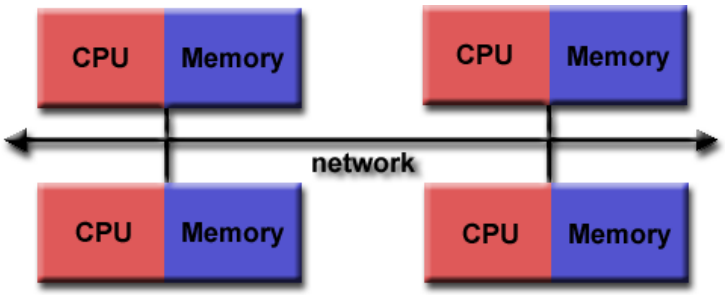


Distributed Memory:

- Requires a communication network to connect inter-processor memory
- Processors have their own local memory. Changes made by one CPU have no effect on others
- Requires communication to exchange data among processors

Parallel Computer Memory Architectures:

Distributed Memory:



- Requires a communication network to connect inter-processor memory
- Processors have their own local memory. Changes made by one CPU have no effect on others
- Requires communication to exchange data among processors

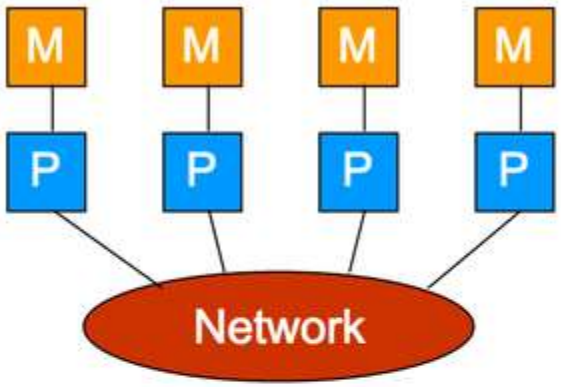
Advantages:

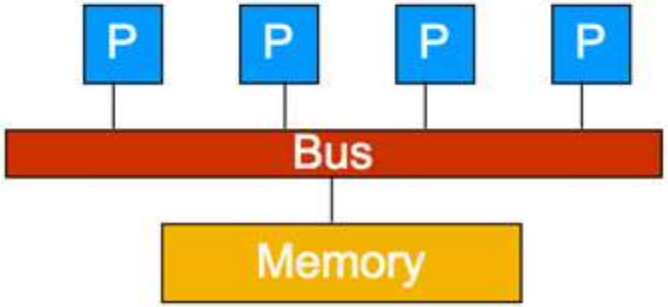
- Memory is scalable with the number of CPUs
- Each CPU can rapidly access its own memory without overhead incurred with trying to maintain global cache coherency

Disadvantages:

- Programmer is responsible for many of the details associated with data communication between processors
- It is usually difficult to map existing data structures to this memory organization, based on global memory

Shared vs Distributed memory



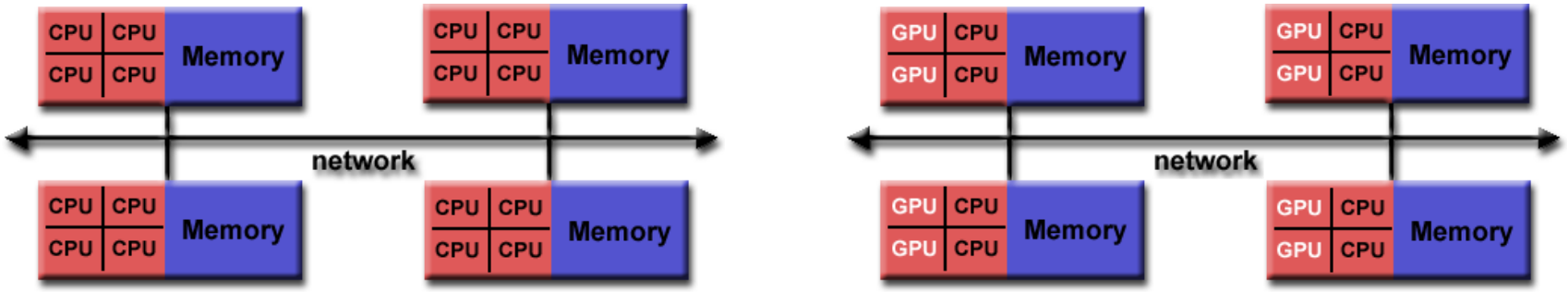


- **Distributed memory**
  - Each processor has its own local memory
  - Message-passing is used to exchange data between processors
- **Shared memory**
  - Single address space
  - All processes have access to the pool of shared memory

Parallel Computer Memory Architectures:

Hybrid Distributed-Shared Memory:

The largest and fastest computers in the world today employ both shared and distributed memory architectures.

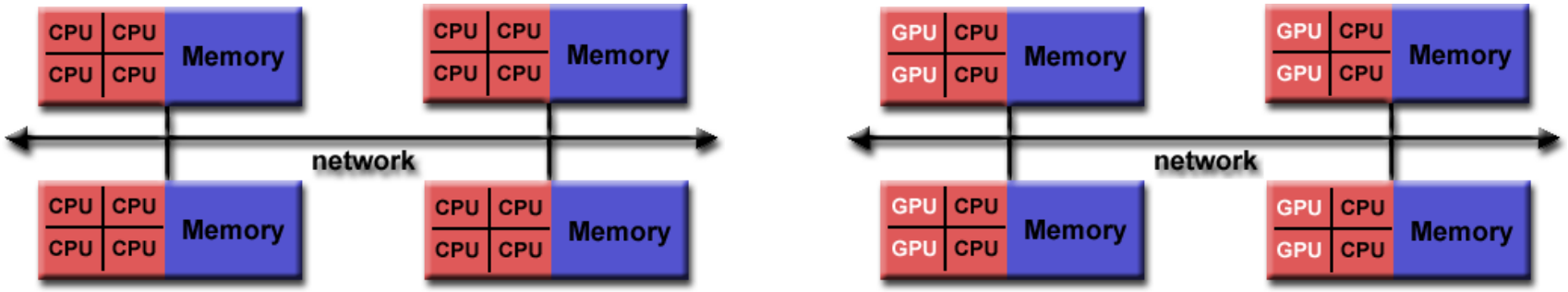


- Shared memory component can be a shared memory machine and/or GPU
- Processors on a compute node share same memory space
- Requires communication to exchange data between compute nodes

Parallel Computer Memory Architectures:

Hybrid Distributed-Shared Memory:

The largest and fastest computers in the world today employ both shared and distributed memory architectures.

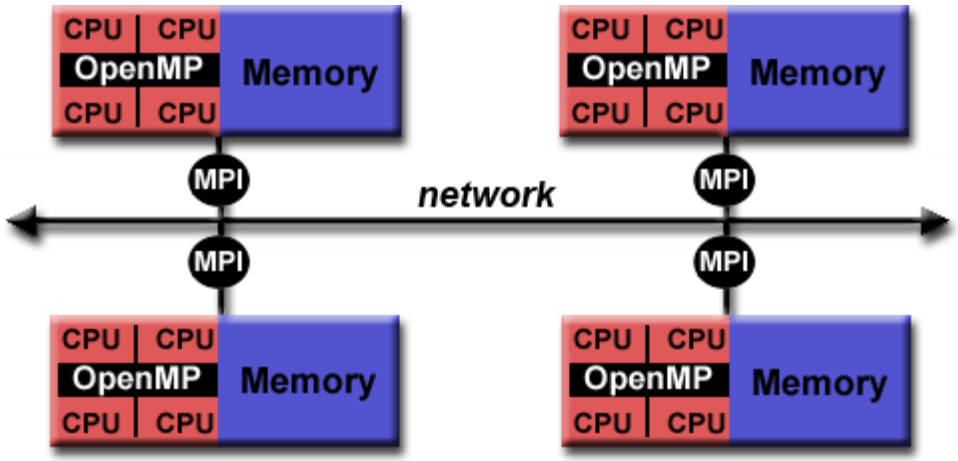


- Shared memory component can be a shared memory machine and/or GPU
- Processors on a compute node share same memory space
- Requires communication to exchange data between compute nodes

Advantages and Disadvantages:

- Whatever is common to both shared and distributed memory architectures
- Increased scalability is an important advantage
- Increased programming complexity is a major disadvantage

Hybrid Parallel Programming Models:

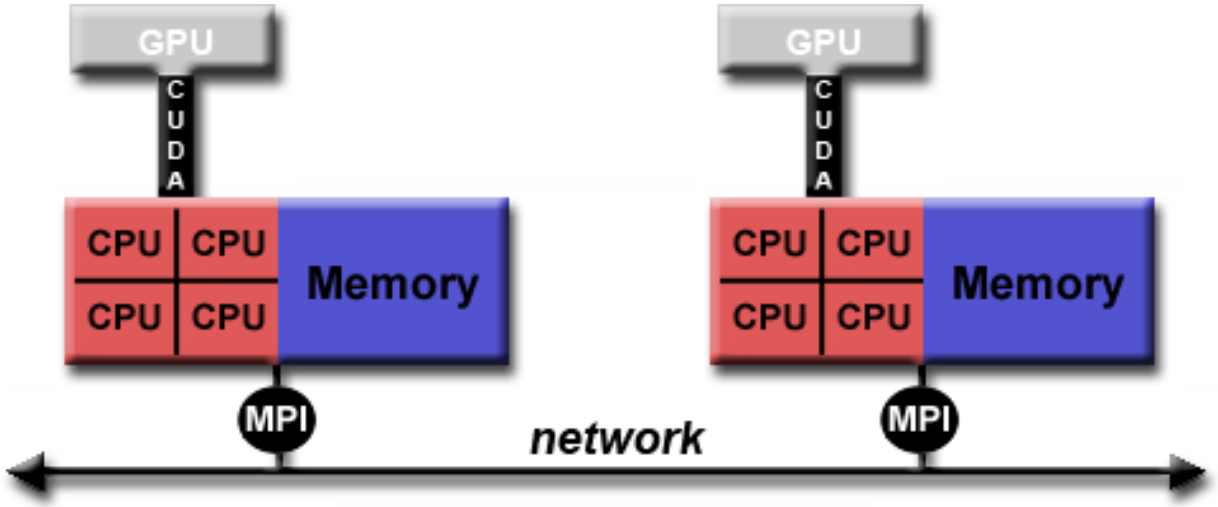


Currently, a common example of a hybrid model is the combination of the message passing model (MPI) with the threads model (OpenMP)

- Threads perform computationally intensive kernels using local, on-node data
- Communications between processes on different nodes occurs over the network using MPI

This hybrid model lends itself well to the increasingly common hardware environment of clustered multi/many-core machines

Hybrid Parallel Programming Models:



Another similar and increasingly popular example of a hybrid model is using MPI with GPU (Graphics Processing Unit) programming

- GPUs perform computationally intensive kernels using local, on-node data
- Communications between processes on different nodes occurs over the network using MPI

Designing parallel programs - communication:

Most parallel applications require tasks to share data with each other.

Cost of communication: Computational resources are used to package and transmit data. Requires frequently synchronization – some tasks will wait instead of doing work. Could saturate network bandwidth.

Latency vs. Bandwidth: Latency is the time it takes to send a minimal message between two tasks. Bandwidth is the amount of data that can be communicated per unit of time. Sending many small messages can cause latency to dominate communication overhead.

Synchronous vs. Asynchronous communication: Synchronous communication is referred to as blocking communication – other work stops until the communication is completed.
Asynchronous communication is referred to as non-blocking since other work can be done while communication is taking place.

Languages using parallel computing:

- C/C++
- Fortran
- MATLAB
- Python
- R
- Perl
- Julia
- And others