Lecture 1 (Architecture)

1 Difference Between Parallel and Distributed

1. Parallel: Focus on doing many things at same time
2. Distributed: How multiple things interact with each other

Concurrency: Unordered computation is synced

2 Threads of Execution

Sequential execution of instructions within a thread, parallel (and distributed) threads are independent

Read: cache lines and false sharing

3 Why Study Parallel and Distributed Computing

1. Clock speed cannot be increased after a certain level, therefore do parallel computation per clock
   - Cannot miniaturise further to reduce the clock speed
2. Need to parallelize memory accesses

4 Structure of Parallel Computer

CPU + CPU or GPU + CPU forms a node

4.1 CPU

1. There are multiple cores
2. There are two levels of individual cache and a level of shared cache
3. A memory and network controller is also present
4. PCIe connections are available to connect to other devices
5. Multi-channel (8 channels) DRAM access is also available

4.2 GPU

1. There are groups of smaller units called *SM*
2. Each *SM* has a group of cores which executes a single instruction at a time called *SIMD* (there are 4 such groups)

4.3 Some Notes on Parallel Execution

1. When executing parallel commands, the states of different cores can be different
2. Some shared states also exist
3. This is what determines status of execution, such as when updating $z = x + y$, action is completed when written to L1 cache for the same core, but not for another core

5 Classes of Parallel Computers - Flynn's Taxonomy

(single - S, multiple - M, instruction - I, data - D) 1. SISD 1. SIMD (example CUDA, i.e., vectorised execution) 1. MISD 1. MIMD

6 Networking Topologies

1. 2D mesh - each nodes is connected to atmost 4 other nodes
2. 3D mesh - connected to atmost 6 nodes now
3. Torus - each is connected to exactly 4 (in 2D) or 6 (in 3D) other nodes (circular connection)
4. Hypercube - Inductively defined:
   - $0D$ hypercube is a single node
   - $(n-1)D$ hypercube is connected to another copy of $(n-1)D$ hypercube such that corresponding elements are connected to each other
5. Tree
6. Fat tree - nigher nodes have more number of connections to reduce bottlenecks
7. Butterfly (or shuffle exchange)
   - each *layer* is connected to the next one and has 2 connections per node on either side (elements of a single layer are disconnected)
   - the connections are such that there is one link to the node immediately next to it and another to the a node which has same modulo 2^i for some i
   - using this connection, data can be sorted by the time it reaches the other end of the network
   - they have switching device to decide between pass through or cross over

6.1 Properties

1. Diameter - distance between the furthest nodes
2. Bisection width - number of edges to be removed to divide the graph into almost two equal halves

3. Blocking - if every node can reach another node irrespective of every other node
4. Switching mechanism - circuit or packet

7 Distributed Memory

1. Each node has its own address space
2. To communicate between nodes via processes the following happens
    - program buffer sends data to OS buffer
    - OS buffer sends it to NIC buffer
    - NIC transfers data using some protocol to the other node's NIC
    - reverse path follows on other node
3. Concept of RDMA (Remote Direct Memory Access) exists as well
4. Virtual memory also comes into picture, will be discussed in detail later