COSC 6374
Parallel Computation

Parallel Computer Architectures

Some slides on network topologies based on a similar presentation by Michael Resch

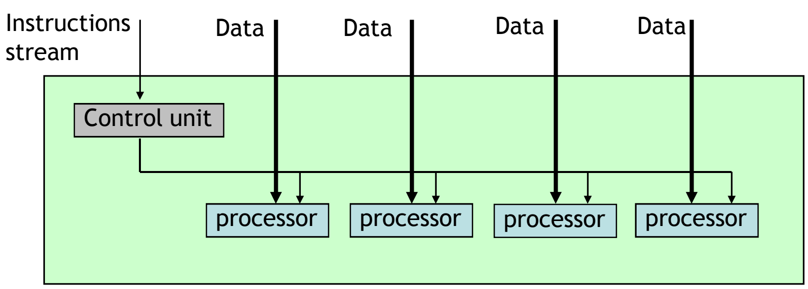
Edgar Gabriel
Spring 2008

Flynn's Taxonomy

- SISD: Single instruction single data
  - Classical von Neumann architecture
- SIMD: Single instruction multiple data
- MISD: Multiple instructions single data
  - Non existent, just listed for completeness
- MIMD: Multiple instructions multiple data
  - Most common and general parallel machine

Single Instruction Multiple Data (I)

- Also known as Array-processors
- A single instruction stream is broadcasted to multiple processors, each having its own data stream



Single Instruction Multiple Data (II)

- Interesting detail: handling of if-conditions
  - First all processors, for which the if-condition is true execute the according code-section, other processors are on hold
  - Second, all processors for the if-condition is not true execute the according code-section, other processors are on hold
- Some architectures in the early 90s used SIMD (MasPar, Thinking Machines)
- No SIMD machines available today
- SIMD concept used in processors of your graphics card

Multiple Instructions Multiple Data (I)

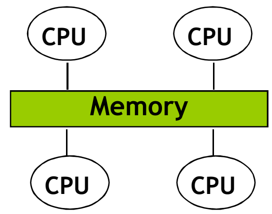
- Each processor has its own instruction stream and input data
- Most general case – every other scenario can be mapped to MIMD
- Further breakdown of MIMD usually based on the memory organization
  - Shared memory systems
  - Distributed memory systems

Shared memory systems (I)

- All processes have access to the same address space
  - E.g. PC with more than one processor
- Data exchange between processes by writing/reading shared variables
  - Shared memory systems are easy to program
  - Current standard in scientific programming: OpenMP
- Two versions of shared memory systems available today
  - Symmetric multiprocessors (SMP)
  - Non-uniform memory access (NUMA) architectures

Symmetric multi-processors (SMPs)

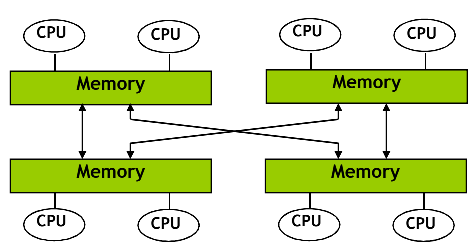
- All processors share the same physical main memory



- Memory bandwidth per processor is limiting factor for this type of architecture
- Typical size: 2-16 processors

NUMA architectures (I)

- Some memory is closer to a certain processor than other memory
  - The whole memory is still addressable from all processors
  - Depending on what data item a processor retrieves, the access time might vary strongly

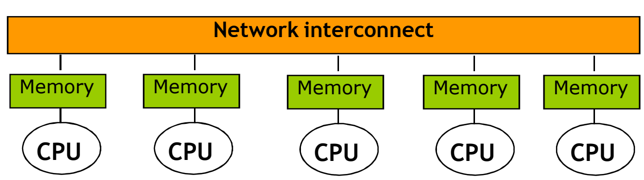


NUMA architectures (II)

- Reduces the memory bottleneck compared to SMPs
- More difficult to program efficiently
  - First touch policy: data item will be located in the memory of the processor which touches the data item first
- To reduce effects of non-uniform memory access, caches are often used
  - ccNUMA: cache-coherent non-uniform memory access architectures
- Largest example as of today: SGI Origin with 512 processors

Distributed memory machines (I)

- Each processor has its own address space
- Communication between processes by explicit data exchange
  - Sockets
  - Message passing
  - Remote procedure call / remote method invocation



Distributed memory machines (II)

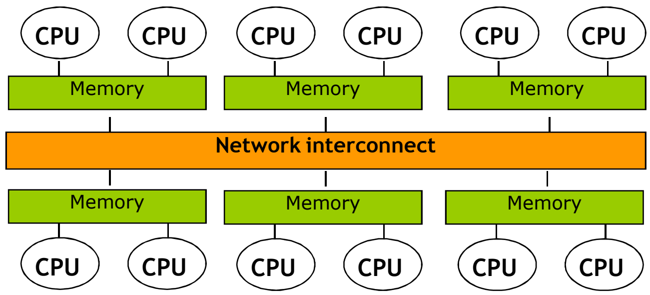
- Performance of a distributed memory machine strongly depends on the quality of the network interconnect and the topology of the network interconnect
  - Of-the-shelf technology: e.g. fast-Ethernet, gigabit-Ethernet
  - Specialized interconnects: Myrinet, Infiniband, Quadrics, …

Distributed memory machines (III)

- Two classes of distributed memory machines:
  - Massively parallel processing systems (MPPs)
    - Tightly coupled environment
    - Single system image (specialized OS)
  - Clusters
    - Of-the-shelf hardware and software components such as
      - Intel P4, AMD Opteron etc.
      - Standard operating systems such as LINUX, Windows, BSD UNIX

Hybrid systems

- E.g. clusters of multi-processor nodes



Grids

- Further evaluation of distributed memory machines
- Several (parallel) machines connected by wide-area links (typically the internet)
  - Machines are in different administrative domains

Network topologies (I)

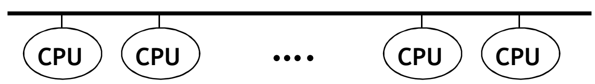
- Important metrics:
  - Latency:
    - minimal time to send a very short message from one processor to another
    - Unit: ms, μs
  - Bandwidth:
    - amount of data which can be transferred from one processor to another in a certain time frame
    - Units: Bytes/sec, KB/s, MB/s, GB/s
Bits/sec, Kb/s, Mb/s, Gb/s,
baud

Network topologies (II)

Metric	Description	Optimal parameter
Link	A direct connection between two processors	
Path	A route between two processors	As many as possible
Distance	Minimum length of a path between two processors	Small
Diameter	Maximum distance in a network	Small
Degree	Number of links that connect to a processor	Small (costs) / Large (redundancy)
Connectivity	Minimum number of links that have to be cut to separate the network	Large (reliability)
Increment	Number of procs to be added to keep the properties of a topology	Small (costs)
Complexity	Number of links required to create a network topology	Small (costs)

Bus-Based Network (I)

- All nodes are connected to the same (shared) communication medium
- Only one communication at a time possible
  - Does not scale



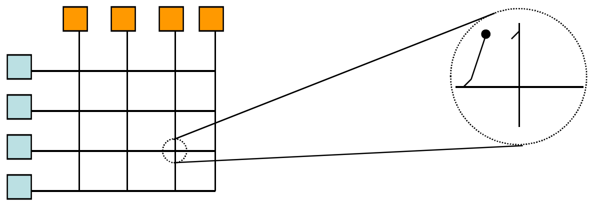
- Examples: Ethernet, SCSI, Token Ring, Memory bus
- Main advantages:
  - simple broadcast
  - cheap

Bus-Based Networks (II)

- Characteristics
  - Distance: 1
  - Diameter: 1
  - Degree: 1
  - Connectivity: 1
  - Increment: 1
  - Complexity:1

Crossbar Networks (I)

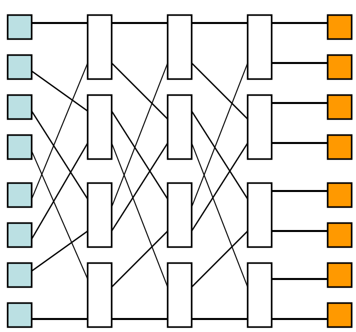
- A grid of switches connecting nxm ports



- a connection from one process to another does not prevent communication between other process pairs
- Scales from the technical perspective
- Does not scale from the financial perspective
- Aggregated Bandwidth of a crossbar: sum of the bandwidth of all possible connections at the same time

Crossbar networks (II)

- Overcoming the financial problem by introducing multi-stage networks



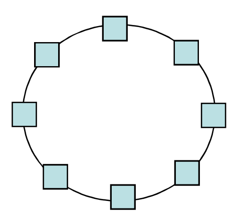
Directly connected networks

- A direct connection between two processors exists
- Network is built from these direct connections
- Relevant topologies
  - Ring
  - Star
  - Fully connected
  - Meshes
  - Toruses
  - Tree based networks
  - Hypercubes

Ring network

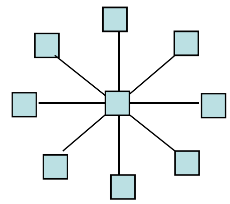
- N: Number of processor connected by the network

- Distance: 1: N/2
- Diameter: N/2
- Degree: 2
- Connectivity: 2
- Increment: 1
- Complexity: N



Star network

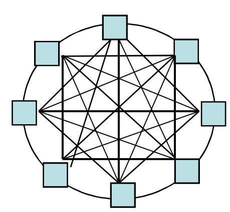
- All communication routed through a central node
  - Central processor is a bottleneck



- Distance: 1 or 2
- Diameter: 2
- Degree: 1 or N
- Connectivity: 1
- Increment: 1
- Complexity: N-1

Fully connected network

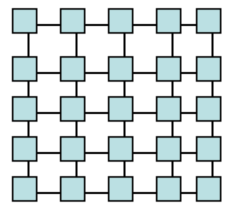
- Every node is connected directly with every other node



- Distance: 1
- Diameter: 1
- Degree: N-1
- Connectivity: N-1
- Increment: 1
- Complexity: N*(N-1)/2

Meshes (I)

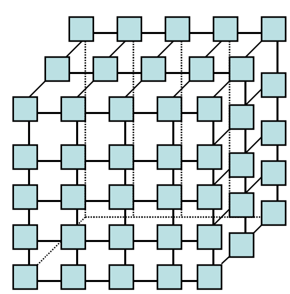
- E.g. 2-D mesh



- Distance: $1 : \sim 2\sqrt{N}$
- Diameter: $\sim 2\sqrt{N}$
- Degree: 2-4
- Connectivity: 2
- Increment: $\sim\sqrt{N}$
- Complexity: ~2N

Meshes (II)

- E.g. 3-D mesh

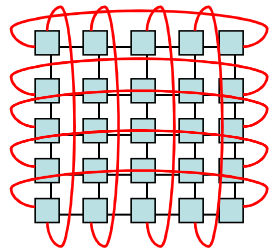


- Distance: $1 : \sim 3\sqrt[3]{N}$
- Diameter: $\sim 3\sqrt[3]{N}$
- Degree: 3-6
- Connectivity: 3
- Increment: $\sim\left(\sqrt[3]{N}\right)^2$
- Complexity: ~

PARALLEL SOFTWARE TECHNOLOGIES LABORATORY PSTL

Toruses (I)

- E.g. 2-D Torus



- Distance: 1:~ $\sqrt{N}$
- Diameter: ~ $\sqrt{N}$
- Degree: 4
- Connectivity: 4
- Increment: ~ $\sqrt{N}$
- Complexity: ~2N

PARALLEL SOFTWARE TECHNOLOGIES LABORATORY PSTL

Toruses (II)

- E.g. 3-D Torus

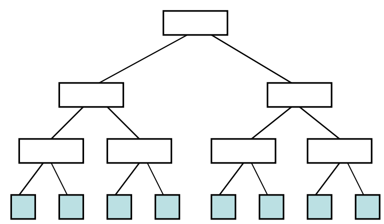
- Distance: 1:~ $\sqrt[3]{N}$
- Diameter: ~ $\sqrt[3]{N}$
- Degree: 6
- Connectivity: 6
- Increment: ~ $\left(\sqrt[3]{N}\right)^2$
- Complexity: ~

Picture not available!

COSC 6374 – Parallel Computation
Edgar Gabriel

Tree based networks (I)

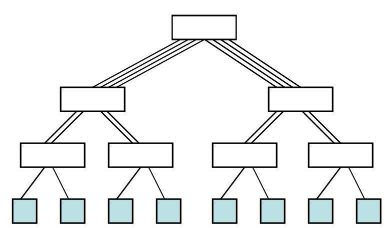
- Most common: binary tree
  - Leafs are computational nodes
  - Intermediate nodes in the tree are switches
  - Higher level switching elements suffer from contention



Tree-based networks (II)

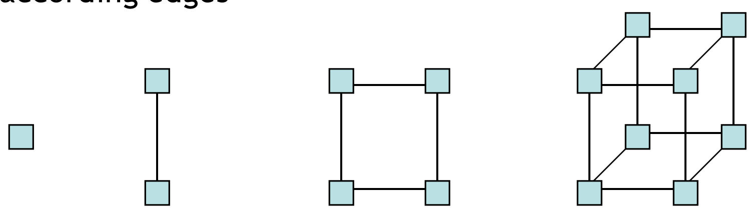
- Fat tree: binary tree which increases the number of communication links between higher level switching elements to avoid contention

- Distance: $1:2\log_2(N)$
- Diameter: $2\log_2(N)$
- Degree: 1
- Connectivity: 1
- Increment: N
- Complexity: ~2N



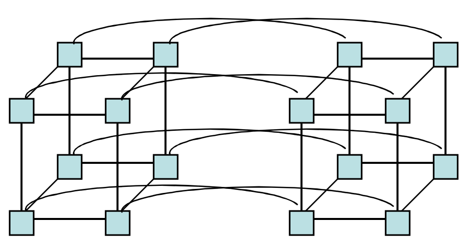
Hypercube (I)

- An n-dimensional hypercube is constructed by doubling two n-1 dimensional hypercubes and connecting the according edges



0-D hypercube   1-D hypercube   2-D hypercube   3-D hypercube

Hypercubes (II)

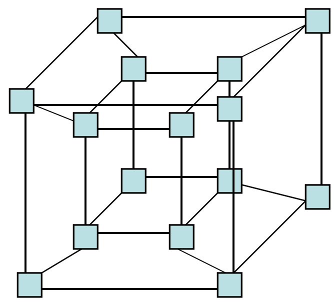


4-D hypercube

Parallel Software Technologies Laboratory PSTL

Hypercubes (III)

- 4-D hypercube also often showed as



- Distance: $1:\log_2(N)$
- Diameter: $\log_2(N)$
- Degree: $\log_2(N)$
- Connectivity: $\log_2(N)$
- Increment: N
- Complexity: $\log_2(N)*N/2$