Parallel Computers

- **Reference:** Chapter 1 of Parallel Programming text by Wilkinson and Allen.
- Need for Parallelism
  - Numerical modeling and simulation of scientific and engineering problems.
  - Solution for problems with deadlines
    - Command & Control problems like ATC.
  - Grand Challenge Problems
    - Sequential solutions could take months or years to run.
- Weather Prediction - Grand Challenge Problem
  - Atmosphere is divided into 3D cells.
  - Data such as temperature, pressure, humidity, wind speed and direction, etc. are recorded at regular time-intervals in each cell.
  - There are about 5×10^8 cells of (1 mile) ^{3}.
  - It would take a modern computer over 100 days to perform necessary calculations for a ten day forecast.
- Parallel Programming - a viable way to increase computational speed.
  - Overall problem is split into parts, each of which are performed by a single processor.

- Ideally, n processors would have n times the computational power of one processor, with each doing 1/nth of the computation.
- Such gains in computational power is rare, due to reasons such as
  - Inability to partition the problem perfectly into n parts of the same computational size.
  - Necessary data transfer between the parts
  - Necessary synchronization between parts.

- Two major styles of partitioning problems
  - (Job) Control parallel programming
    - Problem is divided into the different, nonidentical tasks that have to be performed.
    - The tasks are divided among the processors so that their work load is roughly balanced.
    - This is considered to be *course grained* parallelism.
  - Data parallel programming
    - Each processor performs the same computation on different data sets.
    - Computations may or may not be synchronous.
    - This is considered to be *fine grained* parallelism.

Shared Memory Multiprocessors (SMPs)

- All processors have access to all memory locations .
- The processors access memory through some type of interconnection network.
- This type of memory access is called *uniform memory access* (UMA) .
- A parallel programming language, base on a language like FORTRAN or C/C++ may be available.
- Alternately, programming using *threads* is sometimes used.
- More programming details occur in Chapter 8.
- Difficulty for the SMP architecture to provide fast access to all memory locations lead to hierarchial or distributed memory systems.
  - This type of memory access is called *nonuniform memory access* (NUMA).
- Normally, fast cache is used with NUMA systems to reduce the problem of different memory access time for PEs.
  - This creates the problem of ensuring that all copies of the same date in different memory locations are identical.

(Message-Passing) Multicomputers

- Processors are connected by an interconnection network.
- Each processor has a local memory and can only access its own local memory.
- Data is passed between processors using messages, as dictated by the program.
- **Note:** If the processors run in SIMD mode (i.e., synchronously), then the movement of the data movements can be synchronous:
  - Movement of the data can be controlled by program steps.
  - Much of the message-passing overhead (e.g., routing, hot-spots, headers, etc. can be avoided)
  - Synchronous parallel computers are not usually included in this group of parallel computers.
- A common approach to programming multiprocessors is to use message-passing library library routines in addition to conventional sequential programs (e.g., MPI, PVM)
- The problem is divided into independent *processes* that can be executed concurrently, with each process being executed on a single processor.
- Multicomputers can be scaled to larger sizes much better than shared memory multiprocessors.

Multicomputers (cont.)

- Programming disadvantages of message-passing
  - Programmers must make explicit message-passing calls in the code
  - This is low-level programming and is error prone.
  - Data is not shared but copied, which increases the total data size.
- Programming advantages of message-passing
  - There is no problem with simultaneous access to data.
  - This allows different PCs to operate on the same data independently.
  - Allows PCs on a network to be easily upgraded when faster processors become available.
- Mixed "distributed shared memory" systems.
  - Is a combination of SMPs and multicomputers.
  - Each PC has a local memory and the total local memory is the collection of the local memories.
  - Each memory location has a unique memory address and can be accessed by each PC.
  - Message-passing is used to access "non-local memory" for a PC.
  - Other mixed systems have been developed.

Flynn's Classification Scheme

- SISD - single instruction stream, single data stream
  - Primarily sequential processors
- MIMD - multiple instruction stream, multiple data stream.
  - Includes SMPs and multicomputers
  - processors are asynchronous, since they can independently execute different programs on different data sets.
  - Considered by most researchers to contain the most powerful, least restricted computers.
  - Have some serious message passing (or shared memory) problems that are often ignored when compared to SIMDs (discussed next).
  - May be programmed using a multiple programs, multiple data (MPMD) technique.
  - If the number of processors are large, they are normally programmed using a single program, multiple data (SPMD) technique.
- SIMD - single instruction stream, multiple data stream.
  - One instruction stream is broadcast to all processors.

Flynn's Taxonomy (cont.)

  - Each processor is very simplistic and is essentially an ALU; they do not store the program nor have a program control unit.
  - Individual processors can be inhibited from participating in an instruction (based on a data test).
  - All active processor executes the same instruction synchronously, but on different data (from their own local memory).
  - The data items form an array, and an instruction can act on the complete array in one cycle.

- MISD - Multiple Instruction streams, single data stream.
  - This category is not used very often.
  - Some include pipelined architectures in this category.

Interconnection Network Terminology

- A *link* is the connection between two nodes.
  - A tightly arranged multicomputer with specially designed intefaces is assumed (see fig 1.8)
  - A switch that enables packets to be routed through the node to other nodes without disturbing the processor is assumed.
  - The link between two nodes can be either directional or bidirectional.
  - Either one wire to carry one bit or parallel wires (one wire for each bit in word) can be used.
  - The above choices do not have a major impact on the concepts presented.
- The *bandwidth* is the number of bits that can be transmitted in unit time (i.e., bits per second).
- The *network latency* is the time required to transfer a message through the network.
- The *communication latency* is the total time required to send a message, including sofware overhead and interface delay.
- The *message latency* or *startup time* is the time required to send a zero-length message.
  - Software and hardware overhead, such as
    - finding a route
    - packing and unpacking the message

Network Terminology (cont)

- The *diameter* is the minimal number of links between the two farthest nodes in the network.
  - The diameter of a network gives the maximal distance a single message may have to travel.
- The *bisection width* of a network is the number of links that must be cut to divide the network of n PEs into two (almost) equal parts, $\lceil n/2 \rceil$ and $\lfloor n/2 \rfloor$.

Interconnection Network Examples

- **Completely Connected Network**
  - Each of n nodes has a link to every other node.
  - Requires n(n-1)/2 links
  - Impractical, unless very few processors
- **Line/Ring Network**
  - A *line* consists of a row of n nodes, with connection to adjacent nodes.
  - Called a *ring* when a link is added to connect the two end nodes of a line.
  - The line/ring networks have useful applications (see chapter 5) .

Interconnection Network Examples (cont)

  - Diameter of a line is n-1 and of a ring is $\lfloor n/2 \rfloor$.
  - Routing algorithm: Travel left or right.

- **The Mesh Interconnection Network**
  - Each node in a 2D mesh is connected to all four of its nearest neighbors.
  - The diameter of a $\sqrt{n} \times \sqrt{n}$ mesh is $2(\sqrt{n} - 1)$
  - Has a minimal distance, deadlock-free parallel routing algorithm: First route message up or down and then right or left to its destination.
  - If the horizonal and vertical ends of a mesh to the opposite sides, the network is called a *torus*.
  - Meshes have been used more on actual computers than any other network.
  - A 3D mesh is a generalization of a 2D mesh and has been used in several computers.
  - The fact that 2D and 3D meshes model physical space make them useful for many scientific and engineering problems.

- **Tree Networks**
  - A *binary tree* network is normally assumed to be a complete binary tree.

Interconnection Network Examples (cont)

  - It has a root node, and each interior node has two links connecting it to nodes in the level below it.
  - The height of the tree is $\lfloor \lg n \rfloor$ and its diameter is $2\lfloor \lg n \rfloor$.
  - In an *m-ary tree*, each interior node is connected to *m* nodes on the level below it.
  - The tree is particularly useful for divide-and-conquer algorithms.
  - Unfortunately, the bisection width of a tree is 2 and the communication traffic increases near the root, which can be a bottleneck.
  - In *fat tree* networks, the number of links is increased as the links get closer to the root.
  - The Thinking Machines CM5 computer used a 4-ary fat tree network.

- **Hypercube Network**
  - A 0-dimensional hypercube consists of one node.
  - Recursively, a d-dimensional hypercube consists of two (d-1) dimensional hypercubes, with the corresponding nodes of the d-1 hypercubes linked.

Hypercube Networks

  - Each node in a d-dimensional hypercube has d links.
  - Each node in a hypercube has a d-bit binary address.
  - Two nodes are connected if and only if their binary address differs by one bit.
  - A hypercube has $n = 2^d$ PEs
  - Advantages of the hypercube include
    - its low diameter of *lg(n)* or *d*
    - its large bisection width of *n/2*
    - its regular structure.
  - An important practical disadvantage of the hypercube is that the number of links per node increases as the number of processors increase.
    - Large hypercubes are difficult to implement.
    - Usually overcome by increasing nodes by replacing each node with a ring of nodes.
  - Has a “minimal distance, deadlock-free parallel routing” algorithm called *e-cube routing*:
    - At each step, the current address and the destination address are compared.
    - Routed message to the node whose address is obtained by flipping the leftmost digit of current address where two addresses differ.