Introduction

- Class is an introduction to parallel computing
  - topics include: hardware, applications, compilers, system software, and tools
- Will count for Masters/PhD Comp Credit
- Work required
  - small programming assignments (two)
  - midterm
  - classroom participation
  - project
- Photos will taken of the class to help me learn names
- Will be asked to participate in a study of parallel programming
What is Parallel Computing?

- **Does it include:**
  - super-scalar processing (more than one instruction at once)?
  - client/server computing?
    - what if RPC calls are non-blocking?
  - vector processing (same instruction to several values)?
  - collection of PC’s **not** connected to a network?

- **For this class, parallel computing requires:**
  - more than one processing element
  - nodes connected to a communication network
  - nodes working together to solve a single problem.
Why Parallelism

- **Speed**
  - need to get results faster than possible with sequential
    - a weather forecast that is late is useless
  - could come from
    - more processing elements (P.E.)
    - more memory (or cache)
    - more disks

- **Cost: cheaper to buy many smaller machines**
  - this is only recently true due to
    - VLSI
    - commodity parts
What Does a Parallel Computer Look Like?

- **Hardware**
  - processors
  - communication
  - memory
  - coordination

- **Software**
  - programming model
  - communication libraries
  - operating system
Processing Elements (PE)

- **Key Processor Choices**
  - How many?
  - How powerful?
  - Custom or off-the-shelf?

- **Major Styles of Parallel Computing**
  - SIMD - Single Instruction Multiple Data
    - one master program counter (PC)
  - MIMD - Multiple Instruction Multiple Data
    - separate code for each processor
  - SPMD - Single Program Multiple Data
    - same code on each processor, separate PC’s on each
  - Dataflow - instruction waits for operands
    - “automatically” finds parallelism
MIMD

Processors

Program Counter

Program #1

Program Counter

Program #2

Program Counter

Program #3
SPMD

Processors

Program Counter

Program

Program Counter

Program

Program Counter

Program

Program
Communication Networks

- **Connect**
  - PE’s, memory, I/O

- **Key Performance Issues**
  - latency: time for first byte
  - throughput: average bytes/second

- **Possible Topologies**
  - bus - simple, but doesn’t scale
  - ring - orders delivery of messages
Topologies (cont)

- tree - needs to increase bandwidth near the top

- mesh - two or three dimensions

- hypercube - needs a power of number of nodes
Memory Systems

● Key Performance Issues
  – latency: time for first byte
  – throughput: average bytes/second

● Design Issues
  – Where is the memory
    • divided among each node
    • centrally located (on communication network)
  – Access by processors
    • can all processors get to all memory?
    • is the access time uniform?