Parallel Programming Systems and Models

Ioan Raicu
Center for Ultra-scale Computing and Information Security
Department of Electrical Engineering & Computer Science
Northwestern University

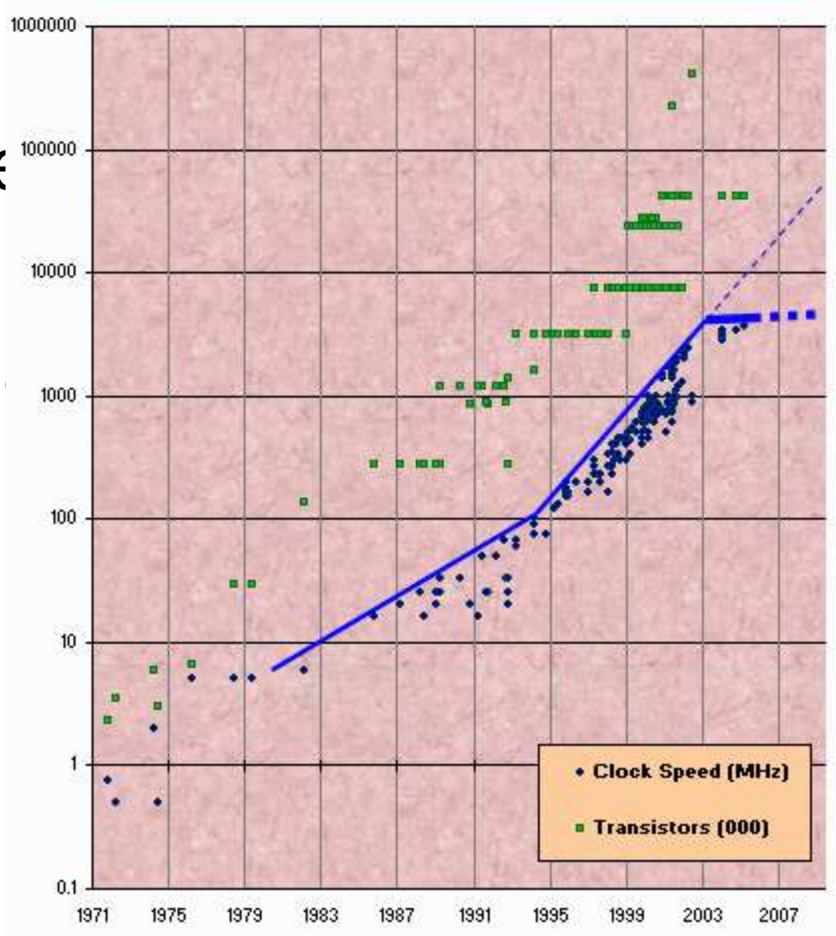
EECS 395 / EECS 495
Hot Topics in Distributed Systems: Data-Intensive Computing
February 9th, 2010

Introduction to Parallel Computing

- Moore's Law
  - The number of transistors that can be placed inexpensively on an integrated circuit will double approximately every 18 months.
  - Self-fulfilling prophecy
    - Computer architect goal
    - Software developer assumption

Introduction to Parallel Computing

- Impediments to Moore's Law
  - Theoretical Limit
  - What to do with all that die
  - Design complexity
  - How do you meet the exp
    increase?

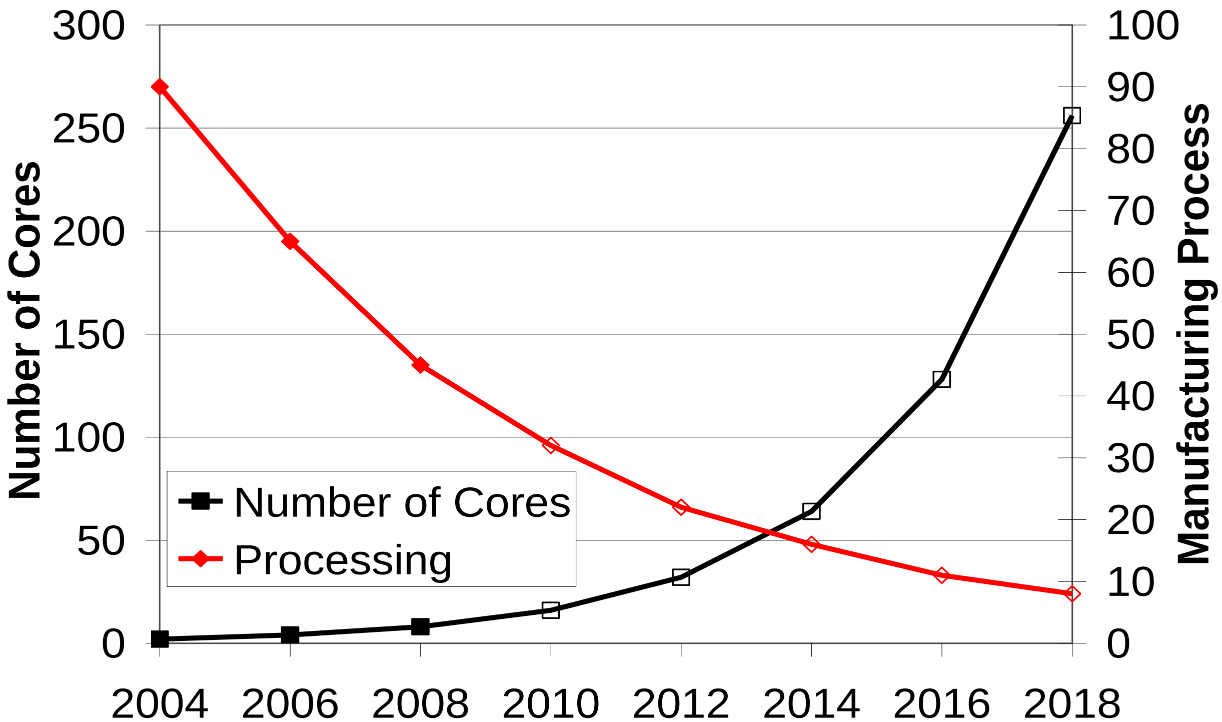


Introduction to Parallel Computing

- von Neumann model
  - Execute a stream of instructions (machine code)
  - Instructions can specify
    - Arithmetic operations
    - Data addresses
    - Next instruction to execute
  - Complexity
    - Track billions of data locations and millions of instructions
    - Manage with:
      - Modular design
      - High-level programming languages

Introduction to Parallel Computing

- Parallelism
  - Continue to increase performance via parallelism.

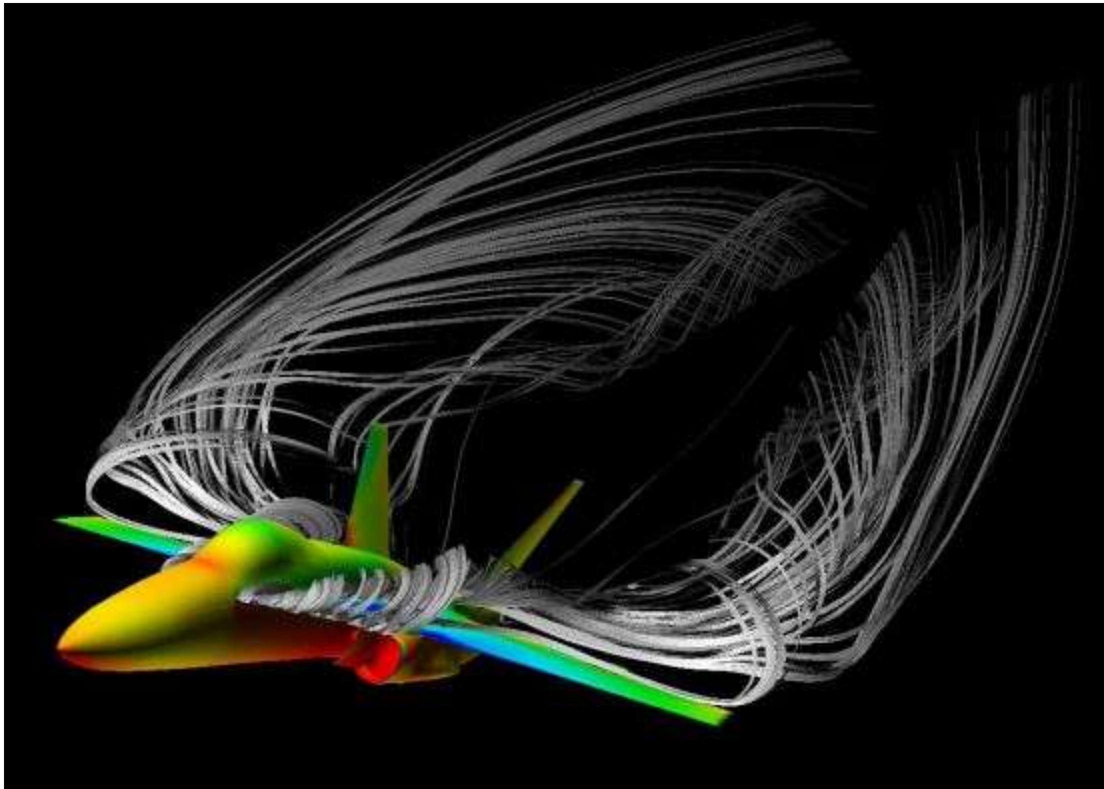


Introduction to Parallel Computing

- From a software point-of-view, need to solve demanding problems
  - Engineering Simulations
  - Scientific Applications
  - Commercial Applications
- Need the performance, resource gains afforded by parallelism

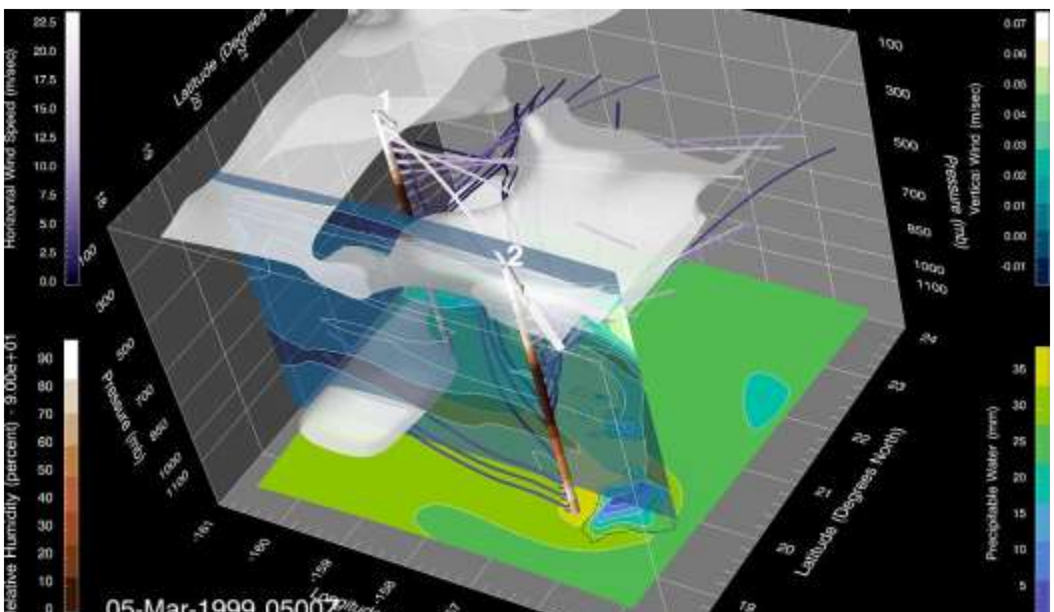
Introduction to Parallel Computing

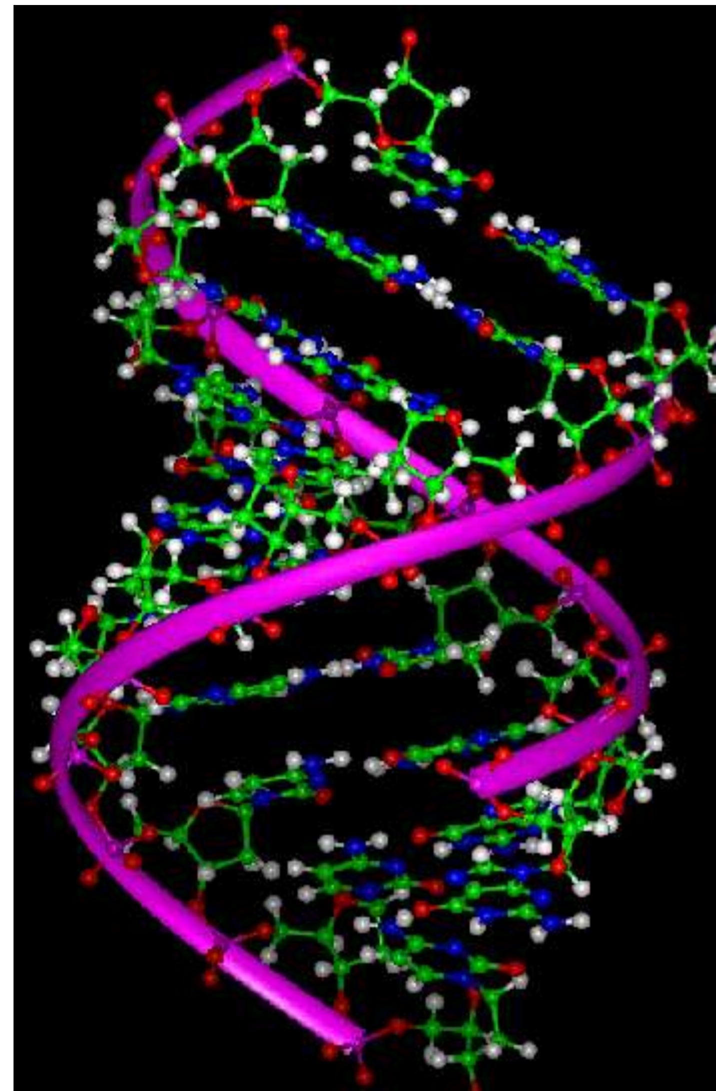
- Engineering Simulations
  - Aerodynamics
  - Engine efficiency



Introduction to Parallel Computing

- Scientific Applications
  - Bioinformatics
  - Thermonuclear processes
  - Weather modeling





Introduction to Parallel Computing

- Commercial Applications
  - Financial transaction processing
  - Data mining
  - Web Indexing

Introduction to Parallel Computing

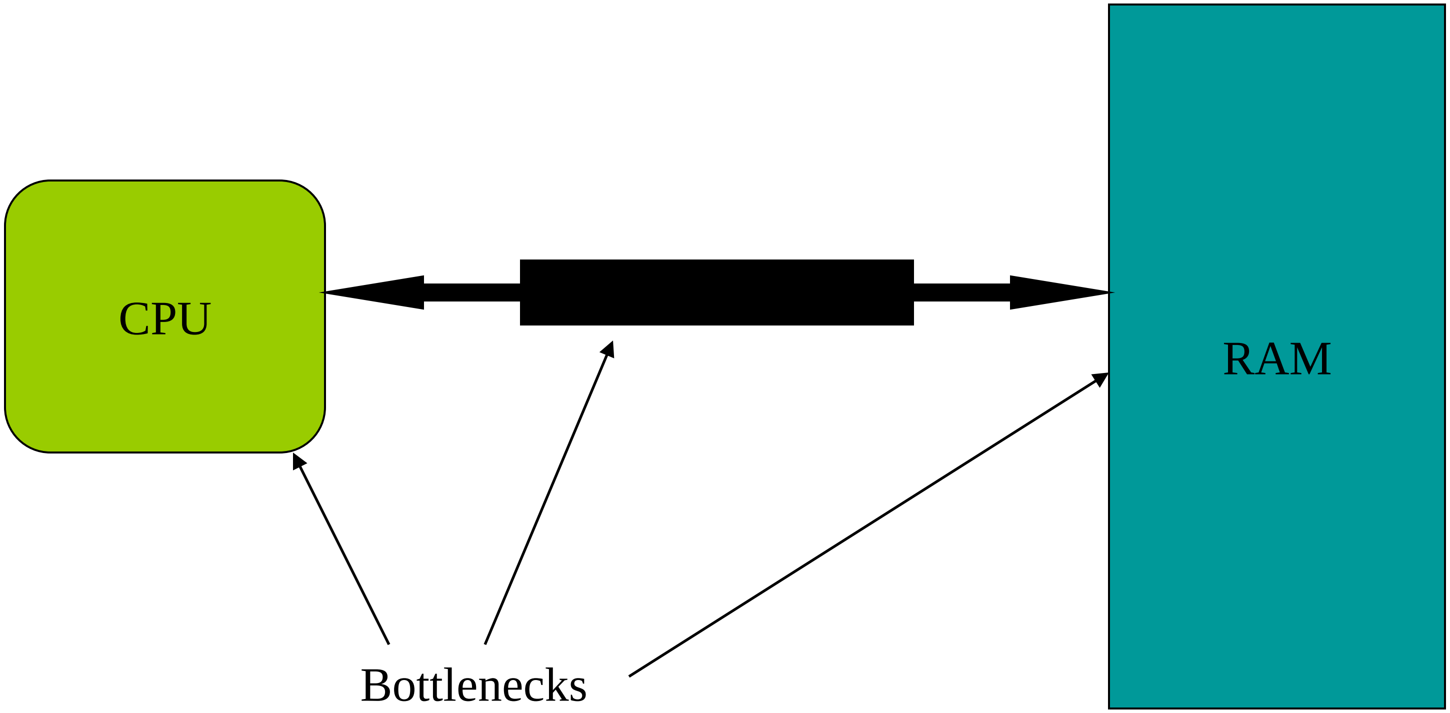
- Unfortunately, greatly increases coding complexity
  - Coordinating concurrent tasks
  - Parallelizing algorithms
  - Lack of standard environments and support

Introduction to Parallel Computing

- The challenge
  - Provide the abstractions, programming paradigms, and algorithms needed to effectively design, implement, and maintain applications that exploit the parallelism provided by the underlying hardware in order to solve modern problems.

Parallel Architectures

- Standard sequential architecture

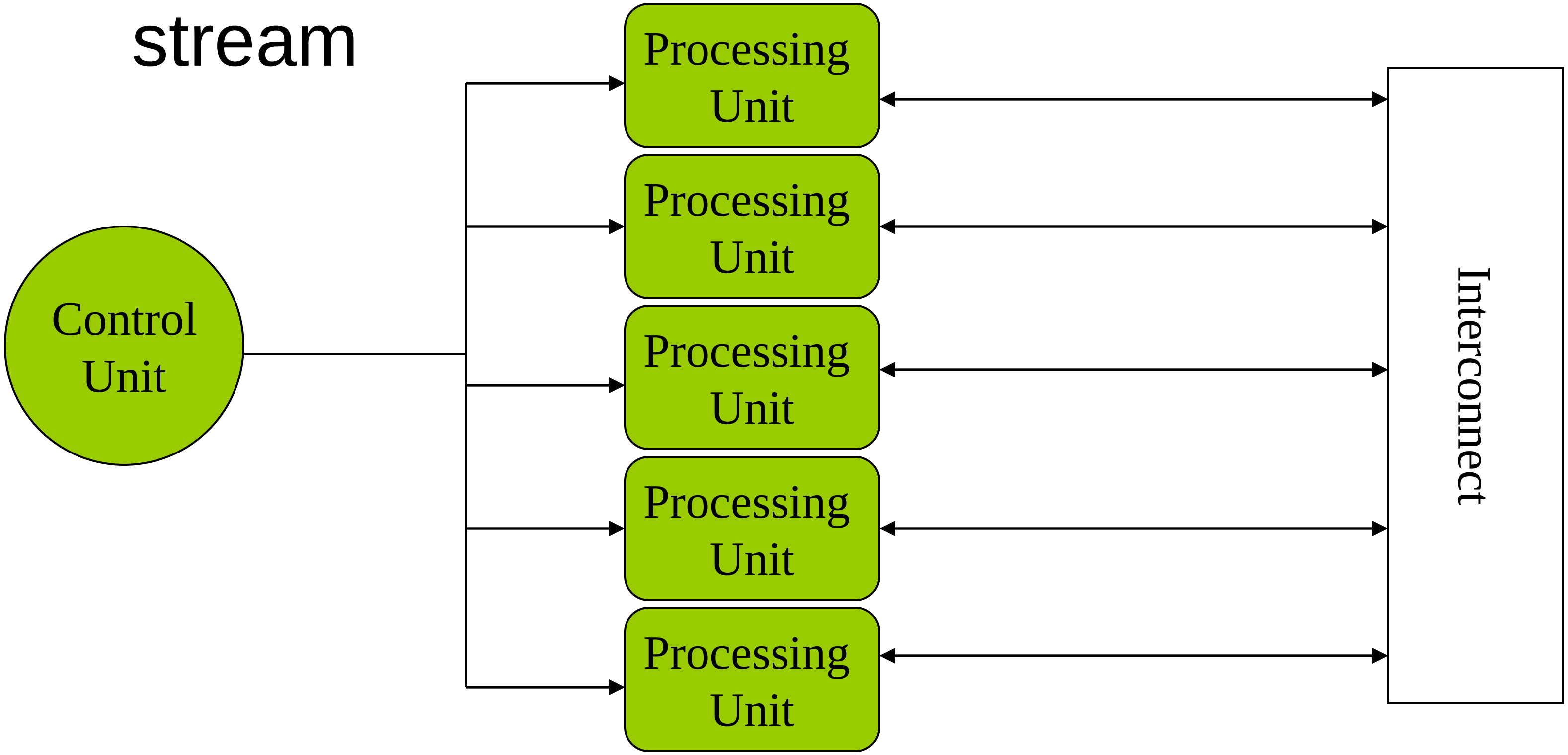


Parallel Architectures

- Use multiple
  - Datapaths
  - Memory units
  - Processing units

Parallel Architectures

- SIMD
  - Single instruction stream, multiple data stream

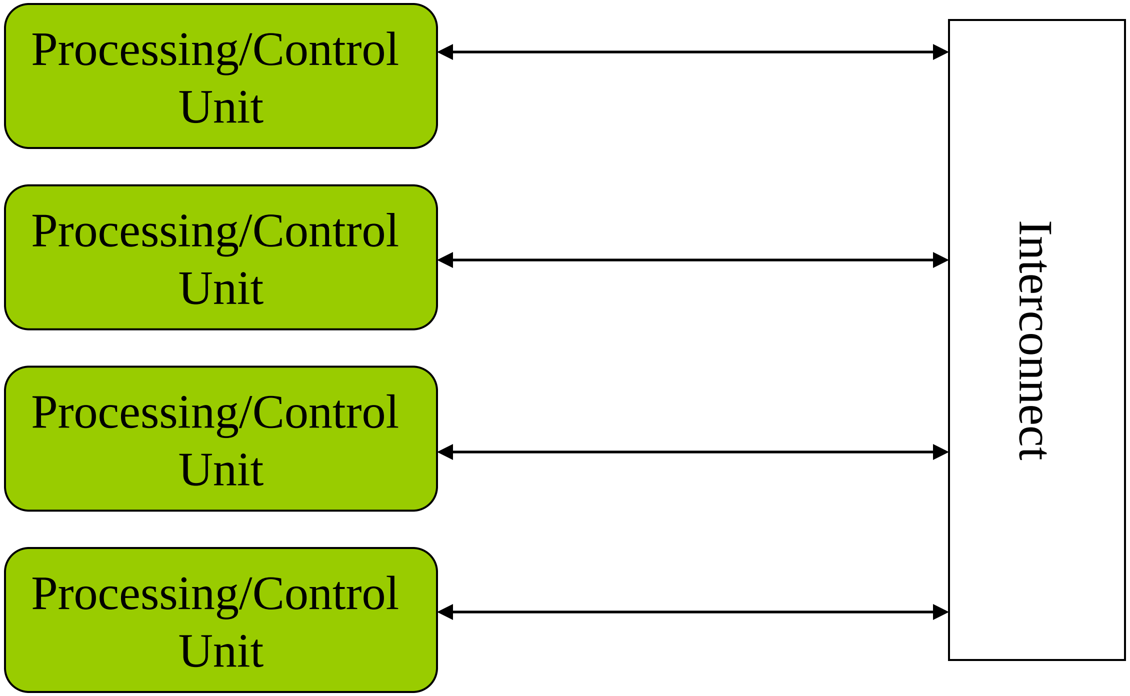


Parallel Architectures

- SIMD
  - Advantages
    - Performs vector/matrix operations well
      - EX: Intel’s MMX chip
  - Disadvantages
    - Too dependent on type of computation
      - EX: Graphics
    - Performance/resource utilization suffers if computations aren’t “embarrasingly parallel”.

Parallel Architectures

- MIMD
  - Multiple instruction stream, multiple data stream



Parallel Architectures

- MIMD
  - Advantages
    - Can be built with off-the-shelf components
    - Better suited to irregular data access patterns
  - Disadvantages
    - Requires more hardware (!sharing control unit)
    - Store program/OS at each processor

- Ex: Typical commodity SMP machines we see today.

Parallel Architectures

- Task Communication
  - Shared address space
    - Use common memory to exchange data
    - Communication and replication are implicit
  - Message passing
    - Use send()/receive() primitives to exchange data
    - Communication and replication are explicit

Parallel Architectures

- Shared address space
  - Uniform memory access (UMA)
    - Access to a memory location is independent of which processing unit makes the request.
  - Non-uniform memory access (NUMA)
    - Access to a memory location depends on the location of the processing unit relative to the memory accessed.

Parallel Architectures

- Message passing
  - Each processing unit has its own private memory
  - Exchange of messages used to pass data
  - APIs
    - Message Passing Interface (MPI)
    - Parallel Virtual Machine (PVM)

Parallel Algorithms

- Algorithm
  - a sequence of finite instructions, often used for calculation and data processing.
- Parallel Algorithm
  - An algorithm that which can be executed a piece at a time on many different processing devices, and then put back together again at the end to get the correct result

Parallel Algorithms

- Challenges
  - Identifying work that can be done concurrently.
  - Mapping work to processing units.
  - Distributing the work
  - Managing access to shared data
  - Synchronizing various stages of execution.

Questions

?