

Parallel and Distributed Systems

Instructor: Sandhya Dwarkadas
Department of Computer Science
University of Rochester

- What is a parallel computer?
 - “A collection of processing elements that communicate and cooperate to solve large problems fast”
- What is a distributed system?
 - “A collection of independent computers that appear to its users as a single coherent system”

Why Parallel or Distributed Computing?

- Fundamentally limited by the speed of a sequential processor
- Resource sharing
- Information exchange
- Collaboration

Hardware Trends

Moore's law (attributed to Gordon Moore, Founder of Intel):
Circuit complexity double every 18 months (revised 1975)

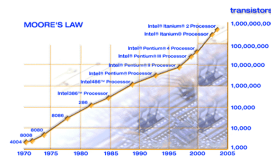


Image from http://www.intel.com/technology/silicon/mooreslaw/pix/mooreslaw_chart.gif

Leveraging Moore's Law

- More transistors – opportunities for exploiting parallelism
 - Implicit parallelism
 - Pipelining
 - Superscalar
 - Explicit parallelism
 - Streaming and multimedia processor extensions
 - E.g., MMX, AltiVec
 - Very long instruction words (VLIW)

Current Trends

- Problems:
 - Fundamental circuit delay and heat limitations
 - Limited amount of instruction-level parallelism
- Solutions: proliferation of (from Sun, IBM, Intel)
 - Multithreading
 - Multicore
 - Multiprocessors

Top 500 List of Supercomputers (www.top500.org)

- Top 5 from the list
 - BlueGene/L – 131072 processors
 - Red Storm (2.4 GHz dual core opterons from Cray) – 26544 processors
 - eServer BlueGene Solution – 40960 processors
 - ASC Purple – eServer pSeries from IBM – 12208
 - IBM Myrinet cluster in Barcelona – 10240 processors

Parallel System Issues

- Data sharing – single versus multiple address spaces
- Process coordination – synchronization using locks, messages, ...
- Distributed (non-uniform access – NUMA) versus centralized (uniform access – UMA) memory
- Connectivity – single shared bus versus network with many different topologies
- Fault tolerance/reliability

Distributed Systems Issues

- Transparency
 - Access (data representation), location, migration, relocation, replication, concurrency, failure, persistence
- Scalability
 - Size, geographically, administratively
- Reliability/fault tolerance

Parallel Programming Models

- Data parallel – HPF, Fortran-D, Power C/Fortran
- Shared memory - pthreads
- Message passing – MPI, PVM
- Global address space

Types of Parallelism

- Data
- Functional (task)

Why is Parallel Computing Hard?

- Amdahl's law – insufficient available parallelism
 - $\text{Speedup} = 1 / (\text{fraction_enhanced} / \text{speedup} + (1 - \text{fraction_enhanced}))$
- Overhead of communication and coordination
- Portability – knowledge of underlying architecture often required

Steps in the Parallelization Process

- Decomposition into tasks
- Assignment to processes
- Orchestration – communication of data, synchronization among processes

Types of Dependences

- Flow (or True) dependence – RAW
- Anti-dependence – WAR
- Output dependence – WAW

Course Recap

- Parallel programming models – shared memory and message passing
- Process coordination – synchronization and data consistency
- Scalability
- Fault tolerance and reliability

Basics of Parallelization

- Dependence analysis
- Synchronization
 - Events
 - Mutual exclusion
- Parallelism patterns