Parallel computer architectures

- Characteristics of interconnection architecture
- Methods for interprocessor communication, synchronization

Why Parallel Architectures

**Basic Motivation: Performance**

- Only way to significantly increase speed
  - Orders of magnitude higher performance beyond uniprocessor
- More cost effective than custom uniprocessor
  - Processor design / fabrication cost >> $100 million
  - Commercial microprocessor costs can be amortized
- Single chip multiprocessors
  - Architects running out of uses for more transistors
  - Except for larger on-chip caches
  - Parallelism becomes effective use of chip density
Technology Trends

Commodity microprocessors caught up to and killed off most custom processors in 1990's

Parallel Architectures Today

- **Common parallel architectures**
  - Desktop multiprocessor
    - Small number (2-4) of processors in single PC
  - Multiprocessor servers (SMP – Symmetric Multi-Processor)
    - Medium number (4-64) of processors in single system
    - HP SuperDome, Sun SunFire, IBM Regatta
  - Clusters of Workstations (COW)
    - Large collections (clusters) of processors / SMPS
    - HP AlphaServer, Beowulf Linux PCs
- **Supercomputer architectures (less common)**
  - Distributed memory multiprocessors (MPP)
  - Constellations – clusters of custom vector processors
Parallel Architecture Overview

- **Motivation**
- **Forms of parallelism**
  - Pipeline, SIMD, MIMD
- **Types of parallel architectures**
  - Shared memory multiprocessor (SMP)
  - Distributed memory multiprocessors (MPP)
  - Cluster
  - Constellation
- **Summary**

Definition of Parallel Computer

- A collection of processing elements that can communicate and cooperate to solve large problems fast
A collection of processing elements …

✦ How many
   - A few, dozens, hundreds, thousands, many thousands or more

✦ How powerful is each
   - 1-bit processors, microprocessor, vector processors

✦ How much memory
   - Kilobytes to gigabytes

… that can communicate and coordinate …

✦ How do they exchange data
   - Shared memory vs. message passing

✦ What is the address space of programs
   - Global vs. local address space

✦ How are they interconnected
   - Bus vs. switching network vs. loosely coupled network

✦ How do they coordinate (synchronize) their execution
   - Shared memory: locks, barriers, monitors
   - Message passing: blocking msgs. / collective communications

✦ How frequently do they coordinate their execution
   - Granularity: refers to the average size of subtasks separated by synchronization points, measured in instructions executed
… to solve large problems fast

- **How large**
  - Sufficiently beyond capability of single processor to make parallelism worthwhile

- **How fast is fast**
  - Fast enough for solution to be of interest

**Forms of Parallelism**

- **Independence among computations**

  - **Parallelism**
    - **Synchronous**
      - **Pipelined**
      - **SIMD**
    - **Asynchronous**
      - **MIMD**

  - **Vector**
A Banking Analogy

Pipeline Parallelism

- Tellers work as an assembly line of workers
SIMD Parallelism

- **Single Instruction Multiple Data (SIMD)**
  - All tellers do the same thing (or remain idle) at the same time

MIMD Parallelism

- **Multiple Instruction Multiple Data (MIMD)**
  - Tellers independently serve different customers at the same time
Pipeline Parallelism

- **Sources of pipeline parallelism**
  - A number of tasks that may be partially overlapped
  - Tasks with multiple segments that may be overlapped

- **Characteristics**
  - Suitable for both fine-grain (instructions / vector) and coarse-grain (task) parallelism
  - Limited scalability

Instruction-Level Pipeline Parallelism

<table>
<thead>
<tr>
<th>Instruction fetch</th>
<th>Instruction decode</th>
<th>Execution</th>
<th>Memory Access</th>
<th>Write back</th>
<th>Instruction fetch</th>
<th>Instruction decode</th>
<th>Execution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SIMD Parallelism

- **Sources of SIMD parallelism**
  - Identical operation executed on multiple data in parallel

- **Characteristics**
  - Single instruction stream for all processors
  - Limited flexibility
  - Suitable for vector, data-parallel computations

MIMD Parallelism

- **Sources of MIMD parallelism**
  - Any parallel computation

- **Characteristics**
  - Different instruction stream for each processor
  - Very flexible
  - Suitable for any (coarse-grain) parallel computation
    - Usually higher synchronization / communication costs

- **Types of MIMD parallelism**
  - MPMD – multiple program multiple data
  - SPMD – single program multiple data
**MPMD (Multiple Program Multiple Data)**

- Each processor has a **different** program to execute
  - Greatest flexibility, complexity

![Diagram of MPMD](image)

**SPMD (Single Program Multiple Data)**

- Each processor has a **same** program to execute
  - Can execute different instructions due to control flow
  - Moderate flexibility, lower complexity
  - Easier to program than MPMD in practice

![Diagram of SPMD](image)
Granularity of Parallelism

- **Granularity**
  - Refers to amount of work that can be executed independently
  - Coarse-grain parallelism → infrequent synchronization
  - Fine-grain parallelism → frequent synchronization

- **Categories**
  - Instruction level
    - \( A = 1, B = 2 \)
  - Vector
  - Loop level
    - Forall \( I = 1,100 \) { … }
  - Task level
    - Parallel case A: { … } B: { … }

Parallel Architecture Overview

- **Motivation**
- **Forms of parallelism**
  - Pipeline, SIMD, MIMD
- **Types of parallel architectures**
  - Shared memory multiprocessor (SMP)
  - Distributed memory multiprocessors (MPP)
  - Cluster
  - Constellation
- **Summary**
Shared-Memory Multiprocessor (SMP)

- **Multiple processors sharing single memory**
  - HP SuperDome, Sun SunFire, IBM Regatta

- **Types**
  1. **UMA (Uniform Memory Access)**
     - Physically shared memory, shared bus
     - Also known as **SMP (Symmetric Multi Processor)**
  2. **NUMA (Non-Uniform Memory Access)**
     - Physically distributed memory, shared interconnect
     - Coherent cache (SGI Origin 2000)
       - Snoops the system bus to maintain cache coherence
     - Non-coherent cache (Cray T3E)
  3. **COMA (Cache Only Memory Architecture)**
     - NUMA that treats all memory as cache

Models of Shared-Memory Multiprocessors

- **Uniform Memory Access (UMA) Model**
  - Interconnect: Bus, Crossbar, Multistage network
  - P: Processor
  - M: Memory
  - C: Cache
  - D: Cache directory

- **Distributed memory (NUMA) Model**

- **Cache-Only Memory Architecture (COMA)**
SMP – Intel Pentium Pro Quad

Cache Coherence

<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
<th>Cache for A</th>
<th>Cache for B</th>
<th>Memory for X</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>CPU A reads X</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>CPU B reads X</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>CPU A writes 0 to X</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

- Caches are coherent if all processors have same value
- Bus snooping
  - All processors listen on shared bus for updates to local cache
- Directory-based
  - Processors maintain directory of all processors with data
  - Send messages to invalidate or update remote cache entries
**Cache Coherent NUMA – SGI Origin 2000**

- Two processors on each node, up to 512 processors
- Hub implements the directory-based cache coherence protocol
- Scalable interconnection network

**Non-Cache Coherent NUMA – Cray T3E**

- Scale up to 1024 processors, 480MB/s links
- Memory controller generates communication requests for non-local references
- No hardware mechanism for coherence
Shared Memory Multiprocessor (SMP)

- **Advantages**
  - Processor can directly reference any memory location
    - Communication occurs *implicitly* as loads and stores
  - Convenient:
    - Simple programming model

- **Disadvantages**
  - May introduce (data race) errors dependent on execution order
  - SMP is not scalable – contention for shared interconnect

Distributed-Memory Multiprocessor (MPP)

- **Large collection of tightly connected processor nodes**
  - Intel Paragon, IBM SP-2
- **Also known as “message-passing” computers**

- **Characteristics**
  - Distributed memory and separate address spaces
  - Non-local memory accesses expensive
  - Large memory & high scalability
Distributed-Memory Multiprocessor (MPP)

- **Architecture**
  - Comprised of multiple autonomous computers (nodes)
  - Each node consists of a processor, local memory, attached storage and I/O peripherals (approx single PC or workstation)
  - Local memory is only accessible by local processors
  - Inter-node communication is carried out by sending messages through the interconnection network

- **Programming**
  - Explicit messages for interprocessor communication

MPP – Intel Paragon
MPP – IBM SP-2

- Made out of essentially complete IBM RS6000 workstations
- Network interface integrated in I/O bus (bandwidth limited by I/O bus)

Cluster Architecture

- Large collections of loosely connected PC / SMP nodes
  - Beowulf cluster (Linux PC), HP AlphaServer SC (Alpha)
  - High-performance interconnects
    - Couple hundred of megabytes/s of bandwidth (300-400) and latencies in the order of a few microseconds (2-5)
    - Quadrics QsNet, Myrinet
  - Low-performance interconnects
    - Tens of megabytes/s of bandwidth and latencies in the order of tens of microseconds (20-50)
    - Gigabit Ethernet

- Characteristics
  - Large memory & very high scalability
  - Hybrid memory model if SMP nodes
  - Communicate using message passing
Cluster Architecture – HP AlphaServer SC

- 4-processor 1.2 GHz EV68 Alpha nodes
- 2-32 GB memory / node
- 280 MB/sec AlphaServer SC PCI adaptor
- 32 GB/sec cross-section bandwidth
  - The amount of information that all the nodes can pass to one another and back again, all at one time, and all in one second

Constellation Architecture

- Small / medium collections of fast vector nodes
  - NEC Earth Simulator, Cray X1
    - 4-8 vector processor nodes
  - IBM BlueGene
    - 64-processor SMP nodes

- Characteristics
  - Large memory & moderate scalability
  - High performance nodes
  - Hybrid memory model
Constellation – Vector Nodes

- **Custom vector processors**
  - NEC SX-6, Cray SV-1

- **Characteristics**
  - Vector operations on vector registers
  - High-bandwidth pipelined memory access
    - Memory bandwidth matches processing rate
    - No cache coherence needed for vector operations
    - Achieves high percentage of peak performance
  - Global shared memory (PVP)
    - Very limited scalability in processor count
  - Easy, but different, programming model
  - Expensive

Constellation – NEC Earth Simulator

<table>
<thead>
<tr>
<th>Number of Processors/Node</th>
<th>8</th>
<th>Peak Performance/Processor</th>
<th>8 Gflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Number of Processors</td>
<td>5120</td>
<td>Peak Performance/Node</td>
<td>64 Gflops</td>
</tr>
<tr>
<td>Total Number of Nodes</td>
<td>640</td>
<td>Total Peak Performance</td>
<td>40 Tflops</td>
</tr>
<tr>
<td>Shared Memory/Node</td>
<td>16 GB</td>
<td>Total Main Memory</td>
<td>10 TB</td>
</tr>
</tbody>
</table>
Constellation – NEC Earth Simulator

- Three levels of parallelism
  - Vector processors
  - Shared memory
  - Message passing
- Currently world’s most powerful computer
  - As fast as next 5 fastest computers combined

Constellation – Cray X-1

- Custom vector processors, 1.86 Gflop / sec peak
- 4 processors / node, 16 nodes / chassis, 4096 processors max
- 32 Gb – 64 Tb, cache coherent, physically distributed, globally addressable memory (7x – 40x bandwidth of PC clusters)
- Modified 2D torus interconnect, 400 Gb/sec peak bandwidth
**Constellation – IBM BlueGene**

- System (64 cabinets, 64x32x32)
- Cabinet (32 Nodes, 8x8x16)
- Node (32 cards, 4x4x2)
- Card (2 chips)
- Chip (2 processors)

- 5.6GFLOPS 4MB
- 11.2GFLOPS 0.5GB
- 18GFLOPS 8GB
- 5.7TFLOPS 256GB
- 360TFLOPS 16TB

---

**Parallel Architectures – Trends**

- **Top 500 List**
  - Based on performance on Linpack benchmark
  - Graph shows number of architecture of each type in Top 500

---

Graph shows the number of architecture of each type in Top 500, with clusters, constellations, MPP, SMP, and vector processors.
Parallel Architecture Trends

◆ Faster interconnects
  - Fabrics improving, but adapters are bottleneck
  - Not keeping up with node performance in clusters

◆ Larger memories
  - SMPs with 64 - 1024 GB memories
  - Deeper cache hierarchies

◆ Processor / memory integration
  - Processor-in-memory (PIM), and memory controllers
  - Relative cost of off-chip communication increases

Parallel Architecture Summary

◆ Parallel architectures provide large performance boost
◆ Different forms of parallelism may be exploited
  - Pipeline, SIMD, MIMD
◆ Many different parallel architectures
  - Shared memory, distributed memory, cluster, constellation
  - Typically built from commodity parts

◆ Challenges
  - Extracting parallel performance
    - Sustained performance usually small fraction of peak
  - Programming model
    - Parallel programming much more labor intensive