CMSC 611: Advanced Computer Architecture

Parallel Computation
Parallel Computers

• Definition: “A parallel computer is a collection of processing elements that cooperate and communicate to solve large problems fast.”
  – Almasi and Gottlieb, Highly Parallel Computing, 1989

• Parallel machines are expected to have a bigger role in the future since:
  – Microprocessors are likely to remain dominant
  – Microprocessor technology is not expected to keep the pace of performance
  – Parallel architectures extend performance
  – There has been steady progress in software development for parallel architectures
Questions about parallel computers:

• How large a collection?
• How powerful are processing elements?
• How do they cooperate and communicate?
• How are data transmitted?
• What type of interconnection?
• What are HW and SW primitives for programmers?
• Does it translate into performance?
Questions about parallel computers:

- How large a collection?
- How powerful are processing elements?
- How do they cooperate and communicate?
- How are data transmitted?
- What type of interconnection?
- What are HW and SW primitives for programmers?
- Does it translate into performance?
Level of Parallelism

- Bit-level parallelism
  - ALU parallelism: 1-bit, 4-bits, 8-bit, ...
- Instruction-level parallelism (ILP)
  - Pipelining, Superscalar, VLIW, Out-of-Order execution
- Process/Thread-level parallelism
  - Divide job into parallel tasks
- Job-level parallelism
  - Independent jobs on one computer system
**Applications**

- **Scientific Computing**
  - Nearly Unlimited Demand (Grand Challenge):
  - Successes in some real industries:
    - Petroleum: reservoir modeling
    - Automotive: crash simulation, drag analysis, engine
    - Aeronautics: airflow analysis, engine, structural mechanics
    - Pharmaceuticals: molecular modeling

<table>
<thead>
<tr>
<th>App</th>
<th>Perf (GFLOPS)</th>
<th>Memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>48 hour weather</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>72 hour weather</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Pharmaceutical design</td>
<td>100</td>
<td>10</td>
</tr>
<tr>
<td>Global Change, Genome</td>
<td>1000</td>
<td>1000</td>
</tr>
</tbody>
</table>
Commercial Applications

- Transaction processing
- File servers
- Electronic CAD simulation
- Large WWW servers
- WWW search engines
- Graphics
  - Graphics hardware
  - Render Farms
Framework

• Extend traditional computer architecture with a communication architecture
  – abstractions (HW/SW interface)
  – organizational structure to realize abstraction efficiently

• Programming Model:
  – Multiprogramming: lots of jobs, no communication
  – Shared address space: communicate via memory
  – Message passing: send and receive messages
  – Data Parallel: several agents operate on several data sets simultaneously and then exchange information globally and simultaneously (shared or message passing)
Communication Abstraction

• Shared address space:
  – e.g., load, store, atomic swap

• Message passing:
  – e.g., send, receive library calls

• Debate over this topic (ease of programming, scaling)
  – many hardware designs 1:1 programming model
Flynn Categories

- **SISD** (Single Instruction Single Data)
- **MISD** (Multiple Instruction Single Data)
- **SIMD** (Single Instruction Multiple Data)
- **MIMD** (Multiple Instruction Multiple Data)
SISD

- Uniprocessor
MISD

- No commercial examples
- Different operations to a single data set
  - Find primes
  - Crack passwords
SIMD

- Vector/Array computers
SIMD Arrays

- Performance keys
  - Utilization
  - Communication
Data Parallel Model

- Operations performed in parallel on each element of a large regular data structure, such as an array
  - One Control Processor broadcast to many processing elements (PE) with condition flag per PE so that can skip
- For distributed memory architecture data is distributed among memories
  - Data parallel model requires fast global synchronization
  - Data parallel programming languages lay out data to processor
  - Vector processors have similar ISAs, but no data placement restriction
### SIMD Utilization

- **Conditional Execution**
  - **PE Enable**
    - if \( f < 0.5 \) {...}
  - **Global enable check**
    - while \( t > 0 \) {...}
Communication: MasPar MP1

- Fast local X-net
- Slow global routing
Comunication: CM2

- Hypercube local routing
- Wormhole global routing
**Communication: PixelFlow**

- Dense connections within block
  - Single swizzle operation collects one word from each PE in block
    - Designed for antialiasing
  - NO inter-block connection
  - NO global routing