Parallel Computer Architectures

Thoai Nam
Outline

- Flynn’s Taxonomy
- Classification of Parallel Computers Based on Architectures
Flynn’s Taxonomy

- Based on notions of instruction and data streams
  - **SISD** (a Single Instruction stream, a Single Data stream)
  - **SIMD** (Single Instruction stream, Multiple Data streams)
  - **MISD** (Multiple Instruction streams, a Single Data stream)
  - **MIMD** (Multiple Instruction streams, Multiple Data stream)

- Popularity
  - **MIMD > SIMD > MISD**
- SISD

  Conventional sequential machines

  IS : Instruction Stream
  DS : Data Stream
  CU : Control Unit
  PU : Processing Unit
  MU : Memory Unit

- SISD

  Conventional sequential machines

  IS : Instruction Stream
  DS : Data Stream
  CU : Control Unit
  PU : Processing Unit
  MU : Memory Unit
- SIMD
  - Vector computers, processor arrays
  - Special purpose computations

**SIMD architecture with distributed memory**

**Program loaded from host**

**Data sets loaded from host**

**PE : Processing Element**

**LM : Local Memory**

**CU**

**PE**

**LM**

**DS**

**IS**
MISD

- Systolic arrays
- Special purpose computations

MISD architecture (the systolic array)
MIMD

- General purpose parallel computers
Classification based on Architecture

- Pipelined Computers
- Dataflow Architectures
- Data Parallel Systems
- Multiprocessors
- Multicomputers
Pipeline Computers (1)

- Instructions are divided into a number of steps (segments, stages)
- At the same time, several instructions can be loaded in the machine and be executed in different steps
Pipeline Computers (2)

- IF – instruction fetch
- ID – instruction decode and register fetch
- EX - execution and effective address calculation
- MEM – memory access
- WB - write back

<table>
<thead>
<tr>
<th>Instruction #</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instruction i</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction i+1</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction i+2</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction i+3</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction i+4</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Dataflow Architecture

- **Data-driven model**
  - A program is represented as a directed acyclic graph in which a node represents an instruction and an edge represents the data dependency relationship between the connected nodes.
  - Firing rule
    - A node can be scheduled for execution if and only if its input data become valid for consumption.

- **Dataflow languages**
  - Id, SISAL, Silage, LISP,...
  - Single assignment, applicative(functional) language
  - Explicit parallelism
The dataflow representation of an arithmetic expression

\[ z = (a + b) \times c \]
Dataflow Computer

- Execution of instructions is driven by data availability
  - What is the difference between this and normal (control flow) computers?

- Advantages
  - Very high potential for parallelism
  - High throughput
  - Free from side-effect

- Disadvantages
  - Time lost waiting for unneeded arguments
  - High control overhead
  - Difficult in manipulating data structures
Dataflow Representation

input d, e, f
    c_0 = 0
for i from 1 to 4
do
    begin
        a_i := d_i / e_i
        b_i := a_i * f_i
        c_i := b_i + c_{i-1}
    end
output a, b, c
Execution on a Control Flow Machine

Assume all the external inputs are available before entering do loop
+ : 1 cycle, * : 2 cycles, / : 3 cycles,

Sequential execution on a uniprocessor in 24 cycles

How long will it take to execute this program on a dataflow computer with 4 processors?
Execution on a Dataflow Machine

Data-driven execution on a 4-processor dataflow computer in 9 cycles

Can we further reduce the execution time of this program?
Data Parallel Systems (1)

- Programming model
  - Operations performed in parallel on each element of data structure
  - Logically single thread of control, performs sequential or parallel steps
  - Conceptually, a processor associated with each data element
SIMD Architectural model

- Array of many simple, cheap processors with little memory each
  - Processors don’t sequence through instructions
- Attached to a control processor that issues instructions
- Specialized and general communication, cheap global synchronization
Vector Processors

- Instruction set includes operations on vectors as well as scalars

- 2 types of vector computers
  - Processor arrays
  - Pipelined vector processors
A sequential computer connected with a set of identical processing elements simultaneously doing the same operation on different data. 

Eg: CM-200
Pipeline Vector Processor

- Stream vector from memory to the CPU
- Use pipelined arithmetic units to manipulate data
- Eg: Cray-1, Cyber-205
Multiprocessor

- Consists of many fully programmable processors each capable of executing its own program
- Shared address space architecture
- Classified into 2 types
  - Uniform Memory Access (UMA) Multiprocessors
  - Non-Uniform Memory Access (NUMA) Multiprocessors
- Uses a central switching mechanism to reach a centralized shared memory
- All processors have equal access time to global memory
- Tightly coupled system
- Problem: cache consistency
Crossbar switching mechanism
- Shared-bus switching mechanism
Packet-switched network
- Distributed shared memory combined by local memory of all processors
- Memory access time depends on whether it is local to the processor
- Caching shared (particularly nonlocal) data?
Current Types of Multiprocessors

- **PVP (Parallel Vector Processor)**
  - A small number of proprietary vector processors connected by a high-bandwidth crossbar switch

- **SMP (Symmetric Multiprocessor)**
  - A small number of COST microprocessors connected by a high-speed bus or crossbar switch

- **DSM (Distributed Shared Memory)**
  - Similar to SMP
  - The memory is physically distributed among nodes.
PVP (Parallel Vector Processor)

VP : Vector Processor
SM : Shared Memory

Crossbar Switch
SMP (Symmetric Multi-Processor)

P/C : Microprocessor and Cache
SM: Shared Memory

Bus or Crossbar Switch
DSM (Distributed Shared Memory)

MB: Memory Bus
P/C: Microprocessor & Cache
LM: Local Memory
DIR: Cache Directory
NIC: Network Interface

Circuitry

Custom-Designed Network
Multicomputers

- Consists of many processors with their own memory
- No shared memory
- Processors interact via message passing → loosely coupled system

P/C: Microprocessor & Cache
M: Memory
Current Types of Multicomputers

- **MPP (Massively Parallel Processing)**
  - Total number of processors > 1000

- **Cluster**
  - Each node in system has less than 16 processors.

- **Constellation**
  - Each node in system has more than 16 processors
MPP
(Massively Parallel Processing)

P/C: Microprocessor & Cache
NIC: Network Interface Circuitry

MB: Memory Bus
LM: Local Memory

Custom-Designed Network
Clusters

MB: Memory Bus
P/C: Microprocessor & Cache
M: Memory
LD: Local Disk
IOB: I/O Bus
NIC: Network Interface Circuitry

Commodity Network (Ethernet, ATM, Myrinet, VIA)
Constellations

P/C: Microprocessor & Cache
NIC: Network Interface Circuitry
IOC: I/O Controller

MB: Memory Bus
SM: Shared Memory
LD: Local Disk

Custom or Commodity Network

>= 16

P/C: Microprocessor & Cache
NIC: Network Interface Circuitry
IOC: I/O Controller

MB: Memory Bus
SM: Shared Memory
LD: Local Disk

Custom or Commodity Network

>= 16
Trend in Parallel Computer Architectures