Lecture 2

Introduction

View of the Field

Ceng505 *Parallel Computing* at October 4, 2010

Dr. Cem Özdoğan
Computer Engineering Department
Çankaya University

Contents

Field I

- Data-intensive applications;

Field I

- Data-intensive applications;
  - transaction processing,

Field I

- Data-intensive applications;
    - transaction processing,
    - information retrieval,

Field I

- Data-intensive applications;
    - transaction processing,
    - information retrieval,
    - data mining and analysis,

Field I

- Data-intensive applications;
  - transaction processing,
  - information retrieval,
  - data mining and analysis,
  - multimedia services,

Field I

- Data-intensive applications;
  - transaction processing,
  - information retrieval,
  - data mining and analysis,
  - multimedia services,
  - computational physics/chemistry/biology and nanotechnology.

Field I

- Data-intensive applications;
  - transaction processing,
  - information retrieval,
  - data mining and analysis,
  - multimedia services,
  - computational physics/chemistry/biology and nanotechnology.
- High performance may come from

Field I

- Data-intensive applications;
  - transaction processing,
  - information retrieval,
  - data mining and analysis,
  - multimedia services,
  - computational physics/chemistry/biology and nanotechnology.
- High performance may come from
  - fast dense circuitry,

Field I

- Data-intensive applications;
  - transaction processing,
  - information retrieval,
  - data mining and analysis,
  - multimedia services,
  - computational physics/chemistry/biology and nanotechnology.
- High performance may come from
  - fast dense circuitry,
  - parallelism.

Field I

- Data-intensive applications;
  - transaction processing,
  - information retrieval,
  - data mining and analysis,
  - multimedia services,
  - computational physics/chemistry/biology and nanotechnology.
- High performance may come from
  - fast dense circuitry,
  - parallelism.
- Parallel processors are computer systems consisting of

Field I

- Data-intensive applications;
  - transaction processing,
  - information retrieval,
  - data mining and analysis,
  - multimedia services,
  - computational physics/chemistry/biology and nanotechnology.
- High performance may come from
  - fast dense circuitry,
  - parallelism.
- Parallel processors are computer systems consisting of
  - multiple *processing units*

Field I

- Data-intensive applications;
    - transaction processing,
    - information retrieval,
    - data mining and analysis,
    - multimedia services,
    - computational physics/chemistry/biology and nanotechnology.
- High performance may come from
    - fast dense circuitry,
    - parallelism.
- Parallel processors are computer systems consisting of
    - multiple *processing units*
    - connected via some *interconnection network*

Field I

- Data-intensive applications;
  - transaction processing,
  - information retrieval,
  - data mining and analysis,
  - multimedia services,
  - computational physics/chemistry/biology and nanotechnology.
- High performance may come from
  - fast dense circuitry,
  - parallelism.
- Parallel processors are computer systems consisting of
  - multiple *processing units*
  - connected via some *interconnection network*
  - plus the software needed to make the processing units work together.

Field II

- *Uniprocessor* – Single processor supercomputers have achieved great speeds and have been pushing hardware technology to the physical limit of chip manufacturing.

Field II

- *Uniprocessor* – Single processor supercomputers have achieved great speeds and have been pushing hardware technology to the physical limit of chip manufacturing.
    - Physical and architectural bounds (Lithography, μm size, destructive quantum effects.

Field II

- *Uniprocessor* – Single processor supercomputers have achieved great speeds and have been pushing hardware technology to the physical limit of chip manufacturing.
    - Physical and architectural bounds (Lithography, μm size, destructive quantum effects.
    - Proposed solutions are maskless lithography process and nanoimprint lithography for the semiconductor).

Field II

- *Uniprocessor* – Single processor supercomputers have achieved great speeds and have been pushing hardware technology to the physical limit of chip manufacturing.
  - Physical and architectural bounds (Lithography, μm size, destructive quantum effects.
  - Proposed solutions are maskless lithography process and nanoimprint lithography for the semiconductor).
  - Uniprocessor systems can achieve to a limited computational power and not capable of delivering solutions to some problems in reasonable time.

Field II

- *Uniprocessor* – Single processor supercomputers have achieved great speeds and have been pushing hardware technology to the physical limit of chip manufacturing.
  - Physical and architectural bounds (Lithography, μm size, destructive quantum effects.
  - Proposed solutions are maskless lithography process and nanoimprint lithography for the semiconductor).
  - Uniprocessor systems can achieve to a limited computational power and not capable of delivering solutions to some problems in reasonable time.
- *Multiprocessor* – Multiple processors cooperate to jointly execute a single computational task in order to speed up its execution.

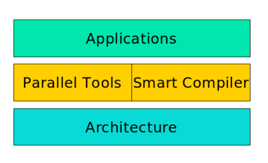


Figure: Abstraction Layers

Field III

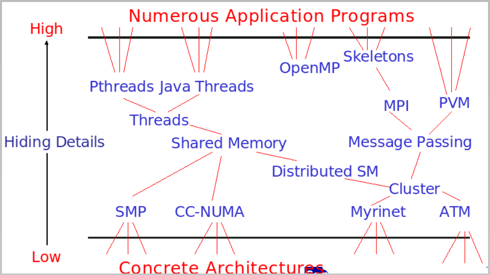


Figure: View of the Field

- New issues arise;

Field III

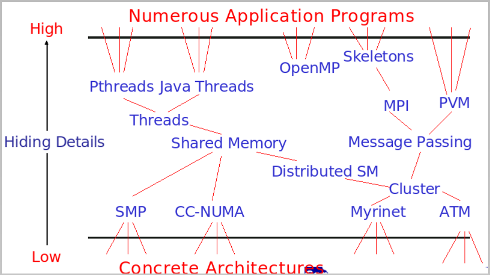


Figure: View of the Field

- New issues arise;
  - Multiple threads of control vs. single thread of control

Field III

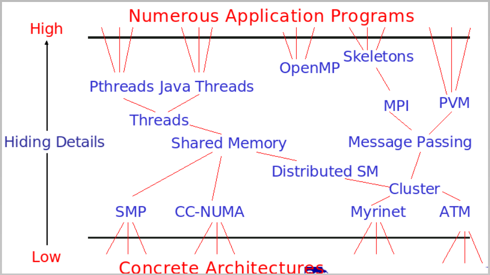


Figure: View of the Field

- New issues arise;
  - Multiple threads of control vs. single thread of control
  - Partitioning for concurrent execution

Field III

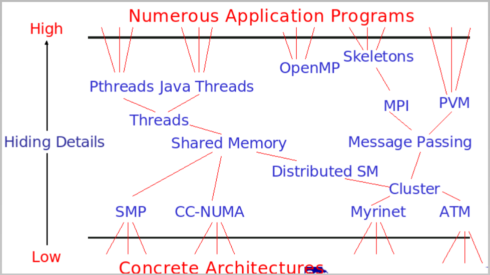


Figure: View of the Field

- New issues arise;
    - Multiple threads of control vs. single thread of control
    - Partitioning for concurrent execution
    - Task Scheduling

Field III

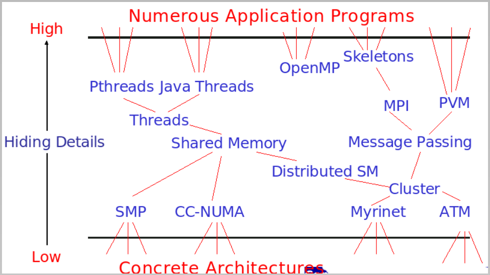


Figure: View of the Field

- New issues arise;
  - Multiple threads of control vs. single thread of control
  - Partitioning for concurrent execution
  - Task Scheduling
  - Synchronization

Field III

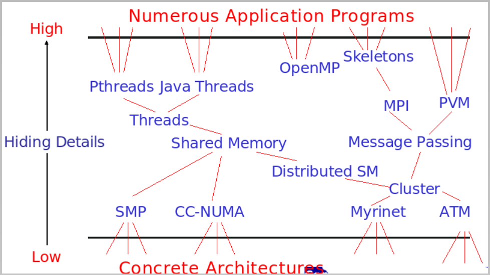


Figure: View of the Field

- New issues arise;
  - Multiple threads of control vs. single thread of control
  - Partitioning for concurrent execution
  - Task Scheduling
  - Synchronization
  - Performance

Trends

- Past Trends in Parallel Architecture (inside the box)

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).
    - Expensive systems.

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).
    - Expensive systems.
    - Quickly becoming outdated.

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).
    - Expensive systems.
    - Quickly becoming outdated.
  - In the form of internally linked processors was the main form of parallelism.

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).
    - Expensive systems.
    - Quickly becoming outdated.
  - In the form of internally linked processors was the main form of parallelism.
  - Advances in computer networks ⇒ in the form of networked autonomous computers.

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).
    - Expensive systems.
    - Quickly becoming outdated.
  - In the form of internally linked processors was the main form of parallelism.
  - Advances in computer networks $\Rightarrow$ in the form of networked autonomous computers.
- New Trends in Parallel Architecture (outside the box)

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).
    - Expensive systems.
    - Quickly becoming outdated.
  - In the form of internally linked processors was the main form of parallelism.
  - Advances in computer networks $\Rightarrow$ in the form of networked autonomous computers.
- New Trends in Parallel Architecture (outside the box)
  - Instead of putting everything in a single box and *tightly couple* processors to memory, the Internet achieved a kind of parallelism by *loosely* connecting everything outside of the box.

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).
    - Expensive systems.
    - Quickly becoming outdated.
  - In the form of internally linked processors was the main form of parallelism.
  - Advances in computer networks $\Rightarrow$ in the form of networked autonomous computers.
- New Trends in Parallel Architecture (outside the box)
  - Instead of putting everything in a single box and *tightly couple* processors to memory, the Internet achieved a kind of parallelism by *loosely* connecting everything outside of the box.
  - Network of PCs and workstations connected via LAN or WAN forms a Parallel System.

Trends

- Past Trends in Parallel Architecture (inside the box)
  - Completely custom designed components; *processors, memory, interconnects, I/O*.
  - The first three are the major components for the aspects of the parallel computation.
    - Longer R&D time (2-3 years).
    - Expensive systems.
    - Quickly becoming outdated.
  - In the form of internally linked processors was the main form of parallelism.
  - Advances in computer networks $\Rightarrow$ in the form of networked autonomous computers.
- New Trends in Parallel Architecture (outside the box)
  - Instead of putting everything in a single box and *tightly couple* processors to memory, the Internet achieved a kind of parallelism by *loosely* connecting everything outside of the box.
  - Network of PCs and workstations connected via LAN or WAN forms a Parallel System.
  - Compete favourably (cost/performance).

Trends

- Past Trends in Parallel Architecture (inside the box)
    - Completely custom designed components; *processors, memory, interconnects, I/O*.
    - The first three are the major components for the aspects of the parallel computation.
        - Longer R&D time (2-3 years).
        - Expensive systems.
        - Quickly becoming outdated.
    - In the form of internally linked processors was the main form of parallelism.
    - Advances in computer networks $\Rightarrow$ in the form of networked autonomous computers.
- New Trends in Parallel Architecture (outside the box)
    - Instead of putting everything in a single box and *tightly couple* processors to memory, the Internet achieved a kind of parallelism by *loosely* connecting everything outside of the box.
    - Network of PCs and workstations connected via LAN or WAN forms a Parallel System.
    - Compete favourably (cost/performance).
    - Utilize unused cycles of systems sitting idle.

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era
2. Time-Sharing Era

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era
2. Time-Sharing Era
3. Desktop Era

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era
2. Time-Sharing Era
3. Desktop Era
4. Network Era. They can generally be classified into two main categories:

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era
2. Time-Sharing Era
3. Desktop Era
4. Network Era. They can generally be classified into two main categories:
    1. shared memory,

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era
2. Time-Sharing Era
3. Desktop Era
4. Network Era. They can generally be classified into two main categories:
    1. shared memory,
    2. distributed memory systems.

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era
2. Time-Sharing Era
3. Desktop Era
4. Network Era. They can generally be classified into two main categories:
   1. shared memory,
   2. distributed memory systems.

   - The number of processors in a single machine ranged from several in a shared memory computer to hundreds of thousands in a massively parallel system.

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era
2. Time-Sharing Era
3. Desktop Era
4. Network Era. They can generally be classified into two main categories:
    1. shared memory,
    2. distributed memory systems.

    - The number of processors in a single machine ranged from several in a shared memory computer to hundreds of thousands in a massively parallel system.
    - Examples of parallel computers during this era include Sequent Symmetry, Intel iPSC, nCUBE, Intel Paragon, Thinking Machines (CM-2, CM-5), MsPar (MP), Fujitsu (VPP500), and others.

Four Decades of Computing

Most computer scientists agree that there have been four distinct paradigms or eras of computing. These are: batch, time-sharing, desktop, and network.

1. Batch Era
2. Time-Sharing Era
3. Desktop Era
4. Network Era. They can generally be classified into two main categories:
    1. shared memory,
    2. distributed memory systems.

    - The number of processors in a single machine ranged from several in a shared memory computer to hundreds of thousands in a massively parallel system.
    - Examples of parallel computers during this era include Sequent Symmetry, Intel iPSC, nCUBE, Intel Paragon, Thinking Machines (CM-2, CM-5), MsPar (MP), Fujitsu (VPP500), and others.
5. Current Trends: Clusters, Grids.

Flynn’s Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.

Flynn's Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn's classification scheme is based on the notion of a stream of information.

Flynn's Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn's classification scheme is based on the notion of a stream of information.
  - Two types of information flow into a processor:

Flynn's Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn's classification scheme is based on the notion of a stream of information.
  - Two types of information flow into a processor:
    1. **Instruction**. The instruction stream is defined as the sequence of instructions performed by the processing unit.

Flynn’s Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn’s classification scheme is based on the notion of a stream of information.
  - Two types of information flow into a processor:
    1. **Instruction**. The instruction stream is defined as the sequence of instructions performed by the processing unit.
    2. **Data**. The data stream is defined as the data traffic exchanged between the memory and the processing unit.

Flynn’s Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn’s classification scheme is based on the notion of a stream of information.
  - Two types of information flow into a processor:
    1. **Instruction**. The instruction stream is defined as the sequence of instructions performed by the processing unit.
    2. **Data**. The data stream is defined as the data traffic exchanged between the memory and the processing unit.
- According to Flynn’s classification, either of the instruction or data streams can be **single** or **multiple**.

Flynn's Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn's classification scheme is based on the notion of a stream of information.
  - Two types of information flow into a processor:
    1. **Instruction**. The instruction stream is defined as the sequence of instructions performed by the processing unit.
    2. **Data**. The data stream is defined as the data traffic exchanged between the memory and the processing unit.
- According to Flynn's classification, either of the instruction or data streams can be **single** or **multiple**.
- Computer architecture can be classified into the following four distinct categories:

Flynn's Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn's classification scheme is based on the notion of a <u>stream of information</u>.
    - Two types of information flow into a processor:
        1. **Instruction**. The instruction stream is defined as the sequence of instructions performed by the processing unit.
        2. **Data**. The data stream is defined as the data traffic exchanged between the memory and the processing unit.
- According to Flynn's classification, either of the instruction or data streams can be **single** or **multiple**.
- Computer architecture can be classified into the following four distinct categories:
    1. single instruction single data streams (SISD)

Flynn's Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn's classification scheme is based on the notion of a stream of information.
  - Two types of information flow into a processor:
    1. **Instruction**. The instruction stream is defined as the sequence of instructions performed by the processing unit.
    2. **Data**. The data stream is defined as the data traffic exchanged between the memory and the processing unit.
- According to Flynn's classification, either of the instruction or data streams can be **single** or **multiple**.
- Computer architecture can be classified into the following four distinct categories:
  1. single instruction single data streams (SISD)
  2. single instruction multiple data streams (SIMD)

Flynn's Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn's classification scheme is based on the notion of a stream of information.
  - Two types of information flow into a processor:
    1. **Instruction**. The instruction stream is defined as the sequence of instructions performed by the processing unit.
    2. **Data**. The data stream is defined as the data traffic exchanged between the memory and the processing unit.
- According to Flynn's classification, either of the instruction or data streams can be **single** or **multiple**.
- Computer architecture can be classified into the following four distinct categories:
  1. single instruction single data streams (SISD)
  2. single instruction multiple data streams (SIMD)
  3. multiple instruction single data streams (MISD)

Flynn's Taxonomy of Computer Architecture I

- The most popular taxonomy of computer architecture was defined by Flynn in 1966.
- Flynn's classification scheme is based on the notion of a <u>stream of information</u>.
  - Two types of information flow into a processor:
    1. **Instruction**. The instruction stream is defined as the sequence of instructions performed by the processing unit.
    2. **Data**. The data stream is defined as the data traffic exchanged between the memory and the processing unit.
- According to Flynn's classification, either of the instruction or data streams can be **single** or **multiple**.
- Computer architecture can be classified into the following four distinct categories:
  1. single instruction single data streams (SISD)
  2. single instruction multiple data streams (SIMD)
  3. multiple instruction single data streams (MISD)
  4. multiple instruction multiple data streams (MIMD).

Flynn’s Taxonomy of Computer Architecture II

- SISD;

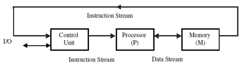


Figure: SISD Architecture.

Flynn's Taxonomy of Computer Architecture II

- SISD;

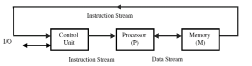


Figure: SISD Architecture.

- SIMD;

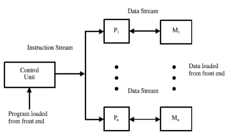


Figure: SIMD Architecture.

Flynn's Taxonomy of Computer Architecture II

- SISD;

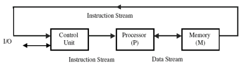


Figure: SISD Architecture.

- SIMD;

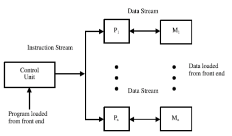


Figure: SIMD Architecture.

- MIMD;

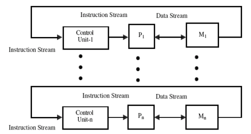


Figure: MIMD Architecture.

Flynn’s Taxonomy of Computer Architecture III

Parallel computers are either SIMD or MIMD.

- When there is only one control unit and all processors execute the same instruction in a synchronized fashion, the parallel machine is classified as SIMD.

Flynn's Taxonomy of Computer Architecture III

Parallel computers are either SIMD or MIMD.

- When there is only one control unit and all processors execute the same instruction in a synchronized fashion, the parallel machine is classified as SIMD.
- In a MIMD machine, each processor has its own control unit and can execute different instructions on different data.

Flynn’s Taxonomy of Computer Architecture III

Parallel computers are either SIMD or MIMD.

- When there is only one control unit and all processors execute the same instruction in a synchronized fashion, the parallel machine is classified as SIMD.
- In a MIMD machine, each processor has its own control unit and can execute different instructions on different data.
- In the MISD category, the same stream of data flows through a linear array of processors executing different instruction streams. In practice, there is no viable MISD machine; however, some authors have considered *pipelined machines* as examples for MISD.

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
    - shared memory
    - or message passing methods.

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based
  - switch-based.

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based
  - switch-based.
- SIMD Computers

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based
  - switch-based.
- SIMD Computers
- MIMD Shared Memory, MIMD Distributed Memory

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based
  - switch-based.

- SIMD Computers
- MIMD Shared Memory, MIMD Distributed Memory
- Bus based, Switch based

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based
  - switch-based.

- SIMD Computers
- MIMD Shared Memory, MIMD Distributed Memory
- Bus based, Switch based
- CC-NUMA

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based
  - switch-based.

- SIMD Computers
- MIMD Shared Memory, MIMD Distributed Memory
- Bus based, Switch based
- CC-NUMA
- Clusters, Grid Computing

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based
  - switch-based.

- SIMD Computers
- MIMD Shared Memory, MIMD Distributed Memory
- Bus based, Switch based
- CC-NUMA
- Clusters, Grid Computing
  - Grids are geographically distributed platforms for computation.

Parallel and Distributed Computers I

- The processing units can communicate and interact with each other using either
  - shared memory
  - or message passing methods.
- The interconnection network for shared memory systems can be classified as
  - bus-based
  - switch-based.

- SIMD Computers
- MIMD Shared Memory, MIMD Distributed Memory
- Bus based, Switch based
- CC-NUMA
- Clusters, Grid Computing
  - Grids are geographically distributed platforms for computation.
  - They provide dependable, consistent, general, and inexpensive access to high end computational capabilities.

Parallel and Distributed Computers II

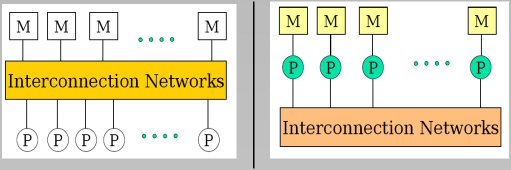


Figure: (a) MIMD Shared Memory, (b) MIMD Distributed Memory.

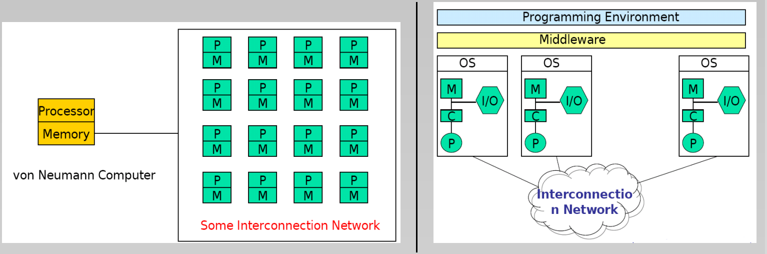


Figure: (a) SIMD Distributed Computers, (b) Clusters.

SIMD Architecture I

- The SIMD model of parallel computing consists of two parts as shown in Fig. 7a. :

SIMD Architecture I

- The SIMD model of parallel computing consists of two parts as shown in Fig. 7a. :
  1. a front-end computer of the usual von Neumann style,

SIMD Architecture I

- The SIMD model of parallel computing consists of two parts as shown in Fig. 7a. :
  1. a front-end computer of the usual von Neumann style,
  2. a processor array.

SIMD Architecture I

- The SIMD model of parallel computing consists of two parts as shown in Fig. 7a. :
  1. a front-end computer of the usual von Neumann style,
  2. a processor array.
- Each processor in the array has a small amount of local memory where the *distributed data resides* while it is being processed in parallel.

SIMD Architecture I

- The SIMD model of parallel computing consists of two parts as shown in Fig. 7a. :
  1. a front-end computer of the usual von Neumann style,
  2. a processor array.
- Each processor in the array has a small amount of local memory where the *distributed data resides* while it is being processed in parallel.
- The similarity between serial and data parallel programming is one of the strong points of *data* parallelism.

SIMD Architecture I

- The SIMD model of parallel computing consists of two parts as shown in Fig. 7a. :
  1. a front-end computer of the usual von Neumann style,
  2. a processor array.
- Each processor in the array has a small amount of local memory where the *distributed data resides* while it is being processed in parallel.
- The similarity between serial and data parallel programming is one of the strong points of *data* parallelism.
- Processors either do nothing or exactly the same operations at the same time.

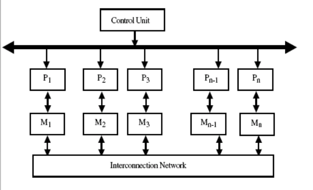
SIMD Architecture I

- The SIMD model of parallel computing consists of two parts as shown in Fig. 7a. :
  1. a front-end computer of the usual von Neumann style,
  2. a processor array.
- Each processor in the array has a small amount of local memory where the *distributed data resides* while it is being processed in parallel.
- The similarity between serial and data parallel programming is one of the strong points of *data* parallelism.
- Processors either do nothing or exactly the same operations at the same time.
- In SIMD architecture, parallelism is exploited by applying simultaneous operations across large sets of data.

SIMD Architecture I

- The SIMD model of parallel computing consists of two parts as shown in Fig. 7a. :
  1. a front-end computer of the usual von Neumann style,
  2. a processor array.
- Each processor in the array has a small amount of local memory where the *distributed data resides* while it is being processed in parallel.
- The similarity between serial and data parallel programming is one of the strong points of *data* parallelism.
- Processors either do nothing or exactly the same operations at the same time.
- In SIMD architecture, parallelism is exploited by applying simultaneous operations across large sets of data.
- There are two main configurations that have been used in SIMD machines (see Fig. 5).

SIMD Architecture II



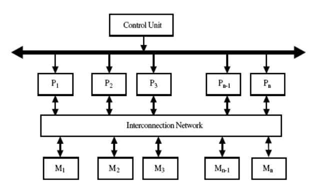
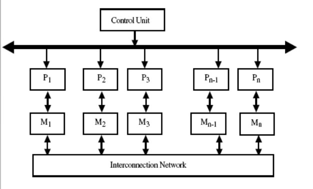


Figure: Two SIMD Schemes.

1. Each processor has its own local memory.

SIMD Architecture II



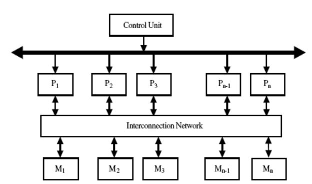
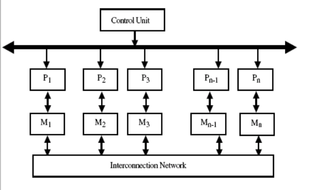


Figure: Two SIMD Schemes.

1. Each processor has its own local memory.
   - Processors can communicate with each other through the interconnection network.

SIMD Architecture II



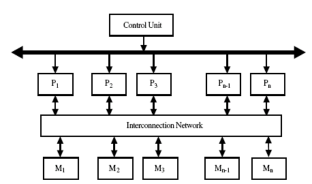


Figure: Two SIMD Schemes.

1. Each processor has its own local memory.
   - Processors can communicate with each other through the interconnection network.
   - If the interconnection network does not provide direct connection between a given pair of processors, then this pair can exchange data via an intermediate processor.

SIMD Architecture II

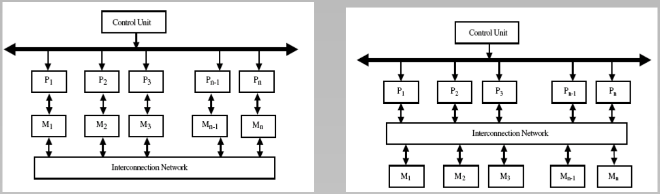


Figure: Two SIMD Schemes.

1. Each processor has its own local memory.
   - Processors can communicate with each other through the interconnection network.
   - If the interconnection network does not provide direct connection between a given pair of processors, then this pair can exchange data via an intermediate processor.
2. In the second SIMD scheme,

SIMD Architecture II

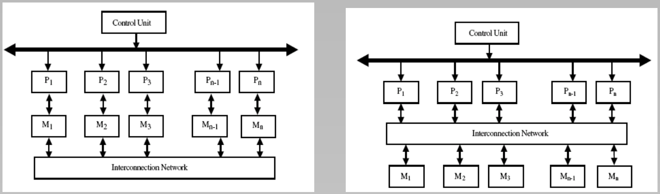


Figure: Two SIMD Schemes.

1. Each processor has its own local memory.
   - Processors can communicate with each other through the interconnection network.
   - If the interconnection network does not provide direct connection between a given pair of processors, then this pair can exchange data via an intermediate processor.
2. In the second SIMD scheme,
   - Processors and memory modules communicate with each other via the interconnection network.

SIMD Architecture II

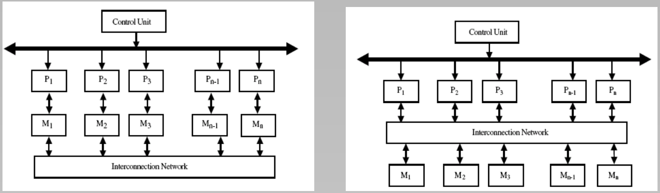


Figure: Two SIMD Schemes.

1. Each processor has its own local memory.
   - Processors can communicate with each other through the interconnection network.
   - If the interconnection network does not provide direct connection between a given pair of processors, then this pair can exchange data via an intermediate processor.
2. In the second SIMD scheme,
   - Processors and memory modules communicate with each other via the interconnection network.
   - Two processors can transfer data between each other via intermediate memory module(s) or possibly via intermediate processor(s).

MIMD Architecture I

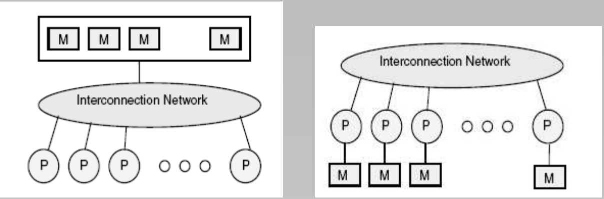


Figure: Two MIMD Categories; Shared Memory and Message Passing MIMD Architectures.

- It was apparent that distributed memory is the only way efficiently to increase the number of processors managed by a parallel and distributed system.

MIMD Architecture I

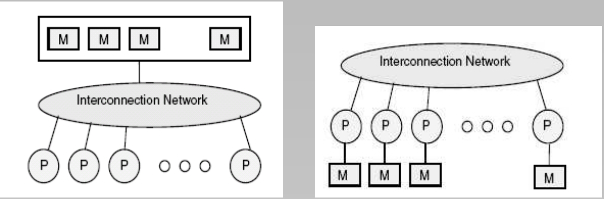


Figure: Two MIMD Categories; Shared Memory and Message Passing MIMD Architectures.

- It was apparent that distributed memory is the only way efficiently to increase the number of processors managed by a parallel and distributed system.
- If scalability to larger and larger systems (as measured by the number of processors) was to continue, systems had to use distributed memory techniques.

MIMD Architecture II

- Two broad categories, see Figure 9:

MIMD Architecture II

- Two broad categories, see Figure 9:
  1. **Shared memory**. Processors exchange information through their **central shared memory**.

MIMD Architecture II

- Two broad categories, see Figure 9:
  1. **Shared memory**. Processors exchange information through their **central shared memory**.
     - Because access to shared memory is balanced, these systems are also called SMP (symmetric multiprocessor) systems.

MIMD Architecture II

- Two broad categories, see Figure 9:
  1. **Shared memory**. Processors exchange information through their **central shared memory**.
     - Because access to shared memory is balanced, these systems are also called SMP (symmetric multiprocessor) systems.
  2. **Message passing.** Also referred to as distributed memory. Processors exchange information through their **interconnection network**.

MIMD Architecture II

- Two broad categories, see Figure 9:
  1. **Shared memory**. Processors exchange information through their **central shared memory**.
     - Because access to shared memory is balanced, these systems are also called SMP (symmetric multiprocessor) systems.
  2. **Message passing.** Also referred to as distributed memory. Processors exchange information through their **interconnection network**.
     - There is no global memory, so it is necessary to *move data from one local memory to another by means of message passing.*

MIMD Architecture II

- Two broad categories, see Figure 9:
  1. **Shared memory**. Processors exchange information through their **central shared memory**.
     - Because access to shared memory is balanced, these systems are also called SMP (symmetric multiprocessor) systems.
  2. **Message passing.** Also referred to as distributed memory. Processors exchange information through their **interconnection network**.
     - There is no global memory, so it is necessary to *move data from one local memory to another by means of message passing.*
     - This is typically done by a **Send/Receive pair** of commands, which must be written into the application software by a programmer

MIMD Architecture II

- Two broad categories, see Figure 9:
  1. **Shared memory**. Processors exchange information through their **central shared memory**.
     - Because access to shared memory is balanced, these systems are also called SMP (symmetric multiprocessor) systems.
  2. **Message passing.** Also referred to as distributed memory. Processors exchange information through their **interconnection network**.
     - There is no global memory, so it is necessary to *move data from one local memory to another by means of message passing.*
     - This is typically done by a **Send/Receive pair** of commands, which must be written into the application software by a programmer
     - Data copying and dealing with consistency issues.

MIMD Architecture III

- Programming in the shared memory model was easier, and designing systems in the message passing model provided scalability.

MIMD Architecture III

- Programming in the shared memory model was easier, and designing systems in the message passing model provided scalability.
- The distributed-shared memory (DSM) architecture began to appear in systems. In such systems,

MIMD Architecture III

- Programming in the shared memory model was easier, and designing systems in the message passing model provided scalability.
- The distributed-shared memory (DSM) architecture began to appear in systems. In such systems,
  - memory is physically distributed; for example, the hardware architecture follows the message passing school of design,

MIMD Architecture III

- Programming in the shared memory model was easier, and designing systems in the message passing model provided scalability.
- The distributed-shared memory (DSM) architecture began to appear in systems. In such systems,
  - memory is physically distributed; for example, the hardware architecture follows the message passing school of design,
  - but the programming model follows the shared memory school of thought.

MIMD Architecture III

- Programming in the shared memory model was easier, and designing systems in the message passing model provided scalability.
- The distributed-shared memory (DSM) architecture began to appear in systems. In such systems,
  - memory is physically distributed; for example, the hardware architecture follows the message passing school of design,
  - but the programming model follows the shared memory school of thought.
  - Thus, the DSM machine is a *hybrid* that takes advantage of both design schools.

Shared Memory Organization I

- A number of basic issues in the design of shared memory systems have to be taken into consideration.

Shared Memory Organization I

- A number of basic issues in the design of shared memory systems have to be taken into consideration.
- These include access control, synchronization, protection, and security.

Shared Memory Organization I

- A number of basic issues in the design of shared memory systems have to be taken into consideration.
- These include access control, synchronization, protection, and security.
    - *Access control* determines which process accesses are possible to which resources.

Shared Memory Organization I

- A number of basic issues in the design of shared memory systems have to be taken into consideration.
- These include access control, synchronization, protection, and security.
  - *Access control* determines which process accesses are possible to which resources.
  - *Synchronization* constraints limit the time of accesses from sharing processes to shared resources.

Shared Memory Organization I

- A number of basic issues in the design of shared memory systems have to be taken into consideration.
- These include access control, synchronization, protection, and security.
  - *Access control* determines which process accesses are possible to which resources.
  - *Synchronization* constraints limit the time of accesses from sharing processes to shared resources.
  - *Protection* is a system feature that prevents processes from making arbitrary access to resources belonging to other processes.

Shared Memory Organization I

- A number of basic issues in the design of shared memory systems have to be taken into consideration.
- These include access control, synchronization, protection, and security.
  - *Access control* determines which process accesses are possible to which resources.
  - *Synchronization* constraints limit the time of accesses from sharing processes to shared resources.
  - *Protection* is a system feature that prevents processes from making arbitrary access to resources belonging to other processes.
- The simplest shared memory system consists of one memory module that can be accessed from two processors.

Shared Memory Organization I

- A number of basic issues in the design of shared memory systems have to be taken into consideration.
- These include access control, synchronization, protection, and security.
  - *Access control* determines which process accesses are possible to which resources.
  - *Synchronization* constraints limit the time of accesses from sharing processes to shared resources.
  - *Protection* is a system feature that prevents processes from making arbitrary access to resources belonging to other processes.
- The simplest shared memory system consists of one memory module that can be accessed from two processors.
- Requests arrive at the memory module through its two ports.

Shared Memory Organization II

Depending on the interconnection network, a shared memory system leads to systems can be classified as:

- **Uniform Memory Access (UMA)**. A shared memory is accessible by all processors through an interconnection network in the same way a single processor accesses its memory.

Shared Memory Organization II

Depending on the interconnection network, a shared memory system leads to systems can be classified as:

- **Uniform Memory Access (UMA)**. A shared memory is accessible by all processors through an interconnection network in the same way a single processor accesses its memory.
  - Therefore, all processors have equal access time to any memory location.

Shared Memory Organization II

Depending on the interconnection network, a shared memory system leads to systems can be classified as:

- **Uniform Memory Access (UMA)**. A shared memory is accessible by all processors through an interconnection network in the same way a single processor accesses its memory.
    - Therefore, all processors have equal access time to any memory location.
- **Nonuniform Memory Access (NUMA)**. Each processor has part of the shared memory attached.

Shared Memory Organization II

Depending on the interconnection network, a shared memory system leads to systems can be classified as:

- **Uniform Memory Access (UMA)**. A shared memory is accessible by all processors through an interconnection network in the same way a single processor accesses its memory.
  - Therefore, all processors have equal access time to any memory location.
- **Nonuniform Memory Access (NUMA)**. Each processor has part of the shared memory attached.
  - However, the access time to modules depends on the distance to the processor. This results in a nonuniform memory access time.

Shared Memory Organization II

Depending on the interconnection network, a shared memory system leads to systems can be classified as:

- **Uniform Memory Access (UMA)**. A shared memory is accessible by all processors through an interconnection network in the same way a single processor accesses its memory.
  - Therefore, all processors have equal access time to any memory location.
- **Nonuniform Memory Access (NUMA)**. Each processor has part of the shared memory attached.
  - However, the access time to modules depends on the distance to the processor. This results in a nonuniform memory access time.
- **Cache-Only Memory Architecture (COMA)**. Similar to the NUMA, each processor has part of the shared memory in the COMA.

Shared Memory Organization II

Depending on the interconnection network, a shared memory system leads to systems can be classified as:

- **Uniform Memory Access (UMA)**. A shared memory is accessible by all processors through an interconnection network in the same way a single processor accesses its memory.
    - Therefore, all processors have equal access time to any memory location.
- **Nonuniform Memory Access (NUMA)**. Each processor has part of the shared memory attached.
    - However, the access time to modules depends on the distance to the processor. This results in a nonuniform memory access time.
- **Cache-Only Memory Architecture (COMA)**. Similar to the NUMA, each processor has part of the shared memory in the COMA.
    - However, in this case the shared memory consists of cache memory.

Shared Memory Organization II

Depending on the interconnection network, a shared memory system leads to systems can be classified as:

- **Uniform Memory Access (UMA)**. A shared memory is accessible by all processors through an interconnection network in the same way a single processor accesses its memory.
  - Therefore, all processors have equal access time to any memory location.
- **Nonuniform Memory Access (NUMA)**. Each processor has part of the shared memory attached.
  - However, the access time to modules depends on the distance to the processor. This results in a nonuniform memory access time.
- **Cache-Only Memory Architecture (COMA)**. Similar to the NUMA, each processor has part of the shared memory in the COMA.
  - However, in this case the shared memory consists of cache memory.
  - A COMA system requires that data be migrated to the processor requesting it.

Message Passing Organization I

- Message passing systems are a class of multiprocessors in which each processor has access to its own local memory.

Message Passing Organization I

- Message passing systems are a class of multiprocessors in which each processor has access to its own local memory.
- Unlike shared memory systems, communications in message passing systems are performed via send and receive operations.

Message Passing Organization I

- Message passing systems are a class of multiprocessors in which each processor has access to its own local memory.
- Unlike shared memory systems, communications in message passing systems are performed via send and receive operations.
- Nodes are typically able to store messages in buffers (temporary memory locations where messages wait until they can be sent or received), and perform send/receive operations at the same time as processing.

Message Passing Organization I

- Message passing systems are a class of multiprocessors in which each processor has access to its own local memory.
- Unlike shared memory systems, communications in message passing systems are performed via send and receive operations.
- Nodes are typically able to store messages in buffers (temporary memory locations where messages wait until they can be sent or received), and perform send/receive operations at the same time as processing.
- The processing units of a message passing system may be connected in a variety of ways ranging from architecture-specific interconnection structures to geographically dispersed networks.

Message Passing Organization II

Two important design factors must be considered in designing interconnection networks for message passing systems. These are the link bandwidth and the network latency.

1. The link bandwidth is defined as the number of bits that can be transmitted per unit time (bits/s).

Message Passing Organization II

Two important design factors must be considered in designing interconnection networks for message passing systems. These are the link bandwidth and the network latency.

1. The link bandwidth is defined as the number of bits that can be transmitted per unit time (bits/s).
2. The network latency is defined as the time to complete a message transfer.