Parallel and Distributed Computing

Distributed Memory Programming Model

Prof. Maryam Saif

Learning Objectives

- Understand distributed memory systems
- Explain the distributed memory architecture
- Describe how processes communicate
- Understand message passing mechanisms
- Compare data distribution strategies:
  - Block
  - Cyclic
  - Block Cyclic

Memory Model

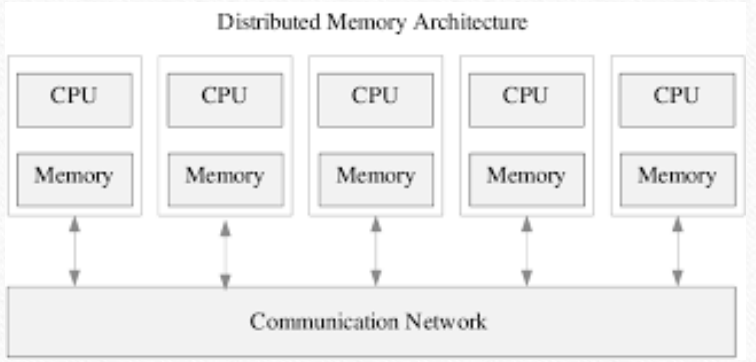
- A memory model defines how:
  - Data is stored
  - Data is accessed by processes
  - Consistency is maintained
  - Determines the programming style for parallel systems
- Two major models:
  - Shared Memory
  - Distributed Memory

Distributed Memory Model

- Each processor has **its own private memory**
- No global shared address space
- Processes cannot directly access remote memory
- Communication is done explicitly via messages
- **Key Characteristics**
  - Explicit communication
  - High scalability
  - Programmer controls data distribution

Distributed Memory Architecture

- System consists of multiple independent nodes
- Each node contains:
  - CPU(s)Local memory
  - Nodes are connected using:
    - High-speed interconnection network (Ethernet, InfiniBand)
- Key Features:
    - Memory is physically distributed
    - No memory contention
    - Communication latency depends on network
    - Suitable for large-scale computing systems (clusters, supercomputers)



How Do Processes Communicate?

- Since memory is private, communication is mandatory
- Processes exchange data explicitly
- Used for:
  - Data sharing
  - SynchronizationTask coordination

Why Data Distribution Matters?

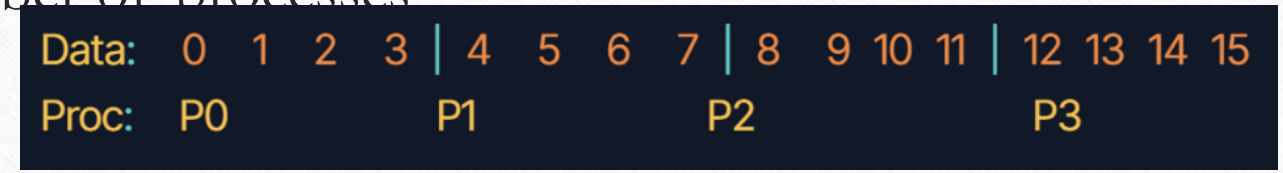
- Determines:
  - Load balance
  - Communication cost
  - Overall performance
- Poor data distribution leads to:
  - Idle processors
  - High communication overhead

Strategies to Divide Data

- Three common strategies:
- Block Distribution
- Cyclic Distribution
- Block Cyclic Distribution

Block Distribution

- Data is divided into **contiguous blocks**
- Divides block per process
- Each process gets one block
- Block size = Total data / Number of processes
- **Example**
  - 16 elements, 4 processes
  - Each process gets 4 consecutive elements



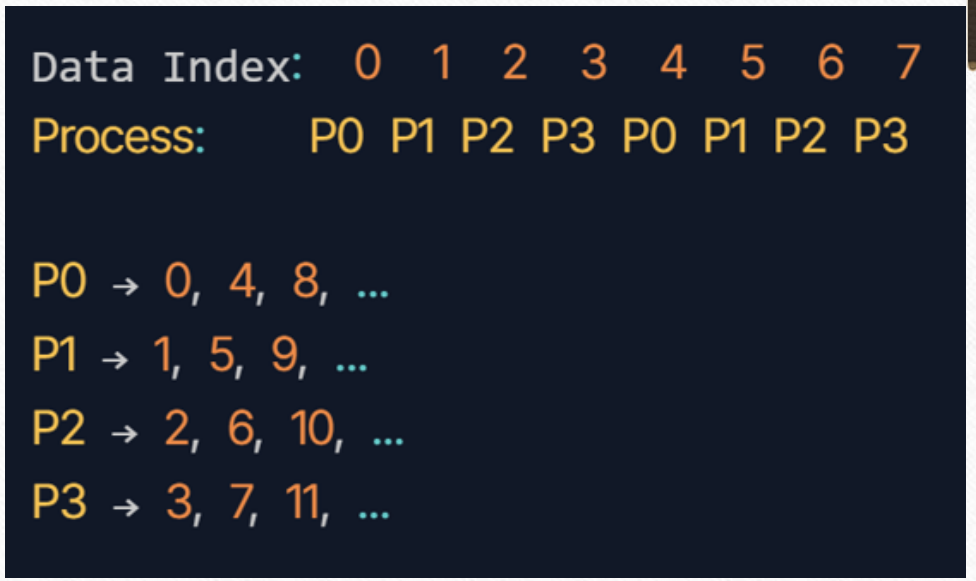
Advantages & Disadvantages

- **Advantages**
- Simple implementation
- Good data locality
- Low communication cost

- **Disadvantages**
- Load imbalance if computation per data item is uneven
- Not suitable for irregular workloads

Cyclic Distribution

- Data elements are assigned to processes in a round-robin manner
- Each process gets non-contiguous elements
- Distribution repeats cyclically
- Each process gets the nth element



Advantages & Disadvantages

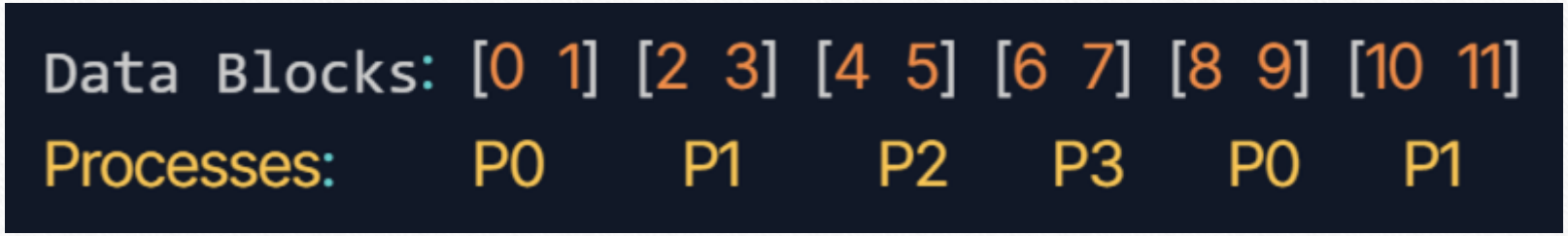
- **Advantages**
- Excellent load balancing
- Suitable for irregular or unpredictable workloads

- **Disadvantages**
- Poor data locality
- Increased communication overhead
- More complex indexing

Block Cyclic

- Hybrid of block and cyclic distribution
- Data divided into **blocks of fixed size**
- Blocks are assigned cyclically to processes
- **Block size (b)** is a key parameter

Example: Block size:2



Advantages & Disadvantages

- **Advantages**
- Better load balance than block distribution
- Better locality than cyclic distribution
- Widely used in numerical algorithms

- **Disadvantages**
- More complex implementation
- Block size selection affects performance

Choosing the Right Strategy

- Block → Uniform workload, low communication
- Cyclic → Highly irregular workload
- Block Cyclic → Large numerical computations (matrices)

Comparison

Strategy	Load Balance	Locality	Complexity
Block	Medium	High	Low
Cyclic	High	Low	Medium
Block Cyclic	High	Medium	High