

Phys 7411: Parallel Programming Considerations

Juana Moreno - moreno@lsu.edu

and

Karen Tomko – ktomko@osc.edu

Spring 2009

Parallel Programming

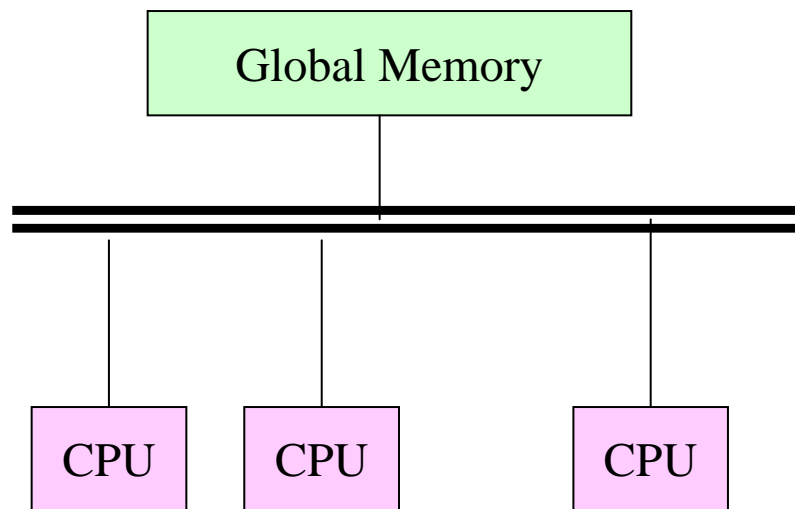
- Architectural considerations
- Decomposing programs for parallelism
- Enhancing parallel performance
- Memory-Hierarchy management
- Parallel debugging
- Performance analysis and tuning
- Parallel input/output
- Reading (Dongarra: Chapter 3)

Topics:

- Decomposition strategies:
 - Task parallelism
 - Data parallelism
- Programming model
 - Shared-memory model
 - Message-passing model
- Achieving high performance: scalability, load balance...

Architectural considerations

- **Shared memory:** symmetric multiprocessor



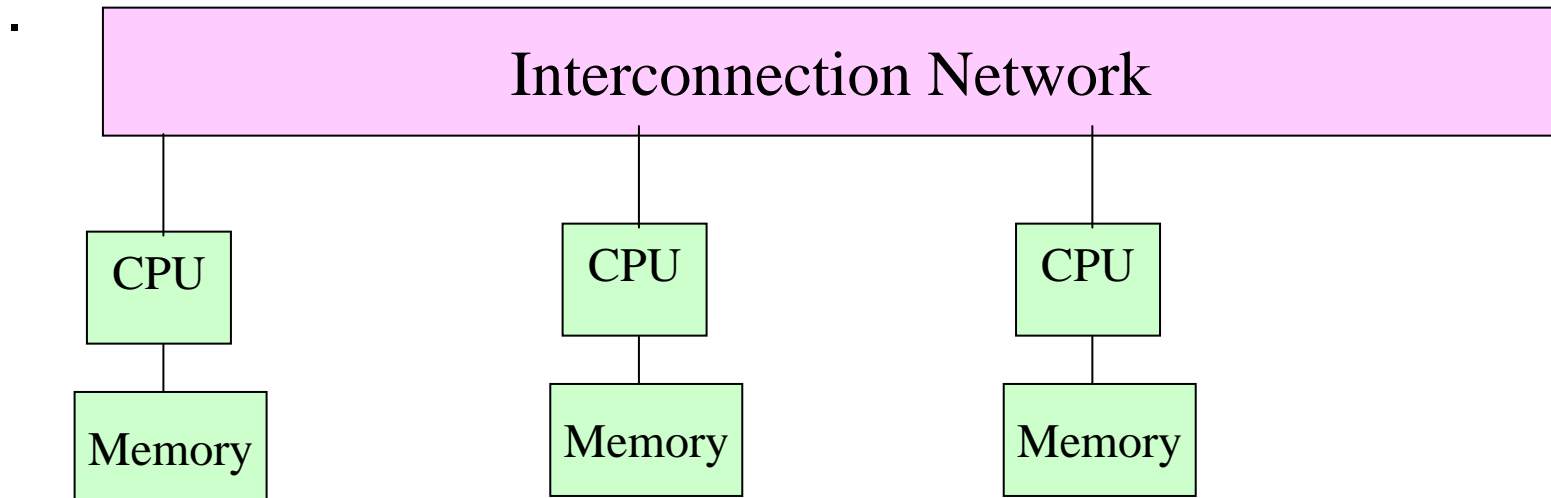
Important the synchronization of accesses to shared data structures \Rightarrow not scalable to large number of processors.

Architectural considerations

- **Distributed memory**

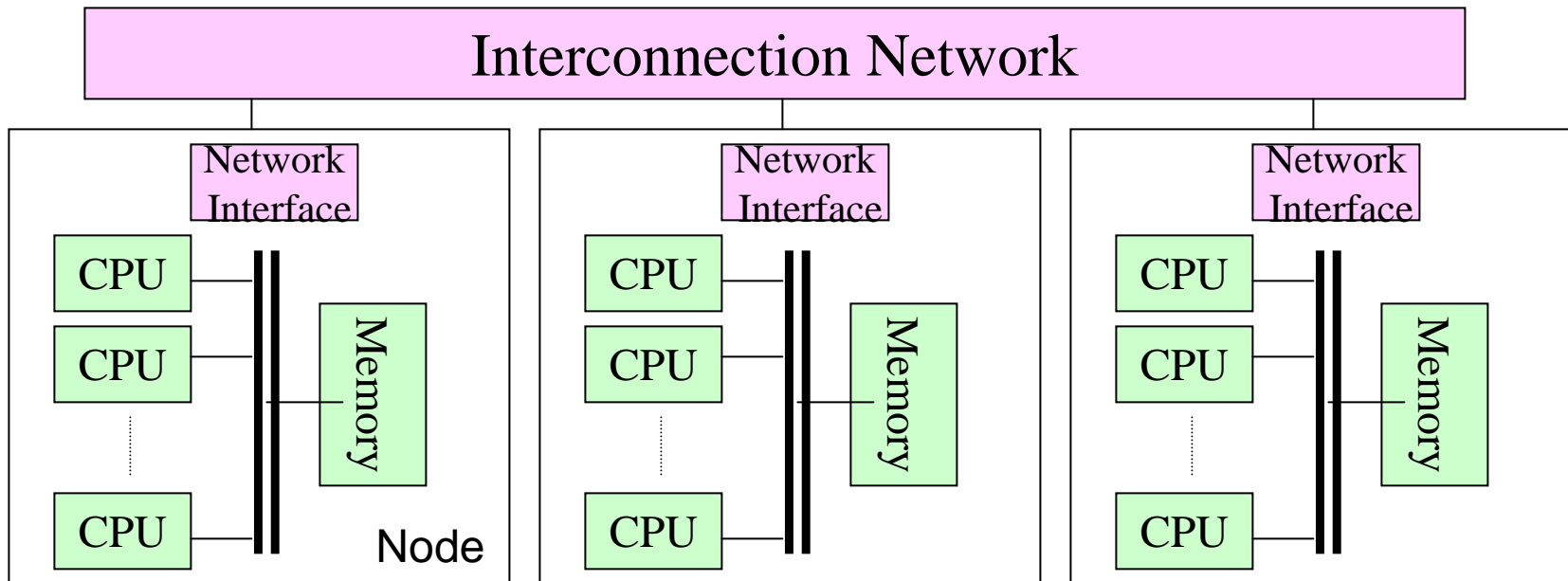
Scalable if the interconnection network is scalable.

Message-passing programming model (typically pays to send fewer, longer messages).



Architectural considerations

- **Hybrid systems:** distributed shared-memory system with symmetric multiprocessors



Architectural considerations

Memory hierarchy

- Cache 1: 2-5 cycles
Cache 2: 10-20 cycles
Cache miss: 100 cycles
- Caches transfer data in a minimum size:
cache block
- Performance: latency (time to fetch a datum)
bandwidth (#bites/time)
length of cache block

Decomposing Programs for Parallelism

- **Identification of parallelism:** C_1 , C_2 can be run in parallel without synchronization if and only if none of the following holds:
 1. C_1 writes into a location that is later read by C_2 : read-after-write (RAW) race.
 2. C_1 reads from a location that is later written into by C_2 : write-after-read (WAR) race.
 1. C_1 writes from a location that C_2 overwrites later: write-after-write (WAW) race.

Decomposing Programs for Parallelism

- **Decomposition strategy:**
 - Task parallelism
 - Data parallelism
 - Pipelining: each processor assigned to a different stage of a multistep sequential computation

Decomposing Programs for Parallelism

- **Decomposition strategy:**
 - Task parallelism
 - Data parallelism
 - Pipelining: each processor assigned to a different stage of a multistep sequential computation

Decomposing Programs for Parallelism

- **Programming models:**
 - Shared memory: need for synchronization
 - Message-passing model: send/receive data commands replace synchronization

Decomposing Programs for Parallelism

- **Implementation styles:** iteratives loops, recursive transversal of tree-like data structures:
- **Iteratives loops**
 - Parallel loop programming: on shared-memory systems; be aware of data races (dependences)
 - SPMD programming (single-program, multiple data)
 - Recursive task programming

