CSCI341

Lecture 38, Introduction to Multicore Architectures

GOAL: PERFORMANCE

Recall:

Power as the overriding issue.

Performance, heat, power efficiency.

PIPELINING

"Exploits potential parallelism among instructions."

"Instruction-level parallelism"

PROCESS-LEVEL PARALLELISM

Utilizing multiple processors by running *independent* programs simultaneously.

PARALLEL PROCESSING PROGRAM

Executing *one* program upon multiple processors simultaneously.

MULTI-PROCESSOR ARCHITECTURES

A system with at least two processors.

MULTI-CORE ARCHITECTURES

A system with multiple processors (“cores”) within a single integrated circuit.

SEQUENTIAL VS. CONCURRENT

		Software	
		Sequential	**Concurrent**
Hardware	Serial	Matrix Multiply written in MatLab running on an Intel Pentium 4	Windows Vista Operating System running on an Intel Pentium 4
	Parallel	Matrix Multiply written in MATLAB running on an Intel Xeon e5345 (Clovertown)	Windows Vista Operating System running on an Intel Xeon e5345 (Clovertown)

THE PROBLEM

(not about the hardware)

It is difficult to write software that uses multiple processors that complete tasks faster.

Why?

MUST YIELD THE BENEFIT

The parallel implementation *must* be faster,
especially as the number of processors increase.

Otherwise, what's the point?

Single-processor instruction-level parallelism has evolved.
(see superscalar & out-of-order execution)

COMPLICATIONS

- scheduling
- load balancing
- time for synchronization
- communication overhead
- Amdahl's law

Example: multiple journalists writing a story.

SMP

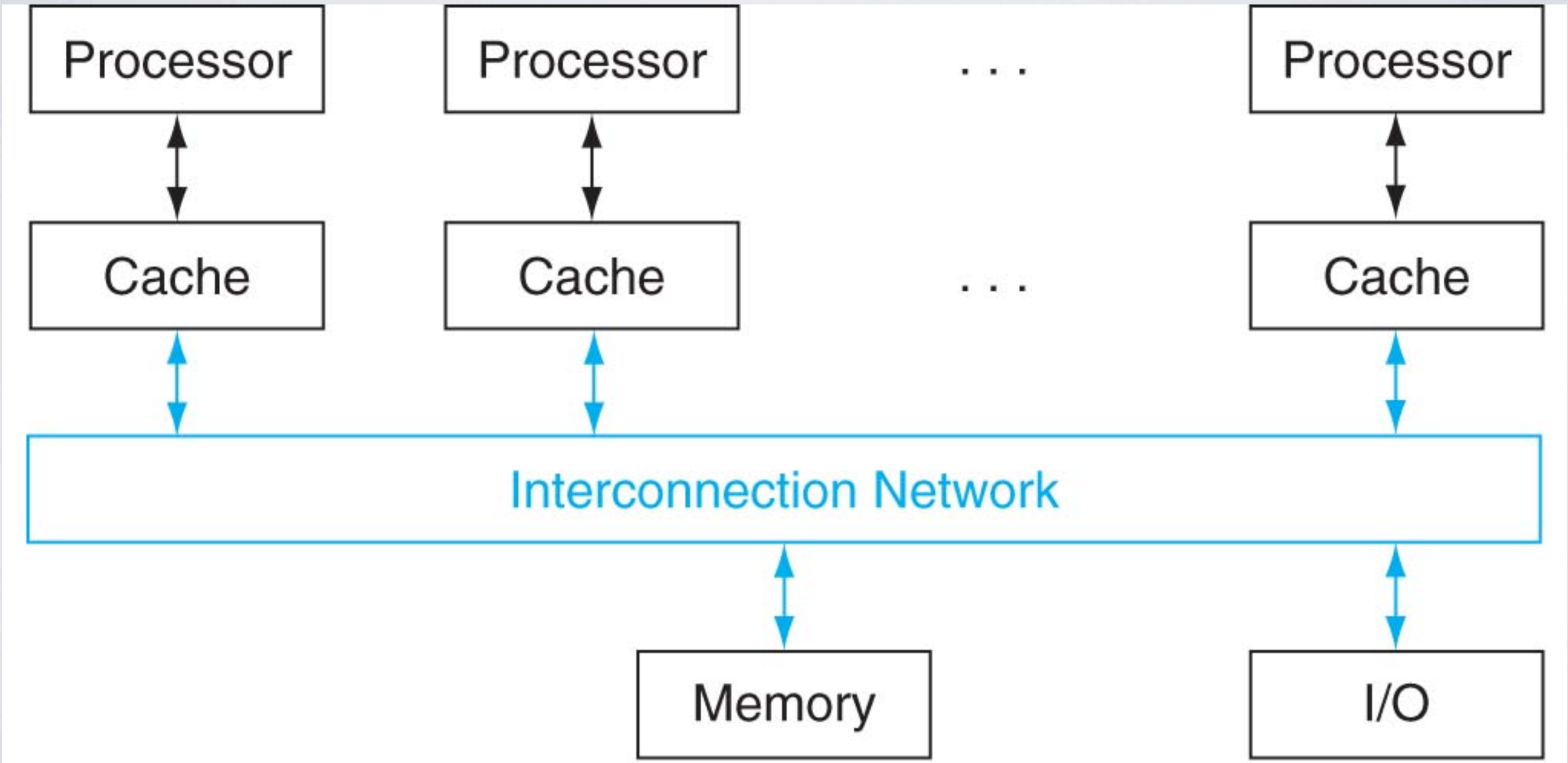
Shared Memory Multiprocessor

Multiple processors, single memory address space.

All cores have access to all data.

(Multi-core architectures generally use this approach)

SMP


Processor
Processor
. . .
Processor
Cache
Cache
. . .
Cache
Interconnection Network
Memory
I/O

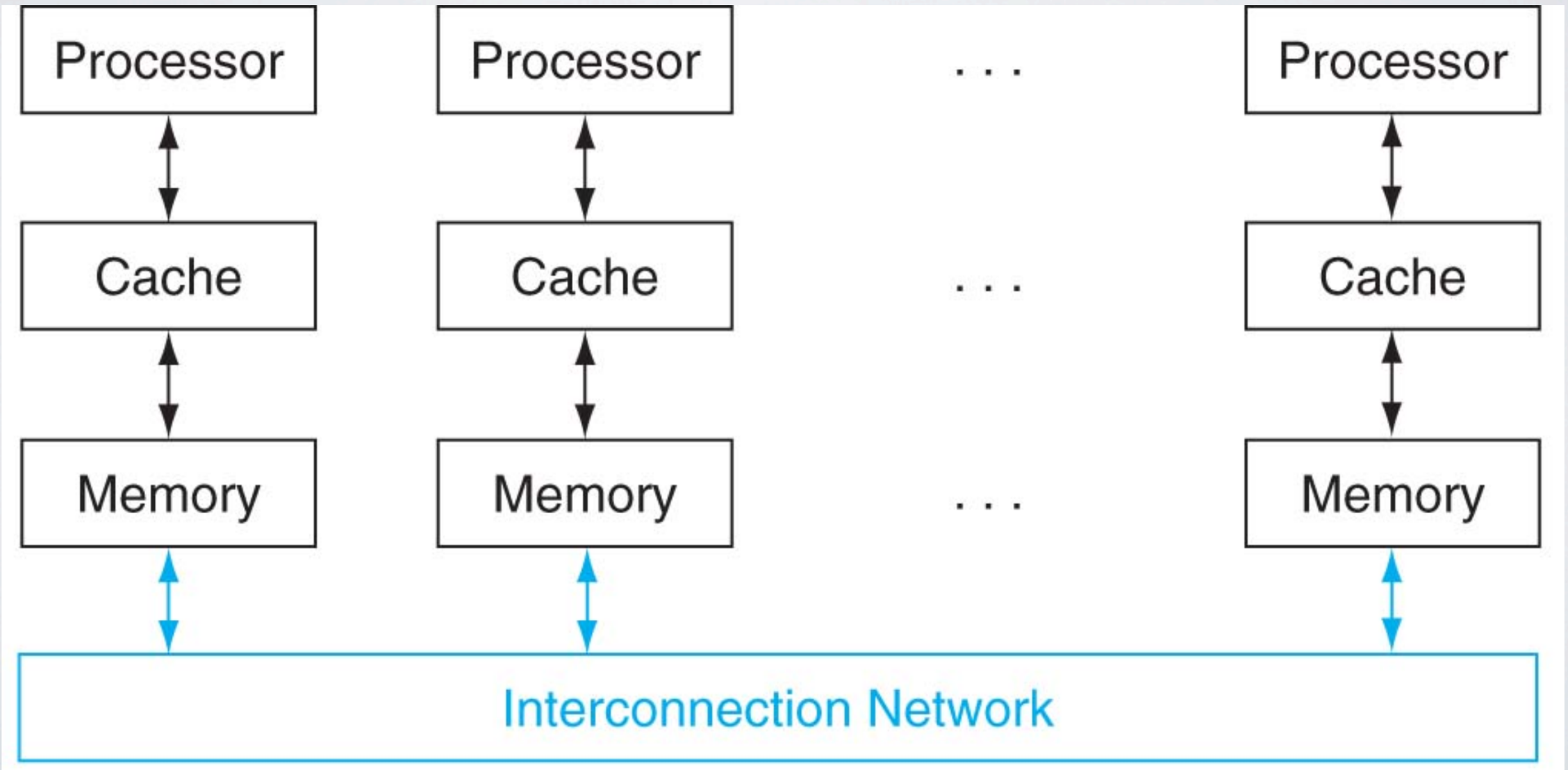
SYNCHRONIZATION

Coordinating operations on shared data between multiple processors.

Common solution: locks.

MESSAGE PASSING

What if each processor has its own address space?



MESSAGE PASSING

Pragmatically, manifests as clusters of individual machines.

But, there's a cost to administering these individual physical machines.

VIRTUAL MACHINES

An additional layer of abstraction on top of hardware.

Multiple cluster nodes on top of hardware, each capable of sending/receiving messages.

SO MUCH MORE...

- Multithreading
- MIMD (Multiple Instruction / Multiple Data Streams)
- Vector architectures (see Cray)
- GPUs

AND MORE...

Storage & I/O (Chapter 6)

One simple approach: memory-mapped I/O

AND MORE...

Many instructions are loads/stores...
how can we exploit the memory hierarchy?

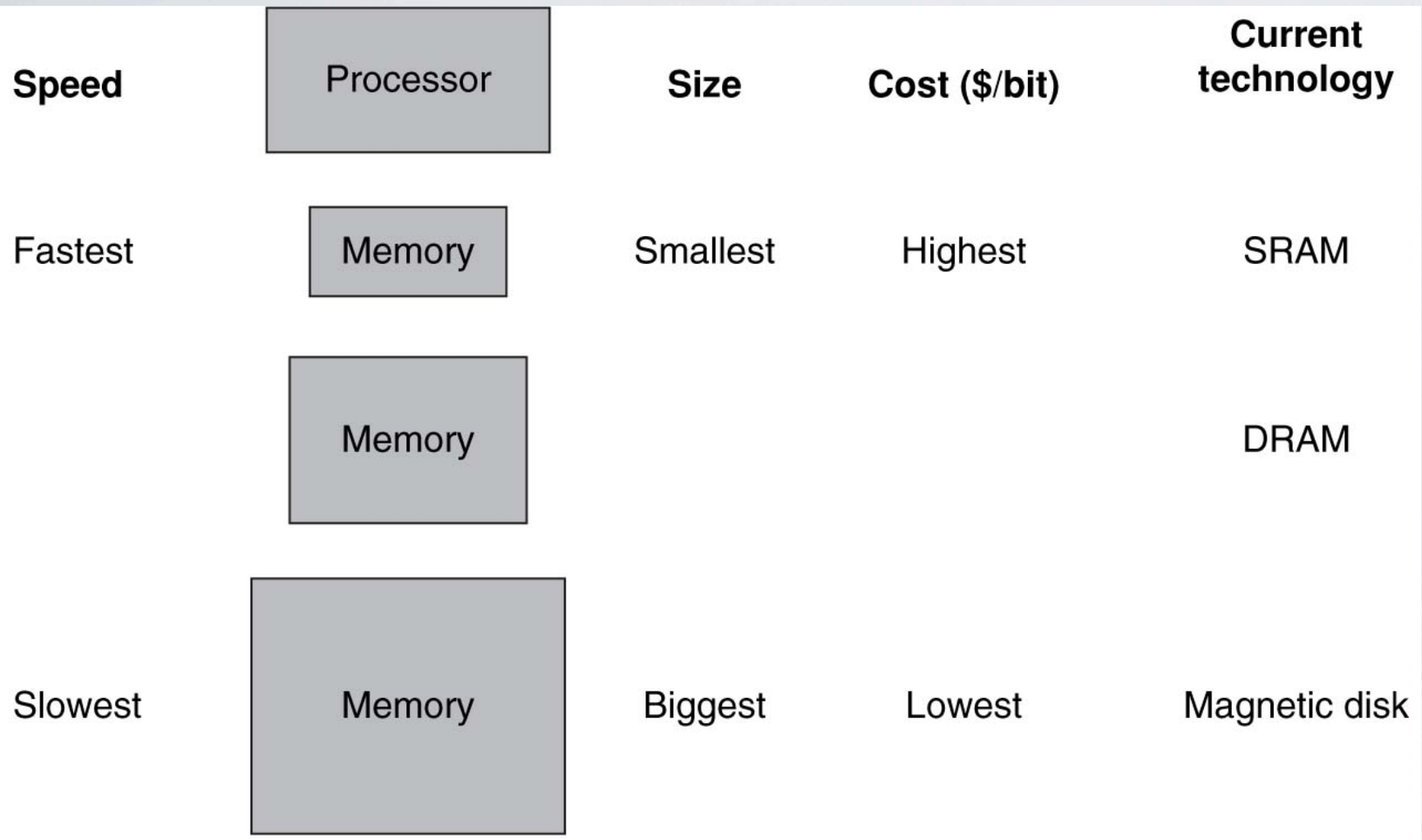
PRINCIPAL OF LOCALITY

- Temporal
- Spatial

PRINCIPAL OF LOCALITY

Memory closest to the processor fastest (most expensive).

HIERARCHY

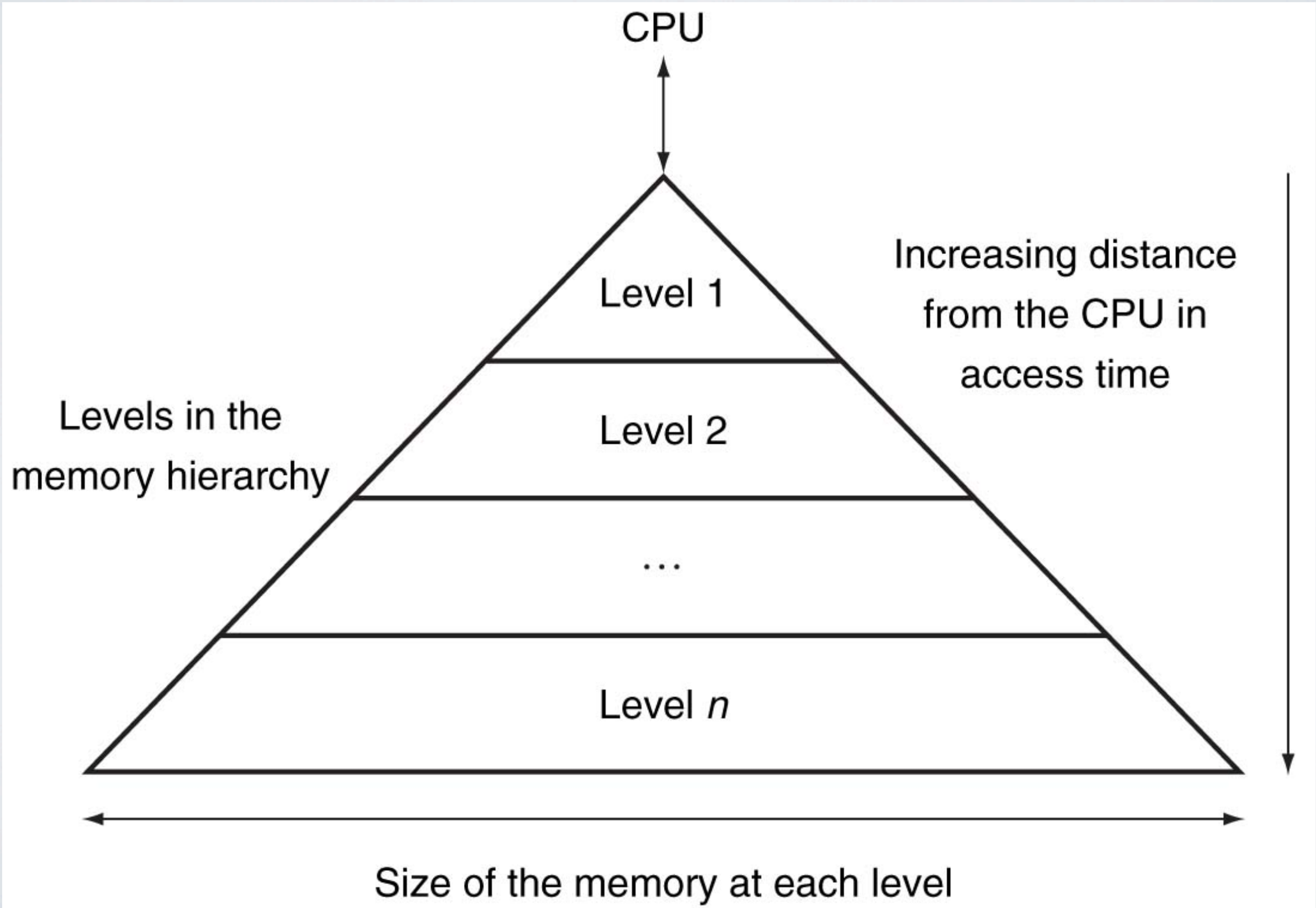


< 3 ns $2000/GB

< 70 ns $20/GB

< 20m ns $0.25/GB

HIERARCHY


CPU
Level 1
Level 2
…
Level n
Increasing distance from the CPU in access time
Levels in the memory hierarchy
Size of the memory at each level

HOMEWORK

- Reading 32
- Final exam program

No more homework!