AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Procesory i Architektura Systemów Komputerowych

Redundant Arrays of Inexpensive Disks (RAID) & Parallel Processors

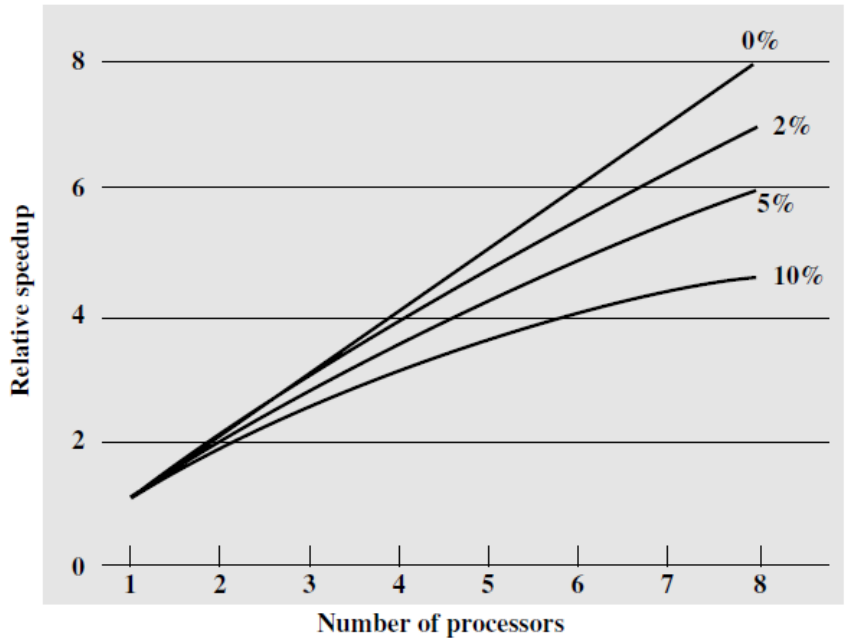
IET
Katedra Elektroniki
Kraków 2015
dr inż. Roman Rumian

Amdahl's law states that:

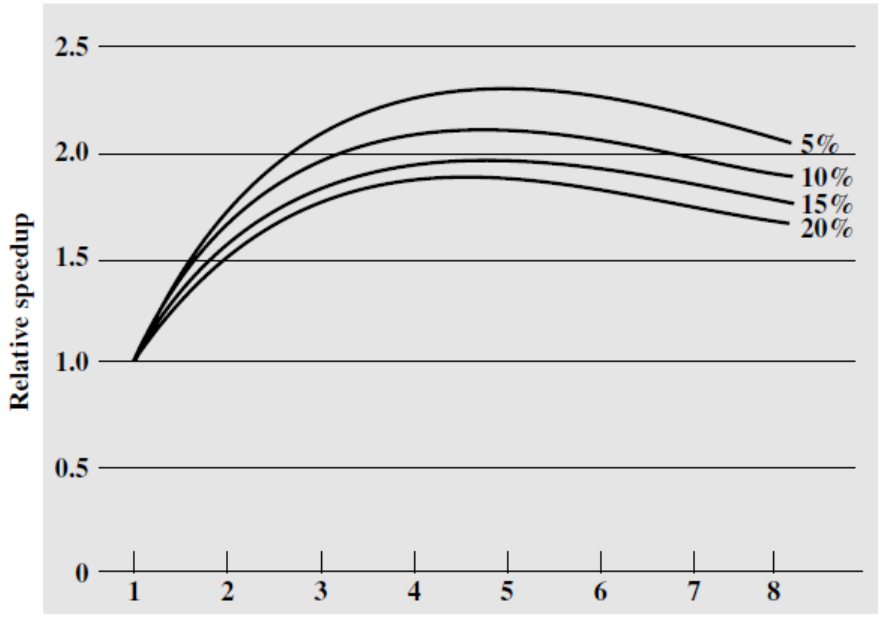
$$\text{Speedup} = \frac{\text{time to execute program on a single processor}}{\text{time to execute program on } N \text{ parallel processors}}$$

$$= \frac{1}{(1 - f) + \frac{f}{N}}$$

The law assumes a program in which a fraction of the execution time involves code that is inherently serial and a fraction f that involves code that is infinitely parallelizable with no scheduling overhead.



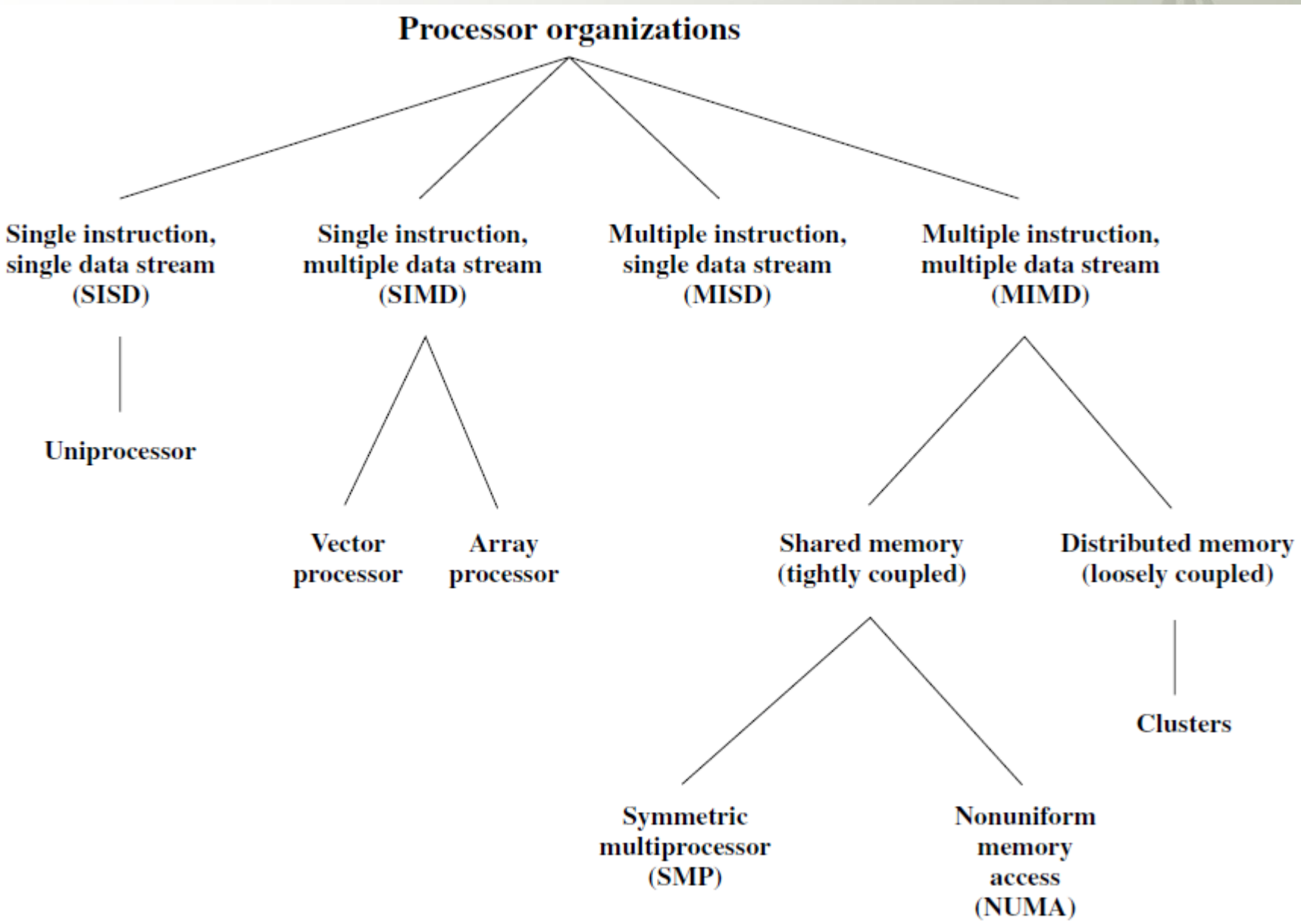
(a) Speedup with 0%, 2%, 5%, and 10% sequential portions



(b) Speedup with overheads

multiprocessor
A computer system with at least two processors. This computer is in contrast to a uniprocessor, which has one, and is increasingly hard to find today.
task-level parallelism or **process-level parallelism**
Utilizing multiple processors by running independent programs simultaneously.
parallel processing program
A single program that runs on multiple processors simultaneously.
cluster
A set of computers connected over a local area network that function as a single large multiprocessor.
multicore microprocessor
A microprocessor containing multiple processors ("cores") in a single integrated circuit. Virtually all microprocessors today in desktops and servers are multicore.
shared memory multiprocessor (SMP)
A parallel processor with a single physical address space.

Getting good speed-up on a multiprocessor while keeping the problem size fixed is harder than getting good speed-up by increasing the size of the problem.
strong scaling
Speedup achieved on a multiprocessor without increasing the size of the problem.
weak scaling
Speedup achieved on a multiprocessor while increasing the size of the problem proportionally to the increase in the number of processors.



A Taxonomy of Parallel Processor Architectures

SISD or Single Instruction stream, Single Data stream.
A uniprocessor.

MIMD or Multiple Instruction streams, Multiple Data streams.
A multiprocessor.

SPMD Single Program, Multiple Data streams. The conventional MIMD programming model, where a single program runs across all processors.

SIMD or Single Instruction stream, Multiple Data streams.
The same instruction is applied to many data streams, as in a vector processor.

		Data Streams	
		Single	Multiple
Instruction Streams	Single	SISD: Intel Pentium 4	SIMD: SSE instructions of x86
	Multiple	MISD: No examples today	MIMD: Intel Core i7

Hardware categorization and examples based on number of instruction streams and data streams: SISD, SIMD, MISD, and MIMD.

Hardware multithreading
Increasing utilization of a processor by switching to another thread when one thread is stalled.

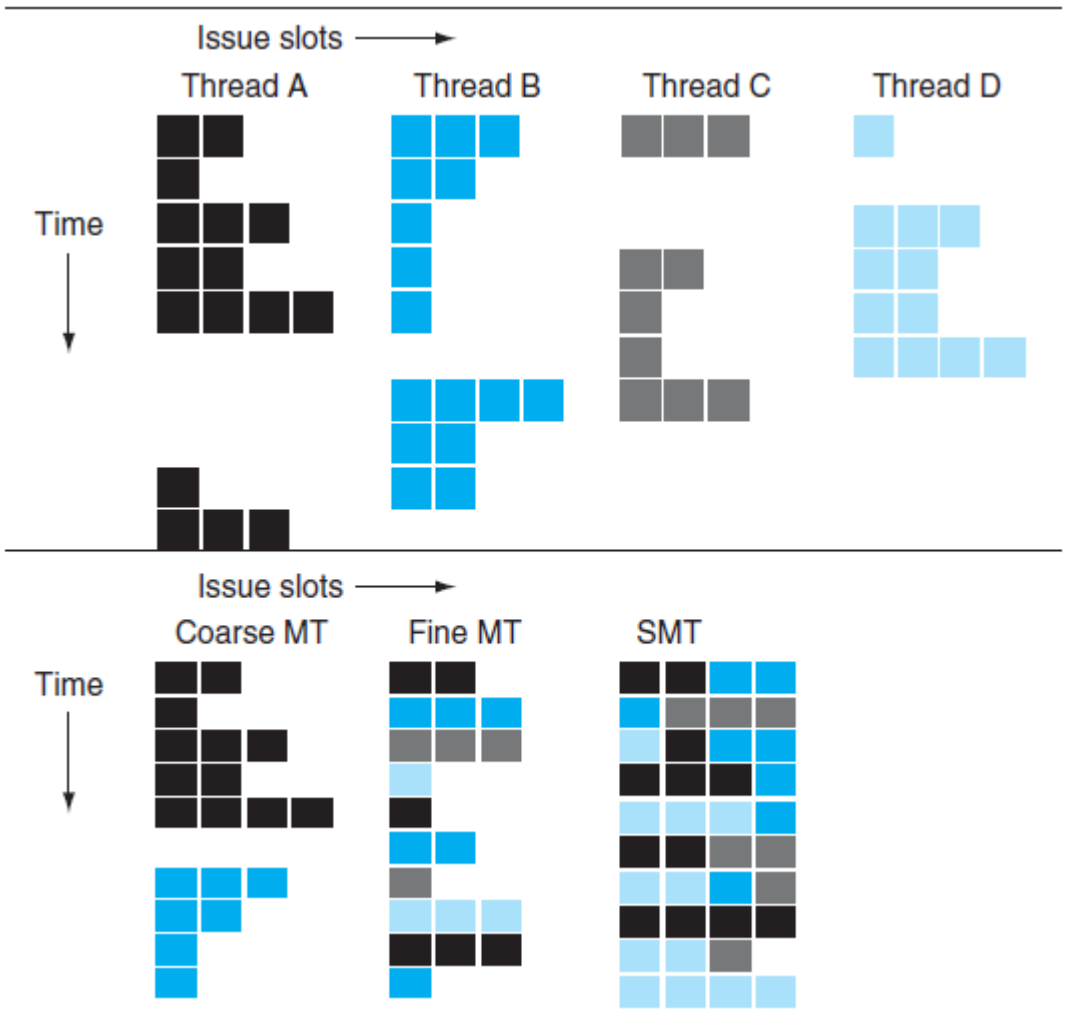
thread
A thread includes the program counter, the register state, and the stack. It is a lightweight process; whereas threads commonly share a single address space, processes don't.

process
A process includes one or more threads, the address space, and the operating system state. Hence, a process switch usually invokes the operating system, but not a thread switch.

fine-grained multithreading
A version of hardware multithreading that implies switching between threads after every instruction.

coarse-grained multithreading
A version of hardware multithreading that implies switching between threads only after
significant events, such as a last-level cache miss.

Simultaneous multithreading (SMT)
A version of multithreading that lowers the cost of multithreading by utilizing the resources needed for multiple issue, dynamically scheduled microarchitecture.



How four threads use the issue slots of a superscalar processor in different approaches.
The four threads at the top show how each would execute running alone on a standard superscalar processor without multithreading support. The three examples at the bottom show how they would execute running together in three multithreading options. The horizontal dimension represents the instruction issue capability in each clock cycle. The vertical dimension represents a sequence of clock cycles.
An empty (white) box indicates that the corresponding issue slot is unused in that clock cycle. The shades of gray and color correspond to four different threads in the multithreading processors. The additional pipeline start-up effects for coarse multithreading, which are not illustrated in this figure, would lead to further loss in throughput for coarse multithreading.

◆ A traditional way to increase system performance is to use multiple processors that can execute in parallel to support a given workload.The two most common multiple-processor organizations are **symmetric multiprocessors** (SMPs) and clusters. More recently, **nonuniform memory access** (NUMA) systems have been introduced commercially.

◆ An SMP consists of multiple similar processors within the same computer, interconnected by a bus or some sort of switching arrangement. The most critical problem to address in an SMP is that of cache coherence. Each processor has its own cache and so it is possible for a given line of data to be present in more than one cache. If such a line is altered in one cache, then both main memory and the other cache have an invalid version of that line.

Cache coherence protocols are designed to cope with this problem.

◆ When more than one processor are implemented on a single chip, the configuration

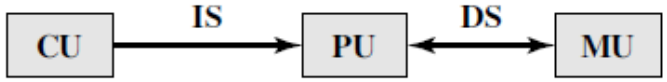
is referred to as **chip multiprocessing**. A related design scheme is to replicate some of the components of a single processor so that the processor can execute multiple threads concurrently; this is known as a **multithreaded processor**.

◆ A **cluster** is a group of interconnected, whole computers working together as a unified computing resource that can create the illusion of being one machine.The term *whole computer* means a system that can run on its own, apart from the cluster.

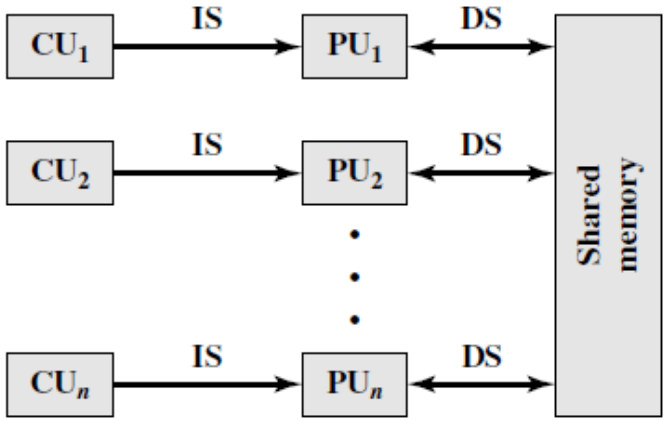
◆ A NUMA system is a shared-memory multiprocessor in which the access time from a given processor to a word in memory varies with the location of the memory word.

◆ A special-purpose type of parallel organization is the vector facility, which is tailored to the processing of vectors or arrays of data.

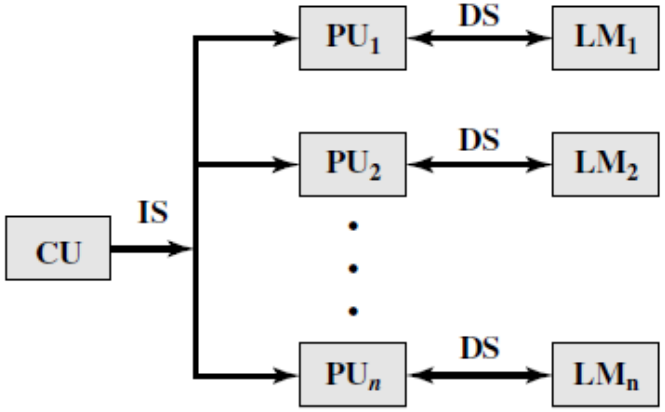
Alternative Computer Organizations



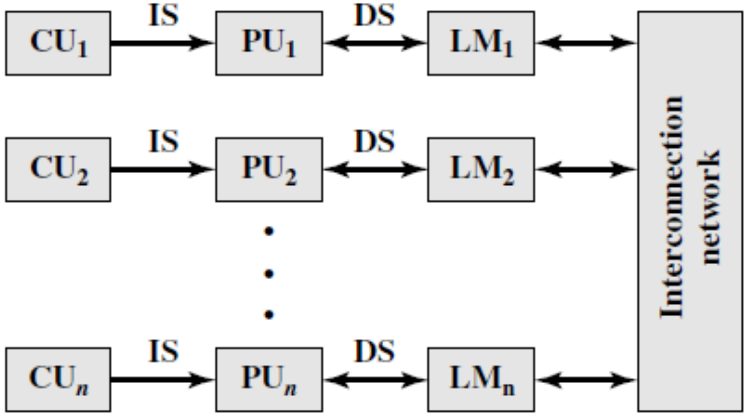
(a) SISD



(c) MIMD (with shared memory)



(b) SIMD (with distributed memory)



(d) MIMD (with distributed memory)

CU = Control unit
IS = Instruction stream
PU = Processing unit
DS = Data stream
MU = Memory unit
LM = Local memory

SISD = Single instruction, single data stream
SIMD = Single instruction, multiple data stream
MIMD = Multiple instruction, multiple data stream

An SMP(SYMMETRIC MULTIPROCESSORS)can be defined as a standalone computer system with the following characteristics:

1. There are two or more similar processors of comparable capability.

2. These processors share the same main memory and I/O facilities and are interconnected by a bus or other internal connection scheme, such that memory access time is approximately the same for each processor.

3. All processors share access to I/O devices, either through the same channels or through different channels that provide paths to the same device.

4. All processors can perform the same functions (hence the term *symmetric*).

5. The system is controlled by an integrated operating system that provides interaction between processors and their programs at the job, task, file, and data element levels.

An SMP organization has a number of potential advantages over a uniprocessor organization, including the following:

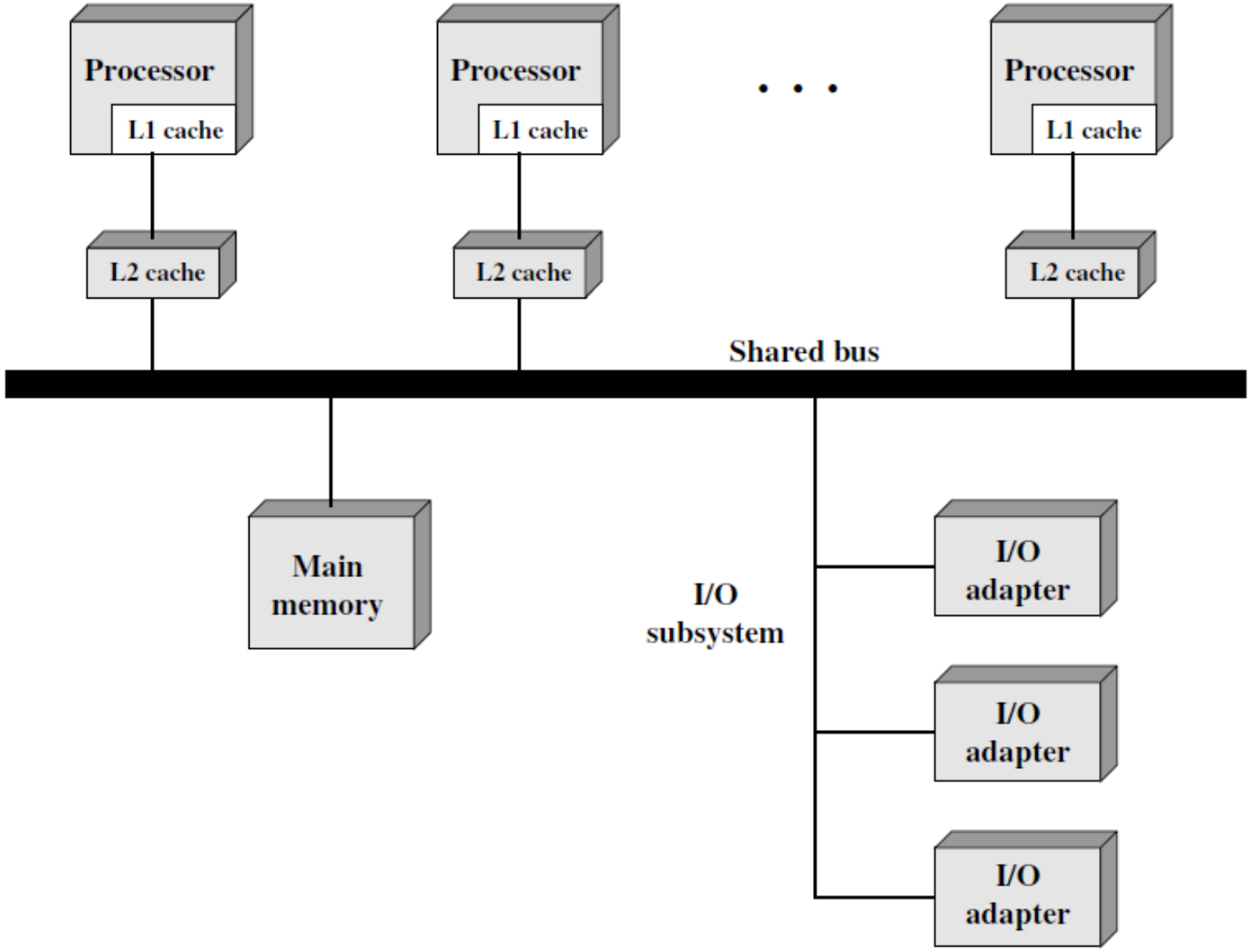
• **Performance:** If the work to be done by a computer can be organized so that some portions of the work can be done in parallel, then a system with multiple processors will yield greater performance than one with a single processor of the same type.

• **Availability:** In a symmetric multiprocessor, because all processors can perform the same functions, the failure of a single processor does not halt the machine. Instead, the system can continue to function at reduced performance.

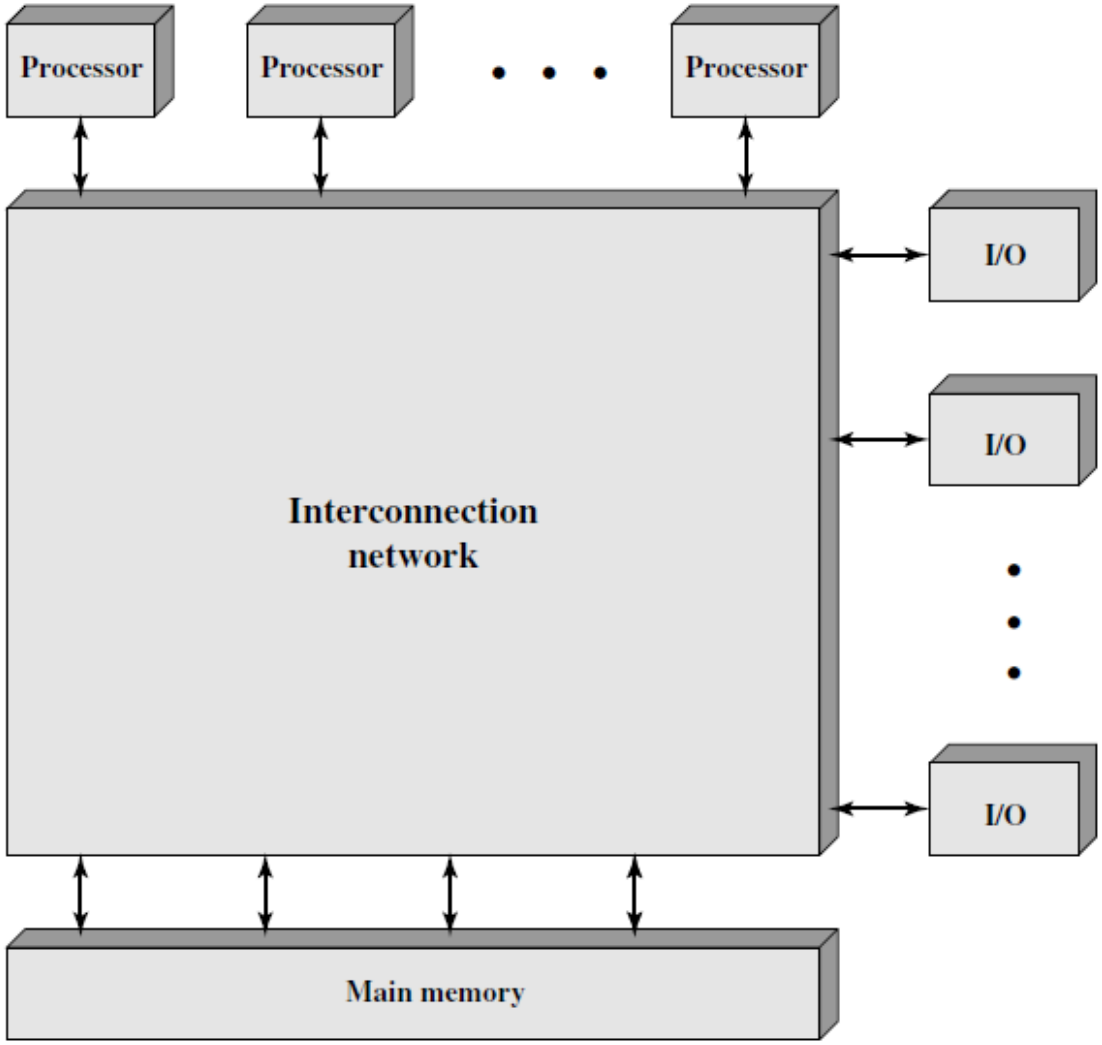
• **Incremental growth:** A user can enhance the performance of a system by adding an additional processor.

• **Scaling:** Vendors can offer a range of products with different price and performance characteristics based on the number of processors configured in the system.

Symmetric Multiprocessor Organization



Tightly Coupled Multiprocessor



A multiprocessor operating system must provide all the functionality of a multiprogramming system plus additional features to accommodate multiple processors. Among the key design issues:

- **Simultaneous concurrent processes:** OS routines need to be reentrant to allow several processors to execute the same IS code simultaneously. With multiple processors executing the same or different parts of the OS, OS tables and management structures must be managed properly to avoid deadlock or invalid operations.
- **Scheduling:** Any processor may perform scheduling, so conflicts must be avoided. The scheduler must assign ready processes to available processors.
- **Synchronization:** With multiple active processes having potential access to shared address spaces or shared I/O resources, care must be taken to provide effective synchronization. Synchronization is a facility that enforces mutual exclusion and event ordering.
- **Memory management:** Memory management on a multiprocessor must deal with all of the issues found on uniprocessor machines, the operating system needs to exploit the available hardware parallelism, such as multiported memories, to achieve the best performance.

The paging mechanisms on different processors must be coordinated to enforce consistency when several processors share a page or segment and to decide on page replacement.

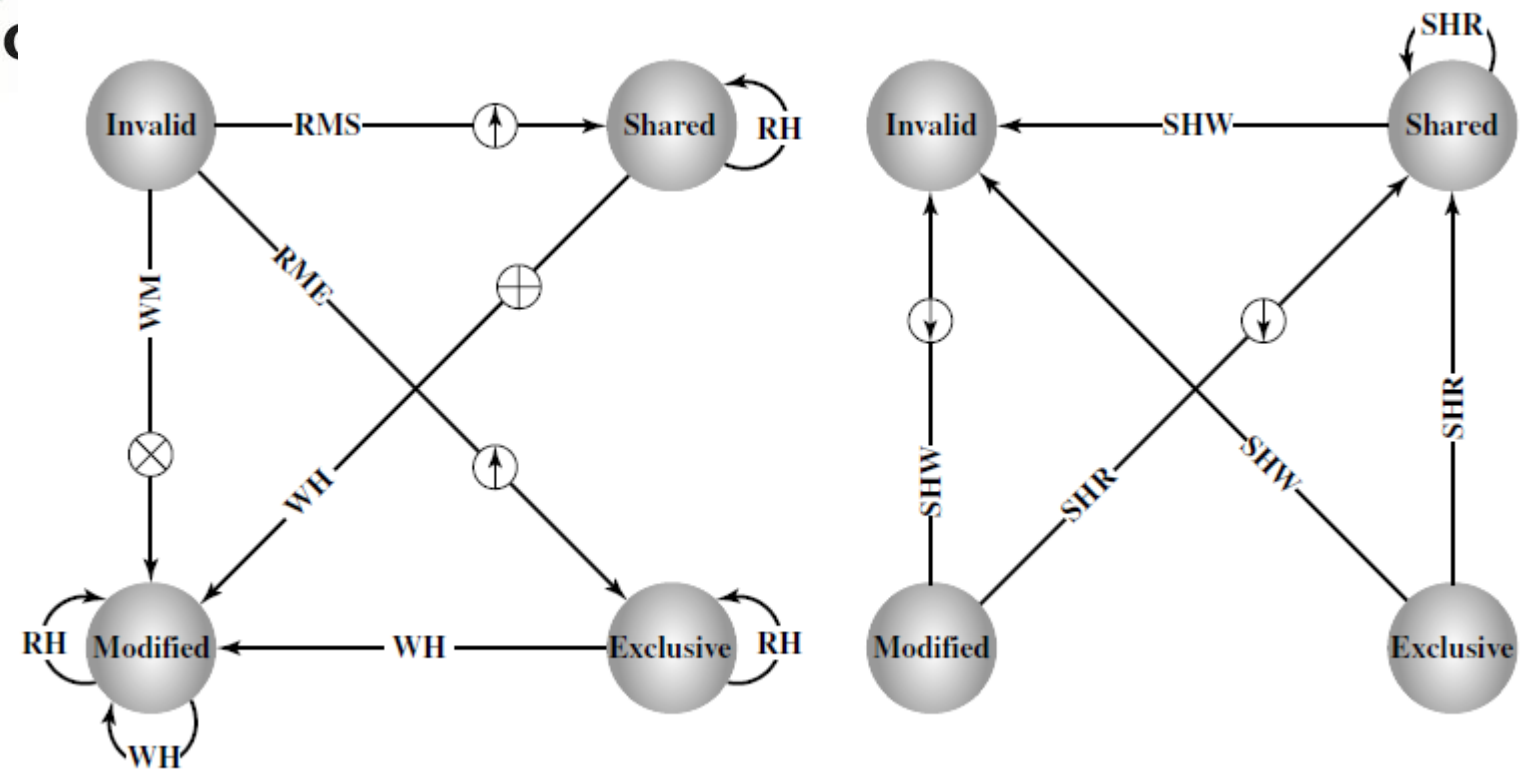
- **Reliability and fault tolerance:** The operating system should provide graceful degradation in the face of processor failure.The scheduler and other portions of the operating system must recognize the loss of a processor and restructure management tables accordingly.

	M Modified	E Exclusive	S Shared	I Invalid
This cache line valid?	Yes	Yes	Yes	No
The memory copy is . . .	out of date	valid	valid	—
Copies exist in other caches?	No	No	Maybe	Maybe
A write to this line . . .	does not go to bus	does not go to bus	goes to bus and updates cache	goes directly to bus

To provide cache consistency on an SMP, the data cache often supports a protocol known as MESI. For MESI, the data cache includes two status bits per tag, so that each line can be in one of four states:

- **Modified:** The line in the cache has been modified (different from main memory) and is available only in this cache.
- **Exclusive:** The line in the cache is the same as that in main memory and is not present in any other cache.
- **Shared:** The line in the cache is the same as that in main memory and may be present in another cache.
- **Invalid:** The line in the cache does not contain valid data.

MESI State Transition Diagram



(a) Line in cache at initiating processor

(b) Line in snooping cache

RH	Read hit
RMS	Read miss, shared
RME	Read miss, exclusive
WH	Write hit
WM	Write miss
SHR	Snoop hit on read
SHW	Snoop hit on write or read-with-intent-to-modify

Dirty line copyback

Invalidate transaction

Read-with-intent-to-modify

Cache line fill

• **Process:** An instance of a program running on a computer. A process embodies two key characteristics:
—**Resource ownership:** A process includes a virtual address space to hold the process image; the process image is the collection of program, data, stack, and attributes that define the process. From time to time, a process may be allocated control or ownership of resources, such as main memory, I/O channels, I/O devices, and files.
—**Scheduling/execution:** The execution of a process follows an execution path (trace) through one or more programs. This execution may be interleaved with that of other processes. Thus, a process has an execution state (Running, Ready, etc.) and a dispatching priority and is the entity that is scheduled and dispatched by the operating system.
• **Process switch:** An operation that switches the processor from one process to another, by saving all the process control data, registers, and other information for the first and replacing them with the process information for the second.2
• **Thread:** A dispatchable unit of work within a process. It includes a processor context (which includes the program counter and stack pointer) and its own data area for a stack (to enable subroutine branching). A thread executes sequentially and is interruptible so that the processor can turn to another thread.
• **Thread switch:** The act of switching processor control from one thread to another within the same process.Typically, this type of switch is much less costly than a process switch.

Thus, a thread is concerned with scheduling and execution, whereas a process is concerned with both scheduling/execution and resource ownership.

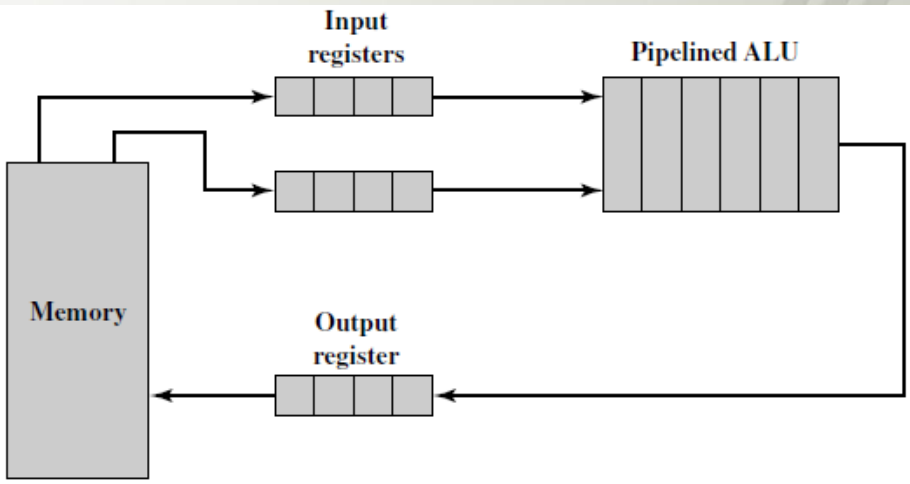
Benefits that can be achieved with clustering:

- **Absolute scalability:** It is possible to create large clusters that far surpass the power of even the largest standalone machines. A cluster can have tens, hundreds, or even thousands of machines, each of which is a multiprocessor.
- **Incremental scalability:** A cluster is configured in such a way that it is possible to add new systems to the cluster in small increments.Thus, a user can start out with a modest system and expand it as needs grow, without having to go through a major upgrade in which an existing small system is replaced with a larger system.
- **High availability:** Because each node in a cluster is a standalone computer, the failure of one node does not mean loss of service. In many products, fault tolerance is handled automatically in software.
- **Superior price/performance:** By using commodity building blocks, it is possible to put together a cluster with equal or greater computing power than a single large machine, at much lower cost.

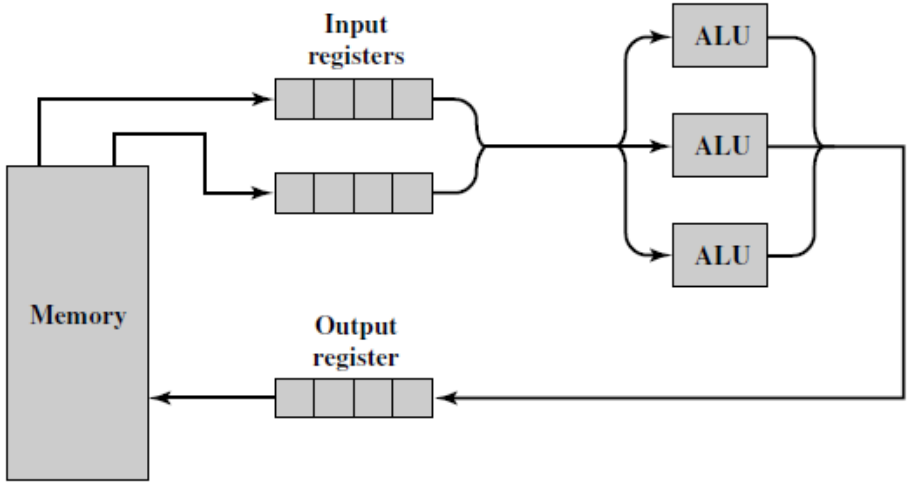
Clustering Methods: Benefits and Limitations

Clustering Method	Description	Benefits	Limitations
Passive Standby	A secondary server takes over in case of primary server failure.	Easy to implement.	High cost because the secondary server is unavailable for other processing tasks.
Active Secondary:	The secondary server is also used for processing tasks.	Reduced cost because secondary servers can be used for processing.	Increased complexity.
Separate Servers	Separate servers have their own disks. Data is continuously copied from primary to secondary server.	High availability.	High network and server overhead due to copying operations.
Servers Connected to Disks	Servers are cabled to the same disks, but each server owns its disks. If one server fails, its disks are taken over by the other server.	Reduced network and server overhead due to elimination of copying operations.	Usually requires disk mirroring or RAID technology to compensate for risk of disk failure.
Servers Share Disks	Multiple servers simultaneously share access to disks.	Low network and server overhead. Reduced risk of downtime caused by disk failure.	Requires lock manager software. Usually used with disk mirroring or RAID technology.

Approaches to Vector Computation

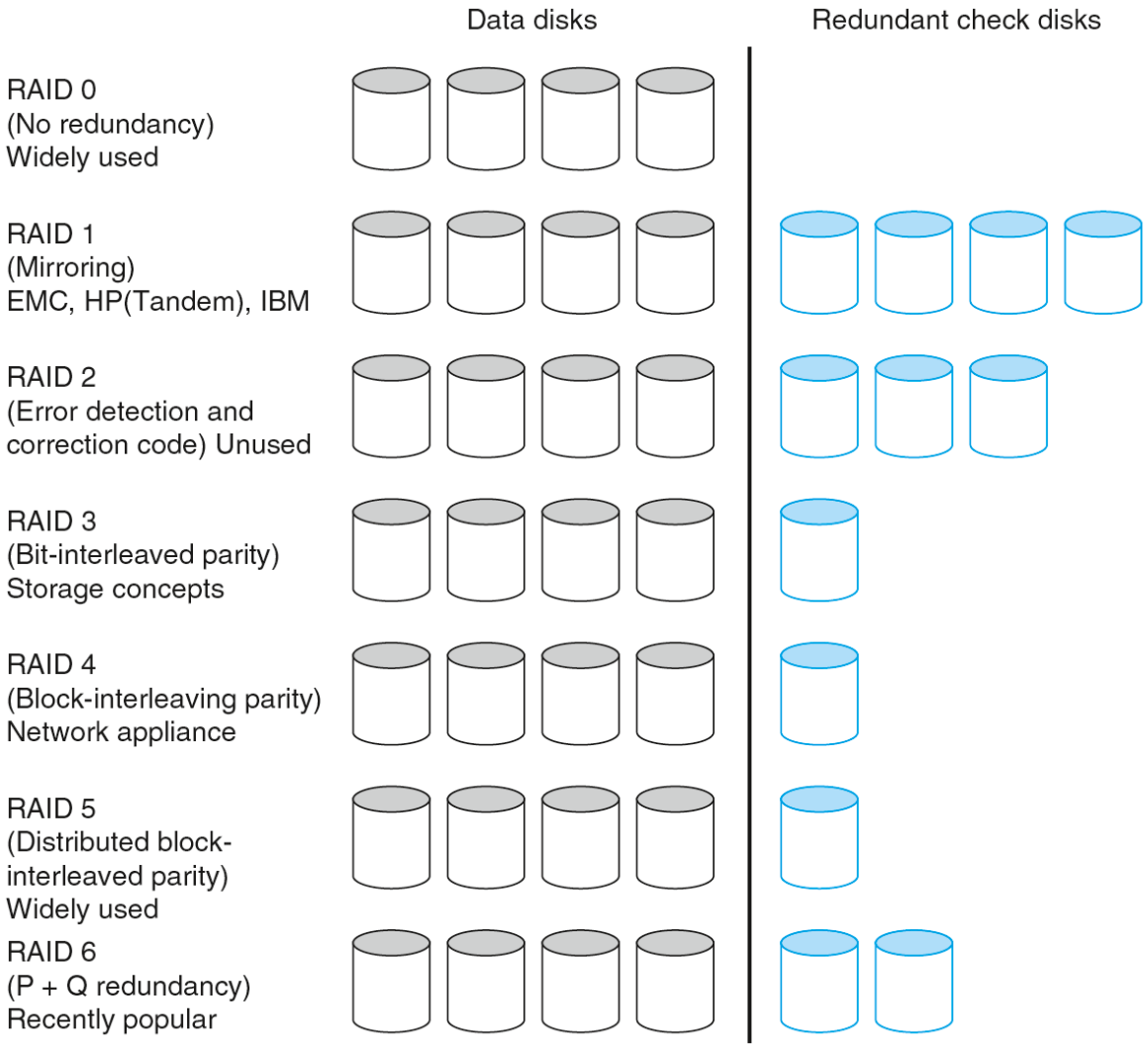


(a) Pipelined ALU



(b) Parallel ALUs

RAID for an example of four data disks showing extra check disks per RAID level and companies that use each level.



redundant arrays of inexpensive disks (RAID)
An organization of disks that uses an array of small and inexpensive disks so as to increase both performance and reliability.

No Redundancy (RAID 0) - striping
Allocation of logically sequential blocks to separate disks to allow higher performance than a single disk can deliver.

Mirroring (RAID 1)
Writing identical data to multiple disks to increase data availability.

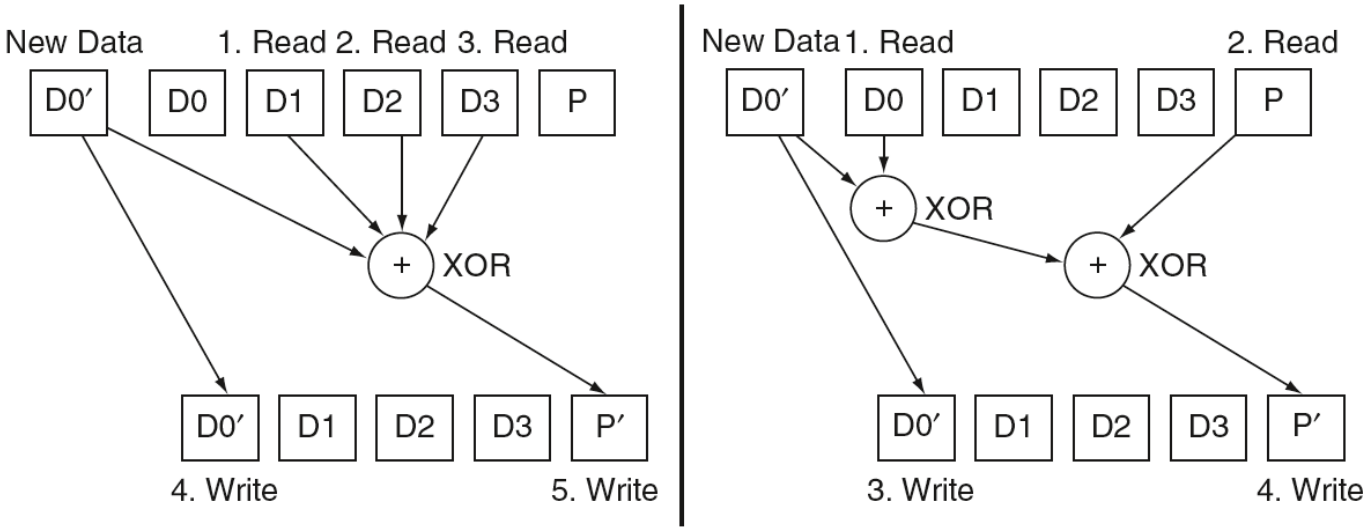
Error Detecting and Correcting Code (RAID 2)
RAID 2 borrows an error detection and correction scheme most often used for memories. Since RAID 2 has fallen into disuse, we'll not describe it here.

Bit-Interleaved Parity (RAID 3)
The cost of higher availability can be reduced to 1/n, where n is the number of disks in a protection group. Rather than have a complete copy of the original data for each disk, we need only add enough redundant information to restore the lost information on a failure. Reads or writes go to all disks in the group, with one extra disk to hold the check information in case there is a failure. RAID 3 is popular in applications with large data sets, such as multimedia and some scientific codes.

protection group
The group of data disks or blocks that share a common check disk or block.

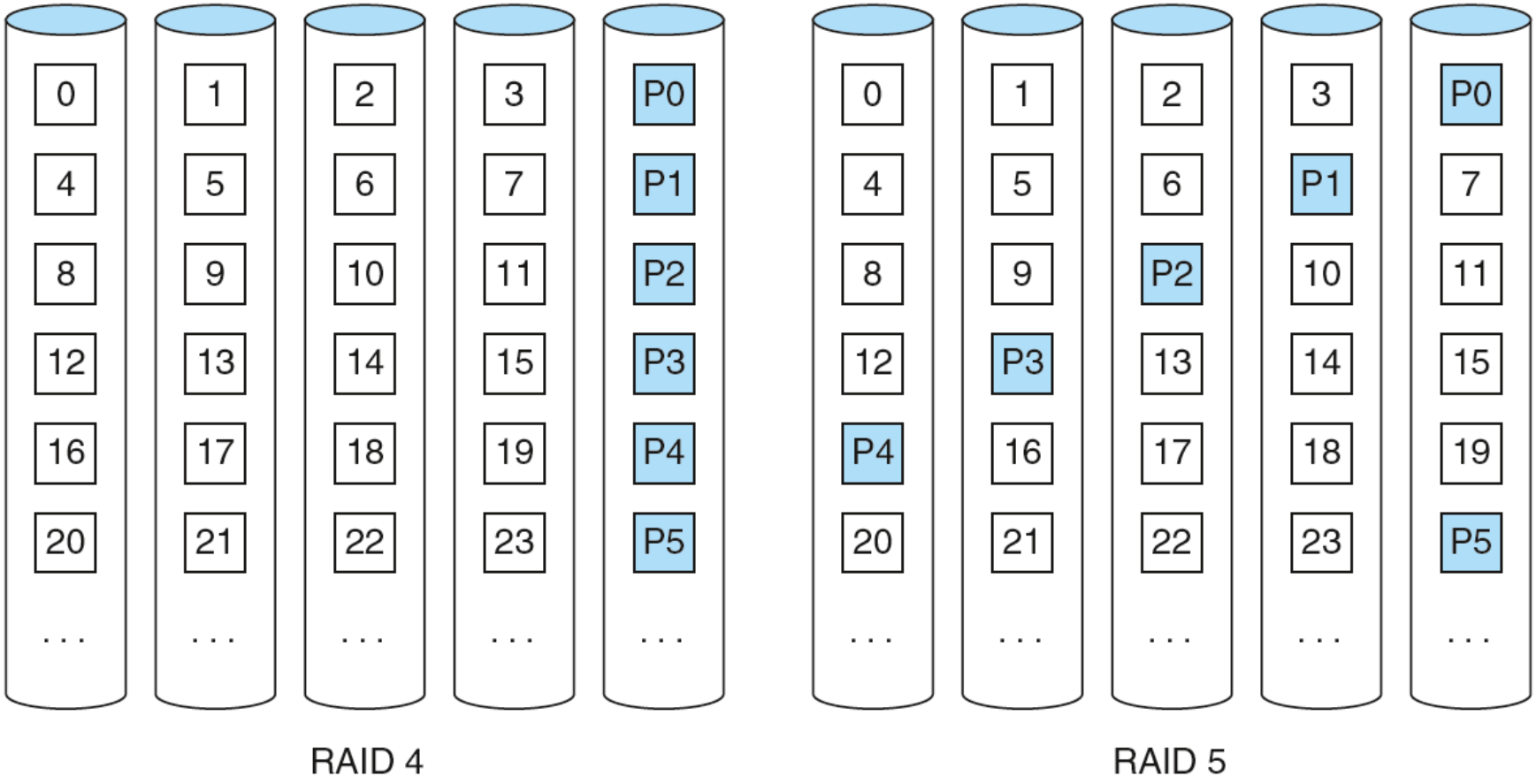
Block-Interleaved Parity (RAID 4)

RAID 4 uses the same ratio of data disks and check disks as RAID 3, but they access data differently. The parity is stored as blocks and associated with a set of data blocks.



Small write update on RAID 4. This optimization for small writes reduces the number of disk accesses as well as the number of disks occupied. This figure assumes we have four blocks of data and one block of parity. The naive RAID 4 parity calculation in the left of the figure reads blocks D1, D2, and D3 before adding block D0? to calculate the new parity P?. (In case you were wondering, the new data D0? comes directly from the CPU, so disks are not involved in reading it.) The RAID 4 shortcut on the right reads the old value D0 and compares it to the new value D0? to see which bits will change. You then read the old parity P and then change the corresponding bits to form P?. The logical function exclusive OR does exactly what we want. This example replaces three disk reads (D1, D2, D3) and two disk writes (D0?, P?) involving all the disks for two disk reads (D0, P) and two disk writes (D0?, P?), which involve just two disks. Increasing the size of the parity group increases the savings of the shortcut. RAID 5 uses the same shortcut.

Block-interleaved parity (RAID 4) versus distributed block-interleaved parity (RAID 5).
By distributing parity blocks to all disks, some small writes can be performed in parallel.



P + Q Redundancy (RAID 6)
Parity-based schemes protect against a single self-identifying failure. When a single failure correction is not sufficient, parity can be generalized to have a second calculation over the data and another check disk of information. This second check block allows recovery from a second failure. Thus, the storage overhead is twice that of RAID 5.

RAID Summary
RAID 1 and RAID 5 are widely used in servers; one estimate is that 80% of disks in servers are found in a RAID organization.
One weakness of the RAID systems is repair. First, to avoid making the data unavailable during repair, the array must be designed to allow the failed disks to be replaced without having to turn off the system.

hot-swapping
Replacing a hardware component while the system is running.
standby spares
Reserve hardware resources that can immediately take the place of a failed component.