Memory Management

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

summary

- goals and requirements
- techniques that do not involve virutal memory

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Memory Management goals

- Subdividing memory to accommodate multiple processes
- Memory needs to be allocated to ensure a reasonable supply of ready processes to consume available processor time

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

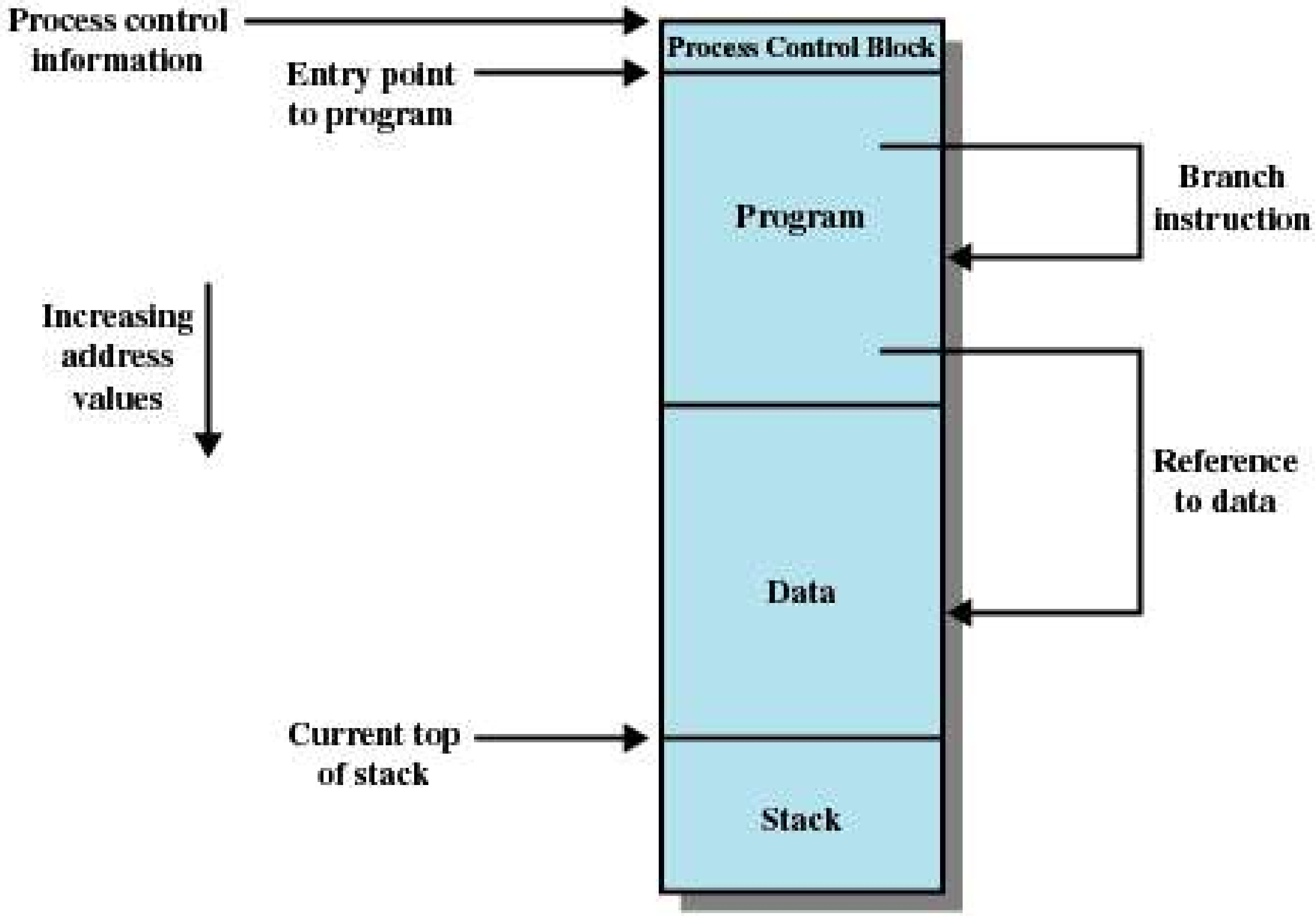
Memory Management Requirements

- Relocation
  - Programmer does not know where the program will be placed in memory when it is executed
  - While the program is executing, it may be swapped to disk and returned to main memory at a different location (relocated)
  - Memory references must be translated in the code to actual physical memory address

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Memory Management Requirements

- Relocation



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Memory Management Requirements

- Protection
  - Processes should not be able to reference memory locations in another process without permission
  - Impossible to check absolute addresses at compile time
  - Must be checked at run time
  - Memory protection requirement must be satisfied by the processor (hardware) rather than the operating system (software)
    - Operating system cannot anticipate all of the memory references a program will make

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Memory Management Requirements

- Sharing
  - Allow several processes to access the same portion of memory
  - Better to allow each process access to the same copy of the program rather than have their own separate copy

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Memory Management Requirements

- Logical Organization
  - Programs are written in modules
  - Modules can be written and compiled independently
  - Different degrees of protection given to modules (read-only, execute-only)
  - Share modules among processes

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Memory Management Requirements

- Physical Organization
  - Memory available for a program plus its data may be insufficient
    - Overlaying allows various modules to be assigned the same region of memory
  - Programmer does not know how much space will be available

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

techniques that do not involves virtual memory

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Fixed Partitioning

- Equal-size partitions
  – Any process whose size is less than or equal to the partition size can be loaded into an available partition
  – If all partitions are full, the operating system can swap a process out of a partition
  – A program may not fit in a partition. The programmer must design the program with overlays

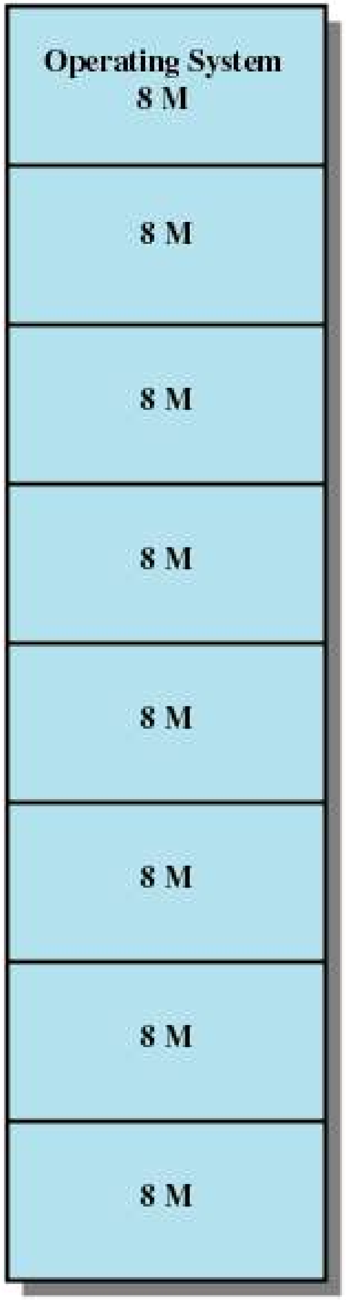
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Fixed Partitioning

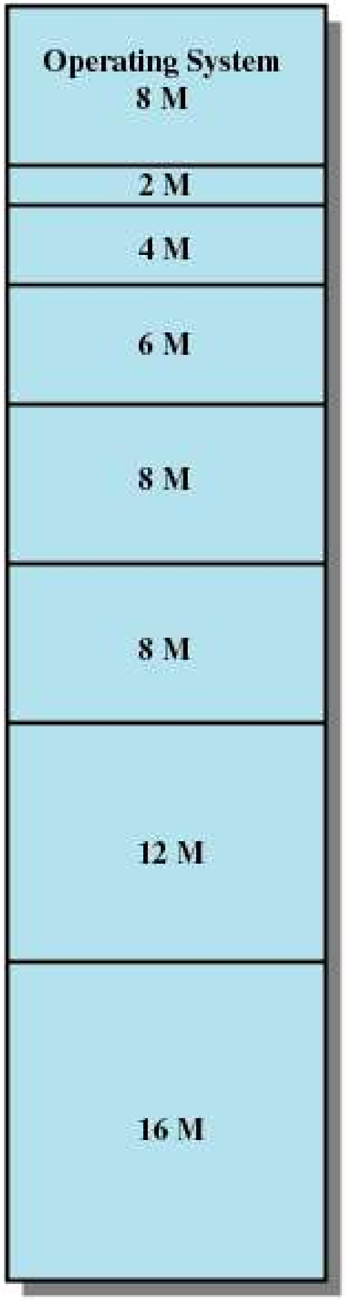
- Main memory use is inefficient. Any program, no matter how small, occupies an entire partition. This is called **internal fragmentation**.

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

partitions size



(a) Equal-size partitions



(b) Unequal-size partitions

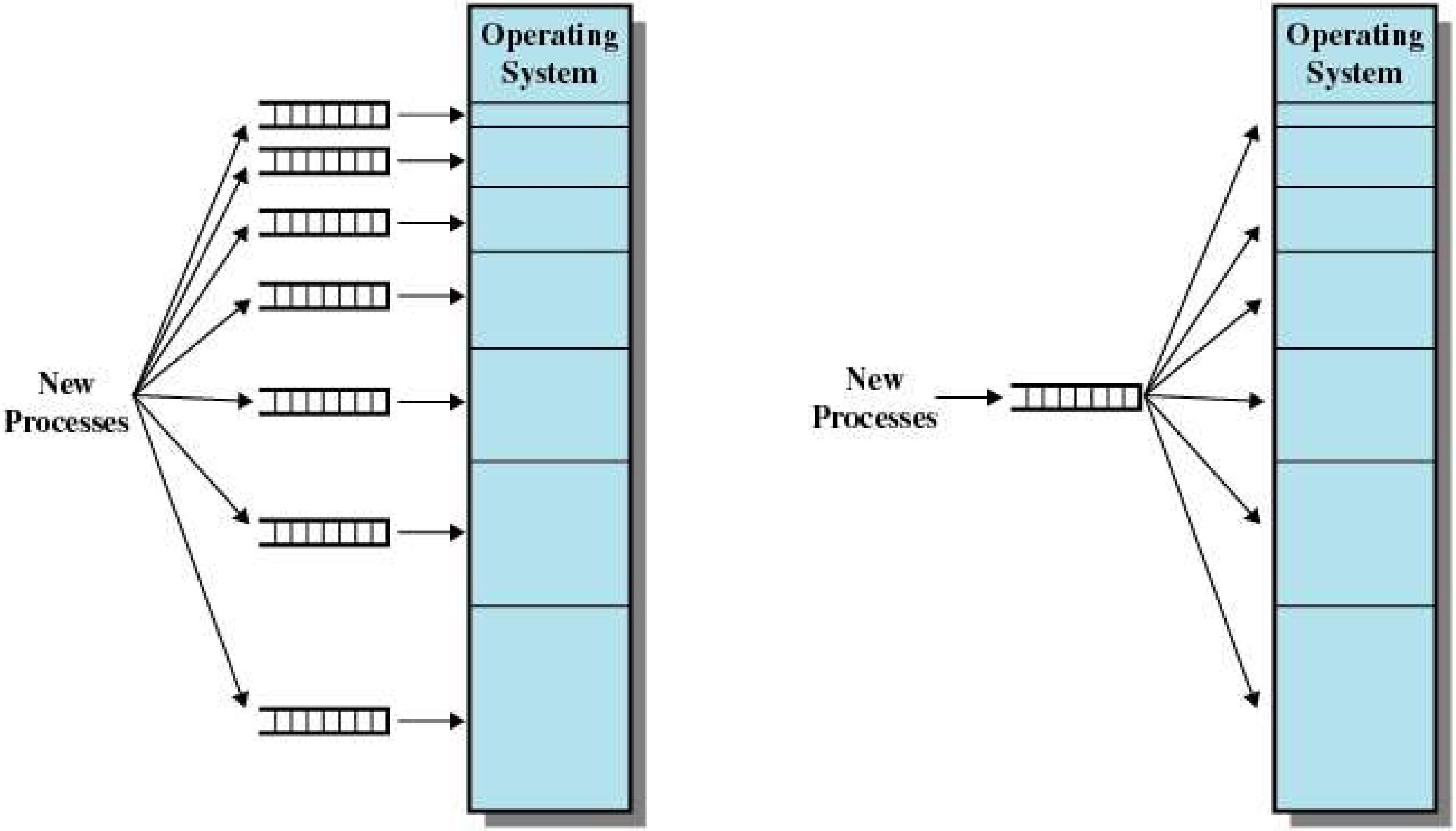
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Placement Algorithm with Partitions

- Equal-size partitions
  - Because all partitions are of equal size, it does not matter which partition is used
- Unequal-size partitions
  - Can assign each process to the smallest partition it will fit into
  - Queue for each partition
  - Processes are assigned in such a way as to minimize wasted memory within a partition

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Placement Algorithm with Partitions



(a) One process queue per partition

(b) Single queue

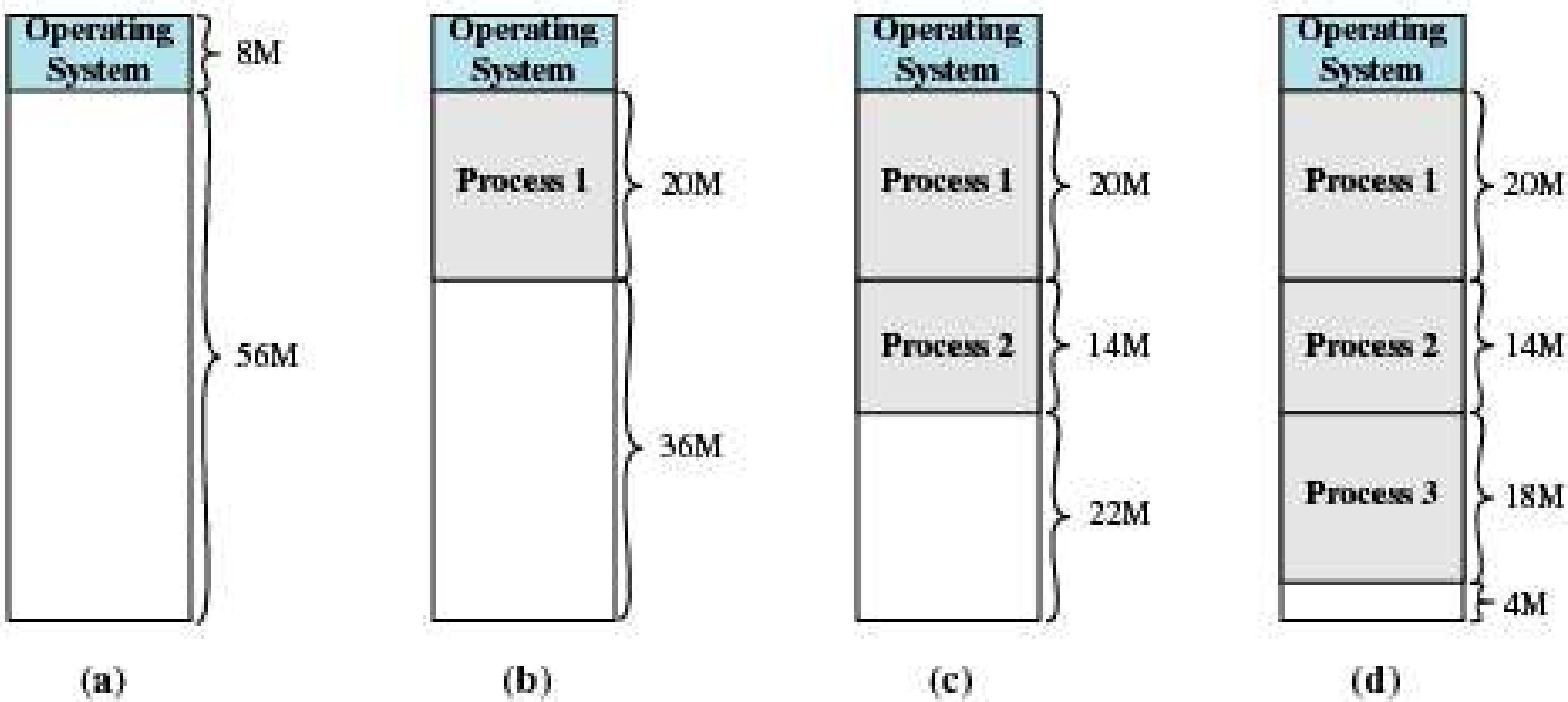
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Dynamic Partitioning

- Partitions are of variable length and number
- Process is allocated exactly as much memory as required
- Eventually get holes in the memory. This is called external fragmentation
- Must use compaction to shift processes so they are contiguous and all free memory is in one block

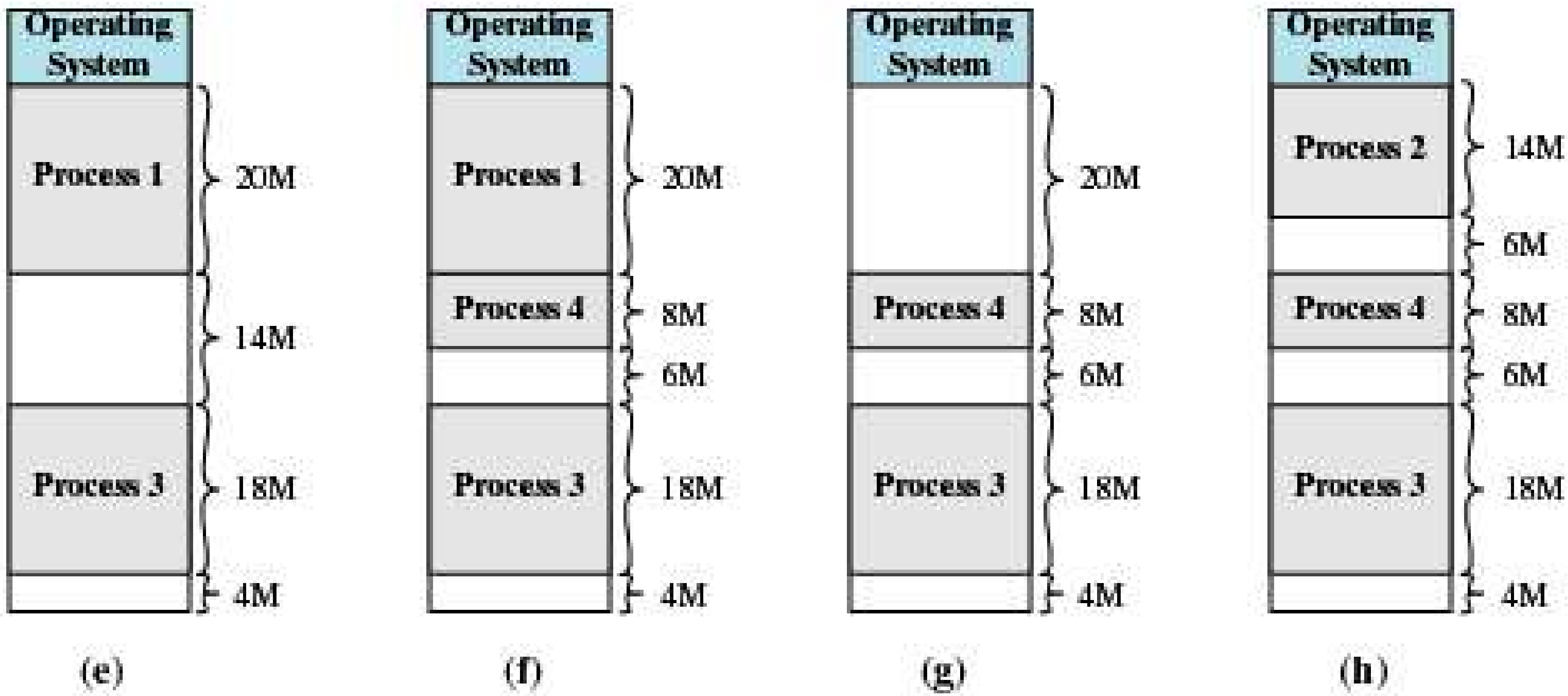
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

external fragmentation



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

external fragmentation



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Dynamic Partitioning Placement Algorithm

- Operating system must decide which free block to allocate to a process
- Best-fit algorithm
  - Chooses the block that is closest in size to the request
  - Worst performer overall
  - Since smallest block is found for process, the smallest amount of fragmentation is left
  - Memory compaction must be done more often

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Dynamic Partitioning Placement Algorithm

- First-fit algorithm
  - Scans memory form the beginning and chooses the first available block that is large enough
  - Fastest
  - May have many process loaded in the front end of memory that must be searched over when trying to find a free block

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

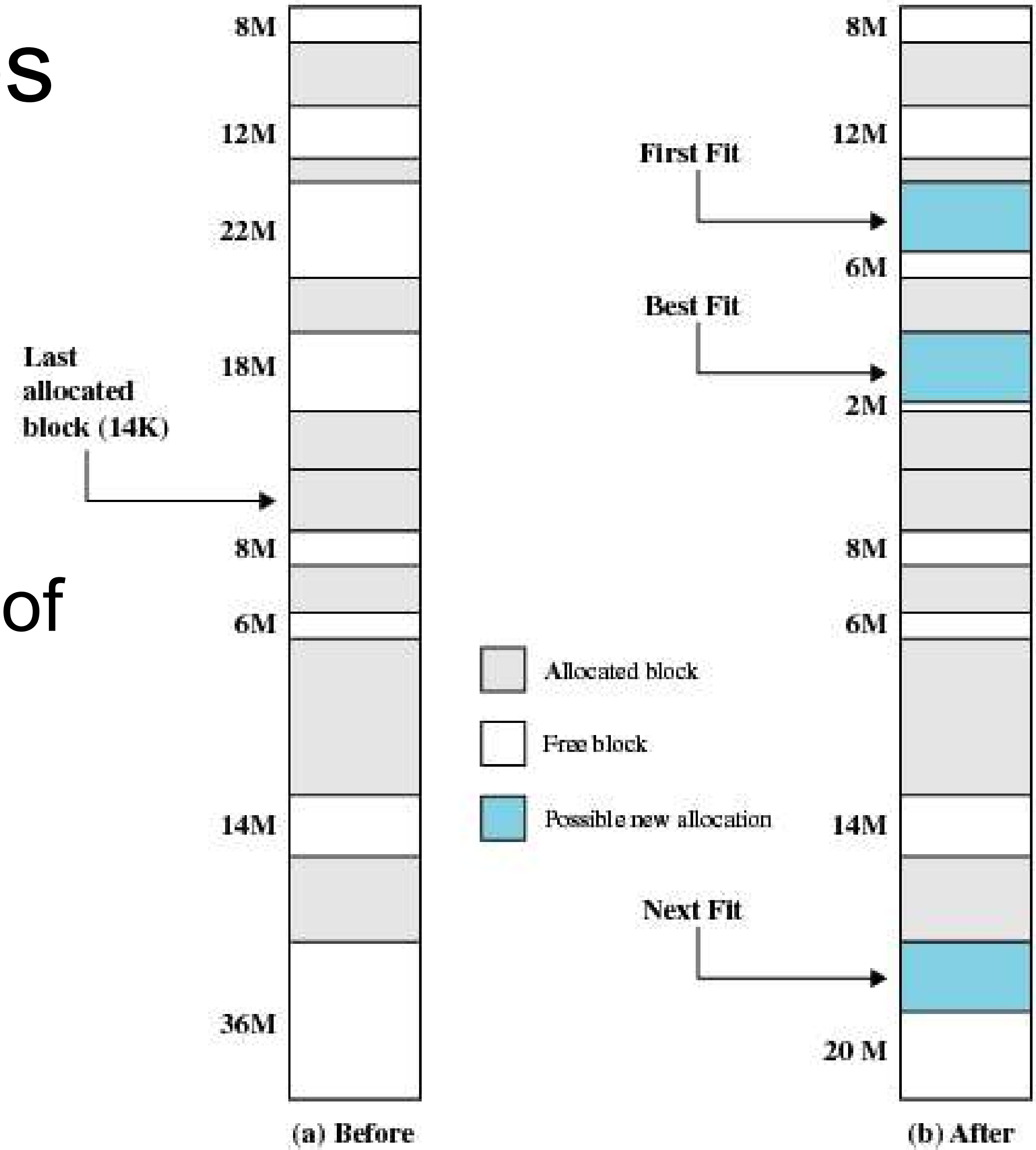
Dynamic Partitioning Placement Algorithm

- Next-fit
  - Scans memory from the location of the last placement
  - More often allocate a block of memory at the end of memory where the largest block is found
  - The largest block of memory is broken up into smaller blocks
  - Compaction is required to obtain a large block at the end of memory

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

examples

- allocation of a block of 16MB



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Buddy System

- Entire space available is treated as a single block of 2^U
- If a request of size s such that $2^{U-1} < s <= 2^U$, entire block is allocated
  - Otherwise block is split into two equal buddies
  - Process continues until smallest block greater than or equal to *s* is generated

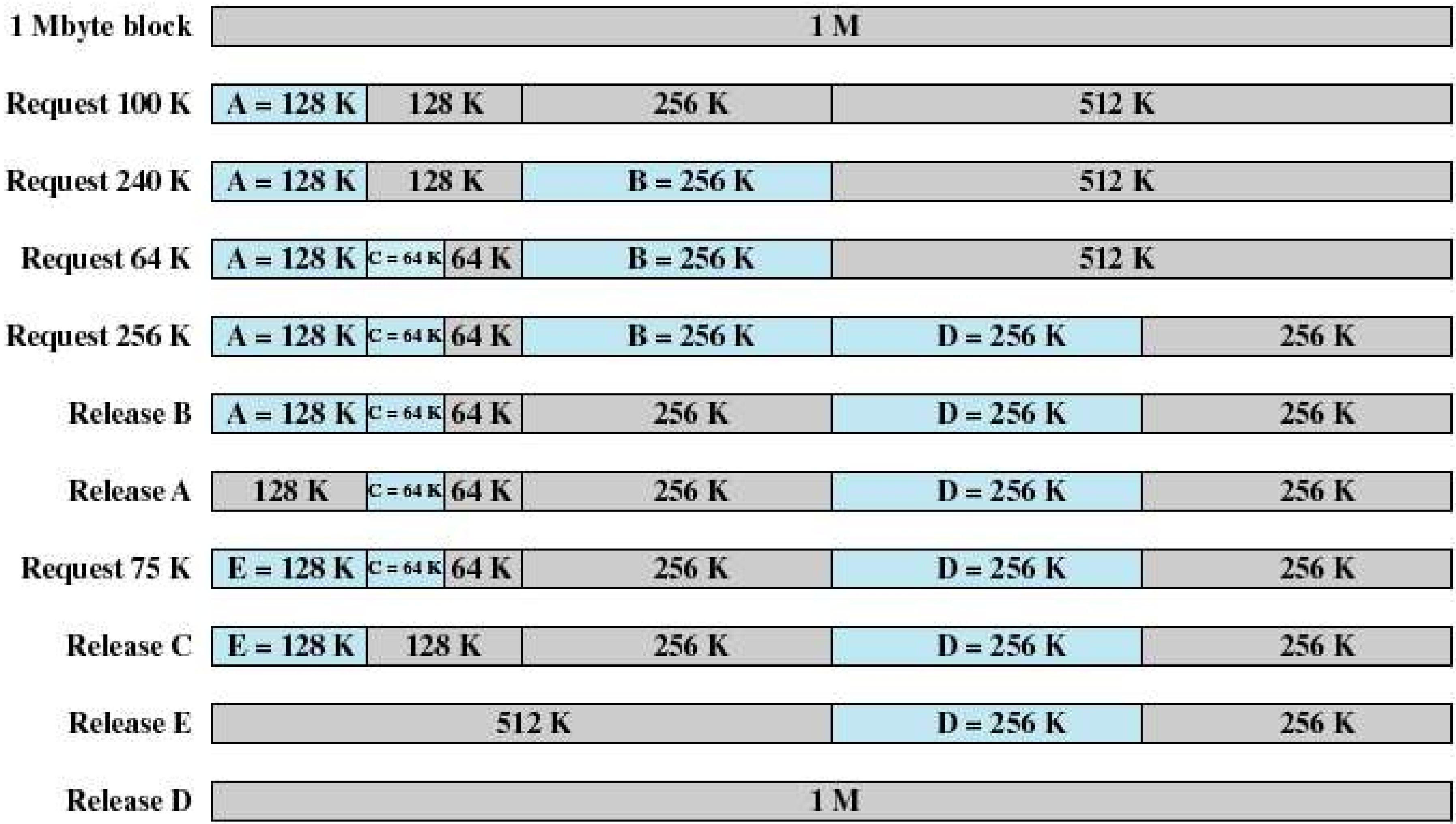
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Buddy System

- it maintains a lists L_i ($i=1..U$) of unallocated blocks (*holes*) of size 2^i
  - splitting: remove a hole from L_{i+1} split it, and put the two buddies it into L_i
  - coalescing: remove two unallocated buddies from L_i and put it into L_{i+1}
  - $2^{U-1} < s <= 2^U$, entire block is allocated

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

buddy system: example



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Buddy System

procedure **get_hole**

input: *i* (prec: $i \leq U$)

output: a block of size 2^i

if (L_i is empty)

b= get_hole(i+1);

< remove b from L_{i+1} >

< split b into two buddies b_1 and b_2>

< put b_1 and b_2 into L_i>

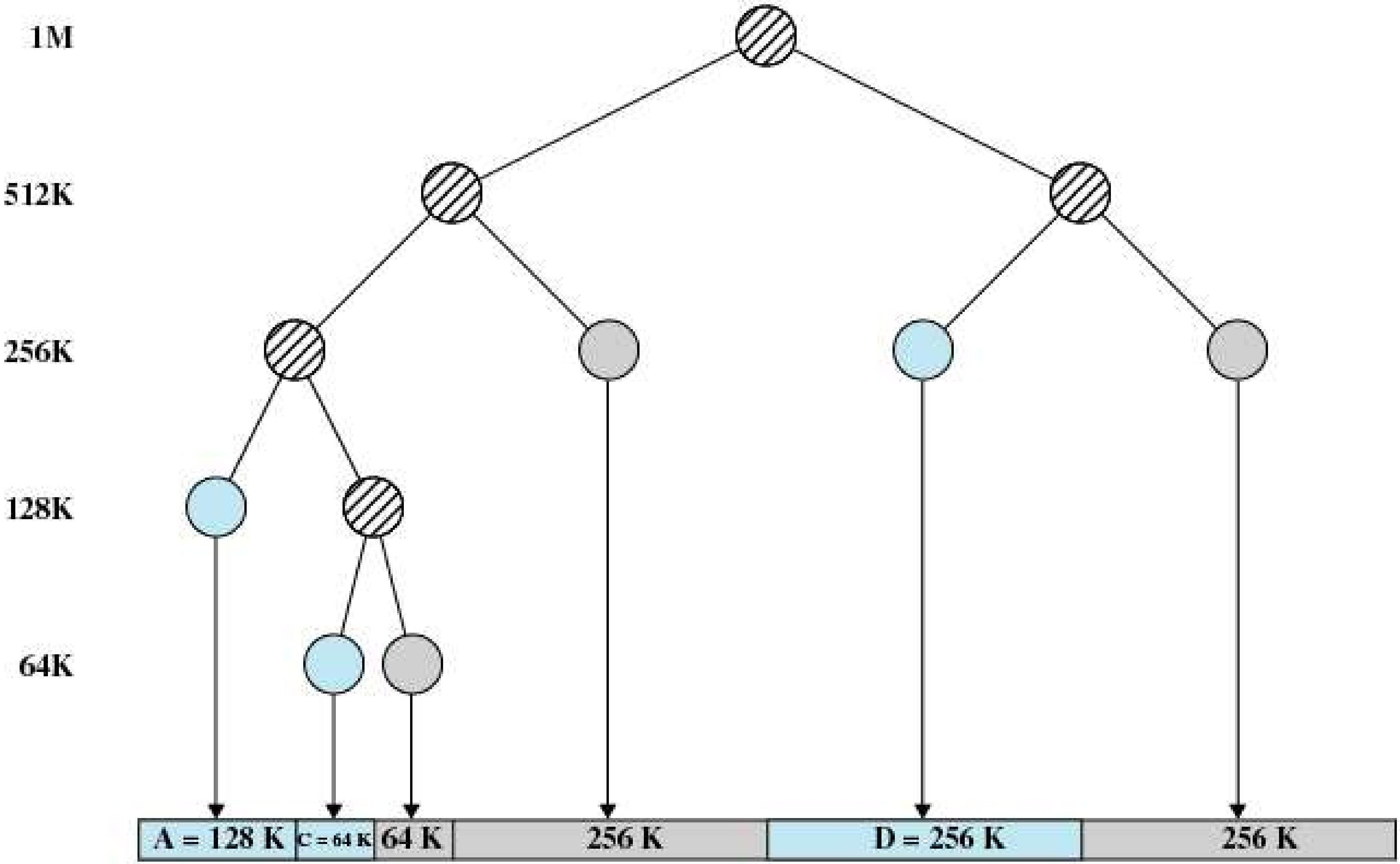
c= < first hole in L_i>

<remove c form L_i>

return c

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

buddy system: tree representation



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Relocation

- When program loaded into memory the actual (absolute) memory locations are determined
- A process may occupy different partitions which means different absolute memory locations during execution (from swapping)
- Compaction will also cause a program to occupy a different partition which means different absolute memory locations

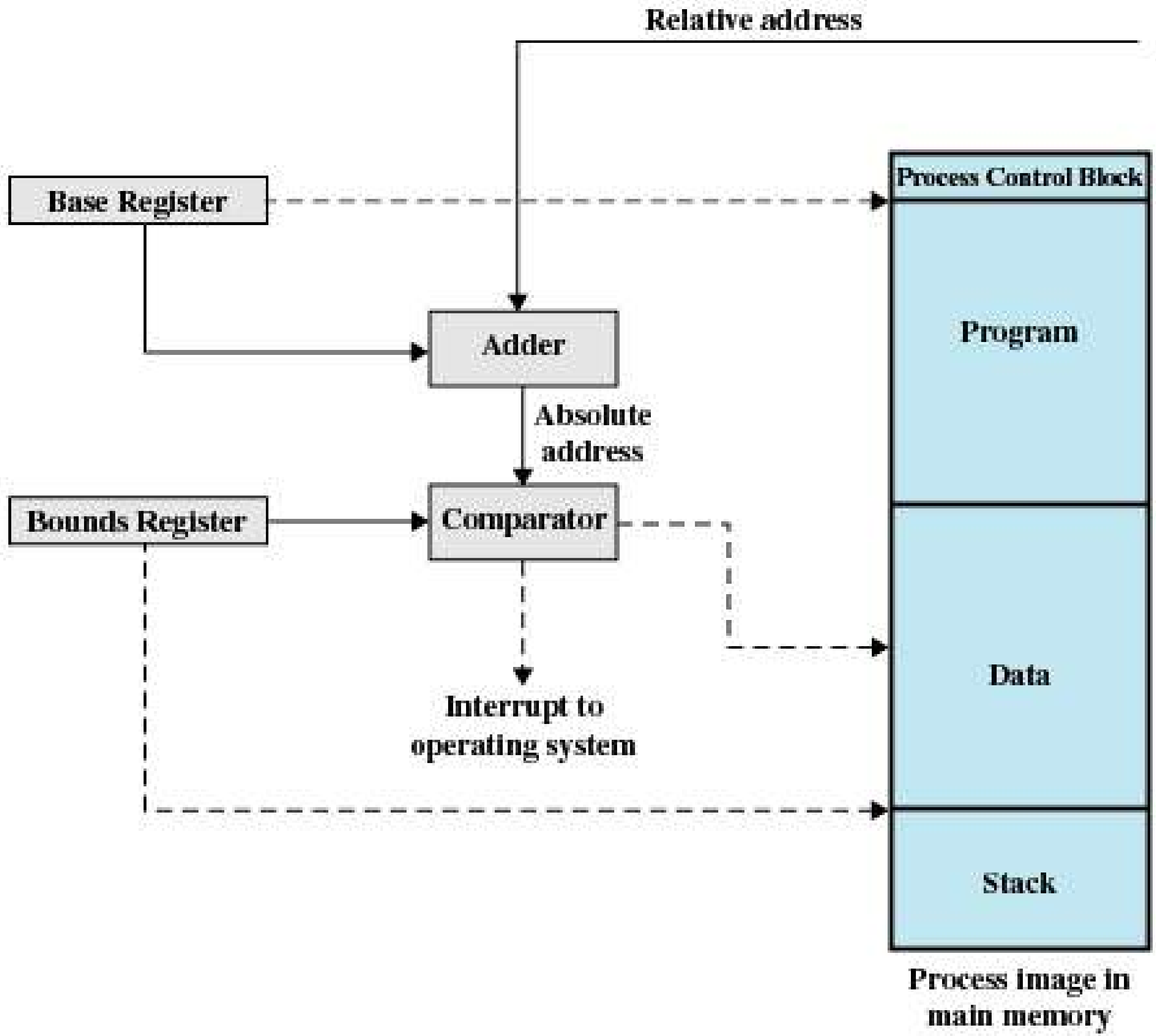
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Addresses

- Logical
  - Reference to a location in a “logical” memory independent of the current assignment of data to memory
  - Translation must be made to the physical address
- Physical
  - The absolute address or actual location in main memory
- Relative (logical or physical)
  - Address expressed as a location relative to some known point

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

hardware support for relocation



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Registers Used during Execution

- Base register
  - Starting address for the process
- Bounds register
  - Ending location of the process
- These values are set when the process is loaded or when the process is swapped in

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Registers Used during Execution

- The value of the base register is added to a relative address to produce an absolute address
- The resulting address is compared with the value in the bounds register
- If the address is not within bounds, an interrupt is generated to the operating system

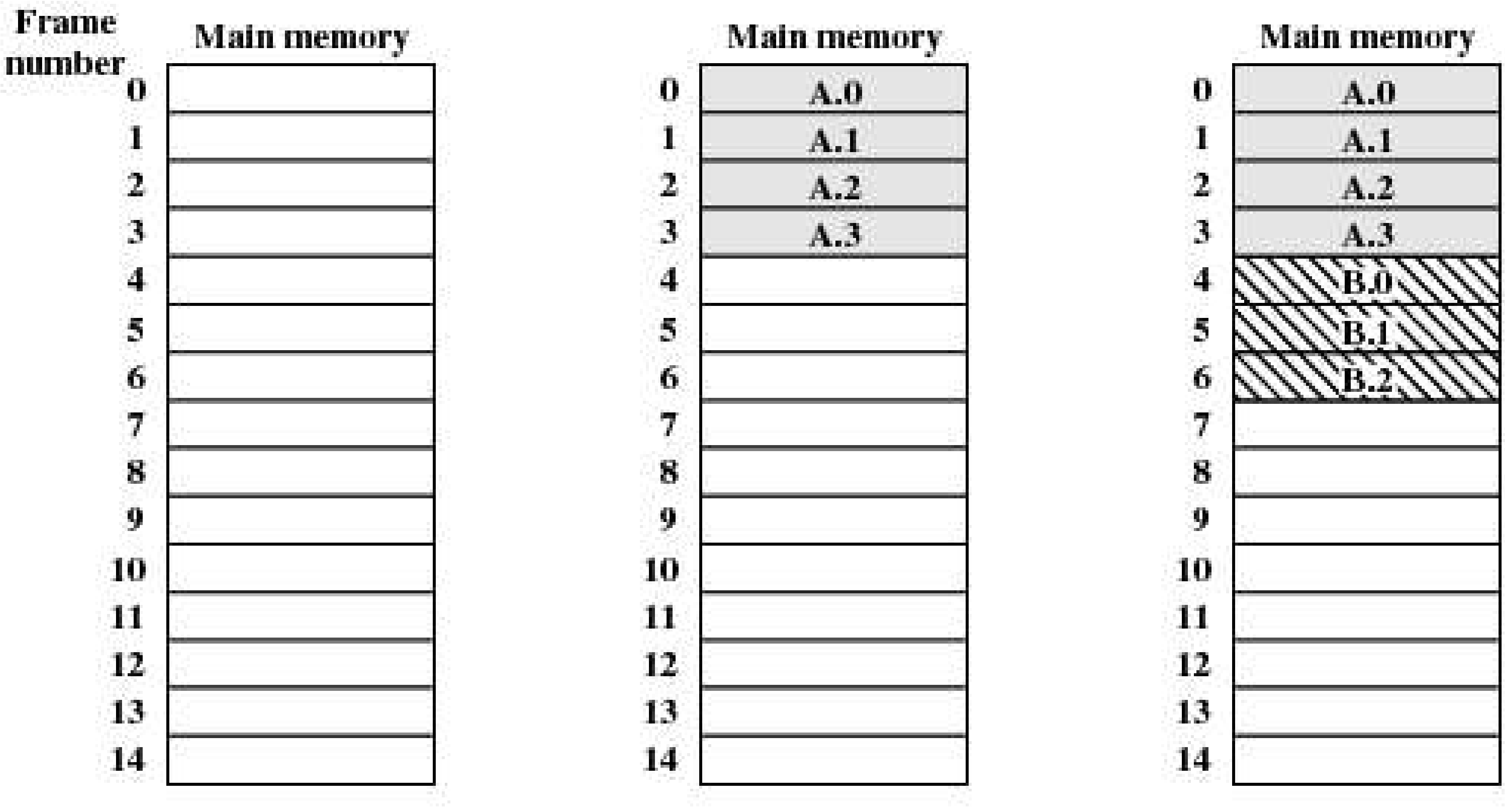
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Paging

- Partition memory into small equal fixed-size chunks and divide each process into the same size chunks
- The chunks of a process are called **pages** and chunks of memory are called **frames**
- Operating system maintains a **page table** for each process
  - Contains the **frame location for each page** in the process
  - **Memory address consist of a page number and offset within the page**

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Assignment of Process Pages to Free Frames



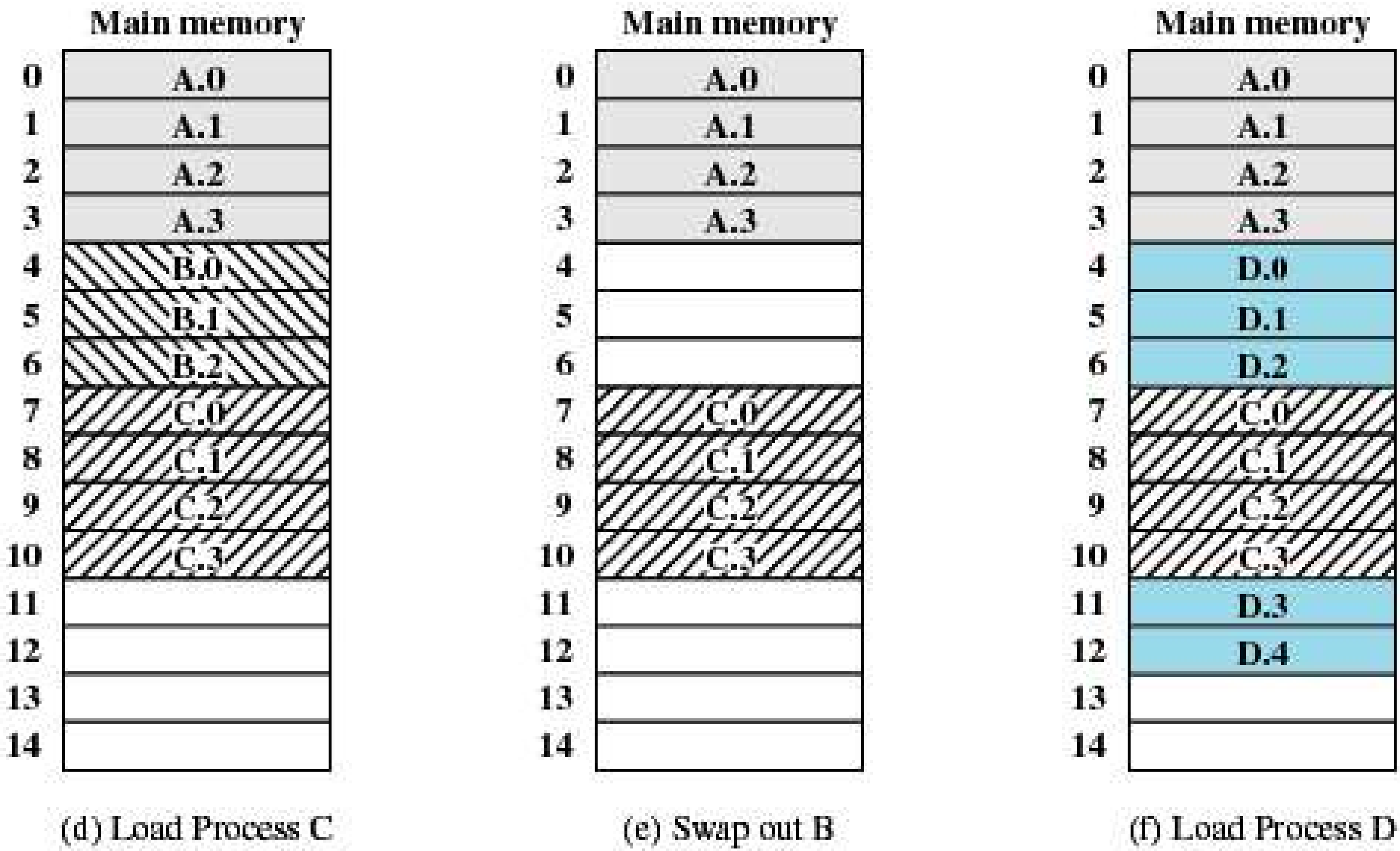
(a) Fifteen Available Frames

(b) Load Process A

(c) Load Process B

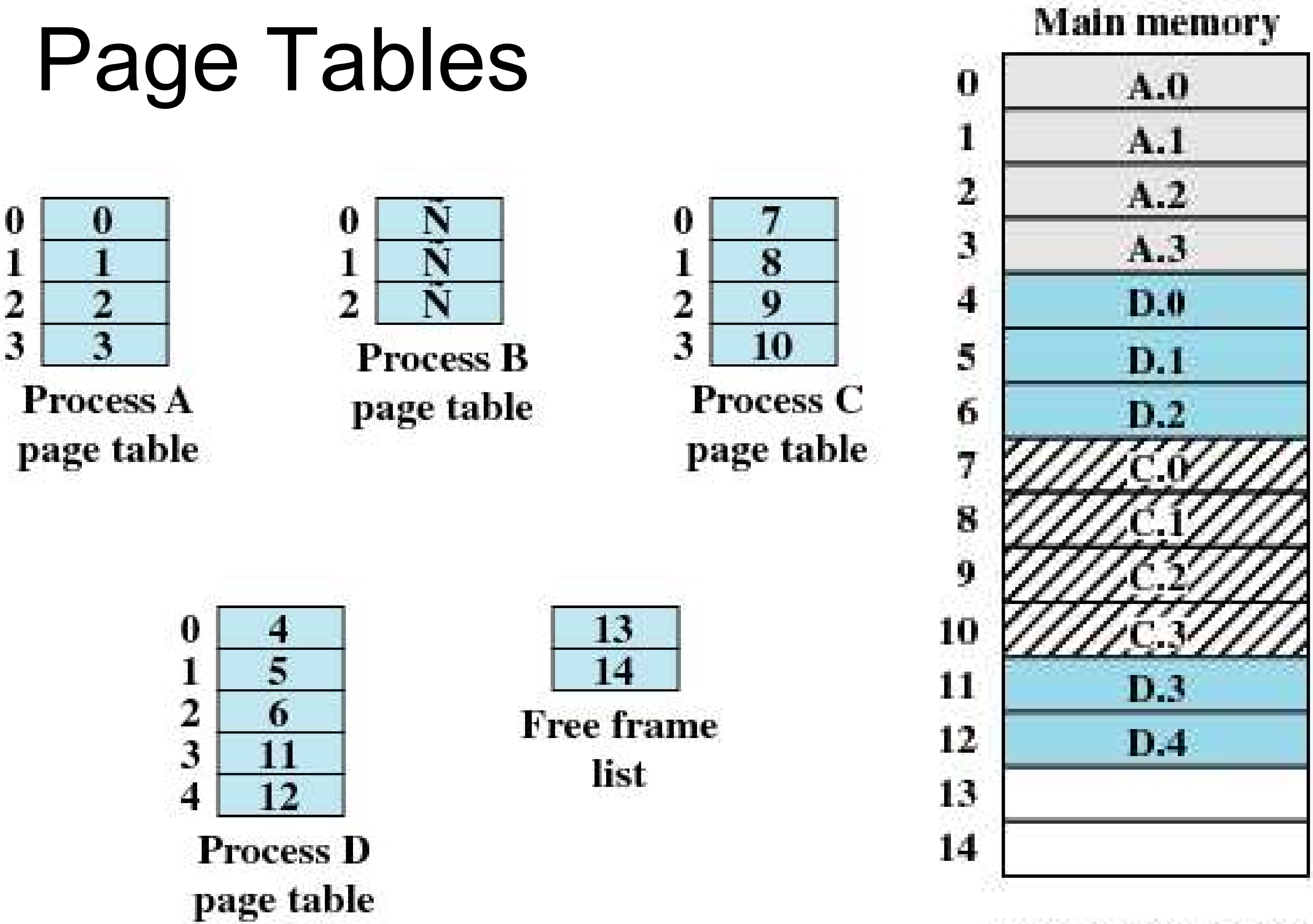
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Assignment of Process Pages to Free Frames



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Page Tables



(f) Load Process D

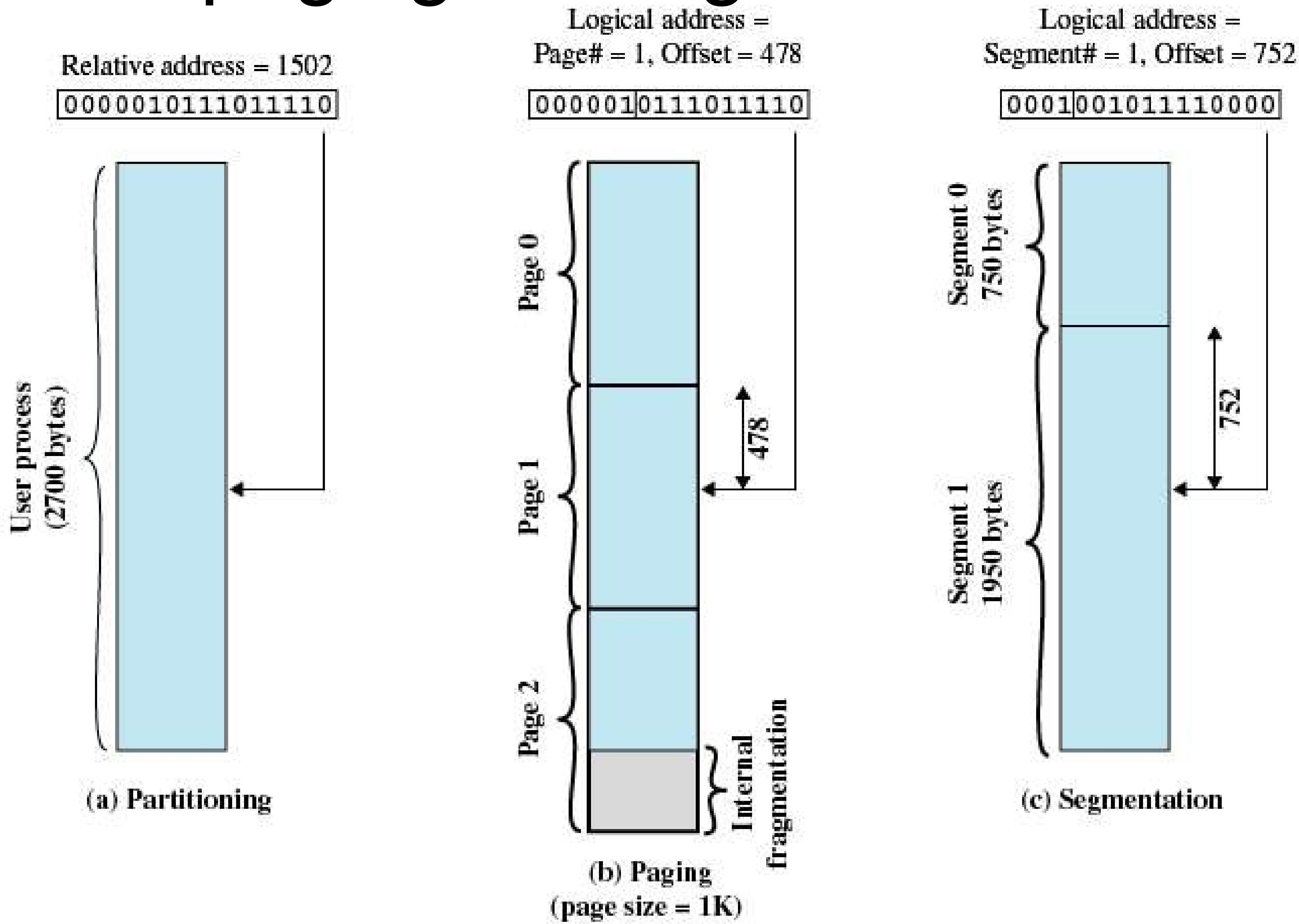
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

Segmentation

- All segments of all programs do **not** have to be of the **same length**
- There is a maximum segment length
- Addressing consist of two parts - a **segment number and an offset**
- Since segments are not equal, segmentation is similar to dynamic partitioning

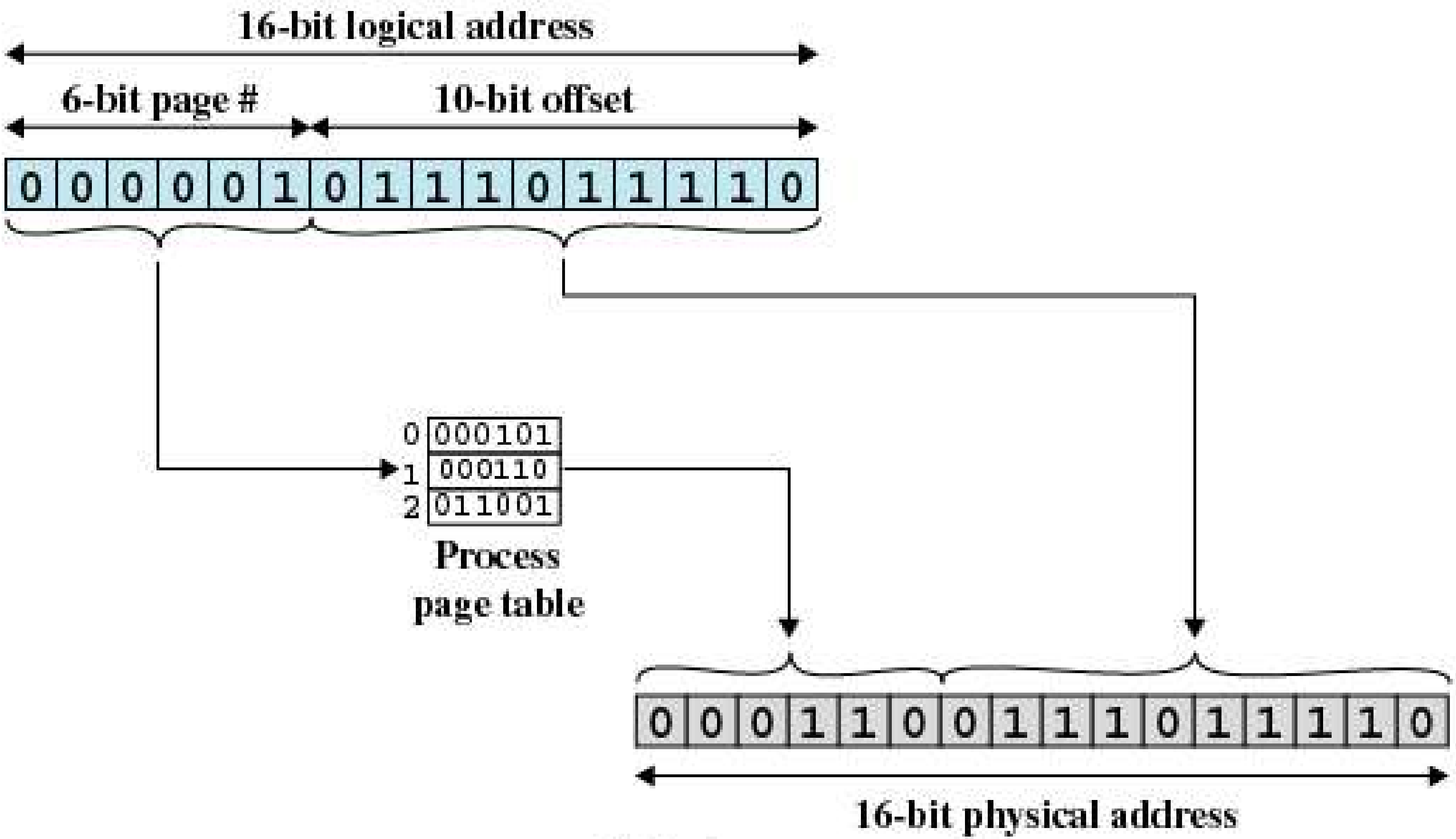
© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

paging vs. segmentation



© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

logical to physical translation
paging

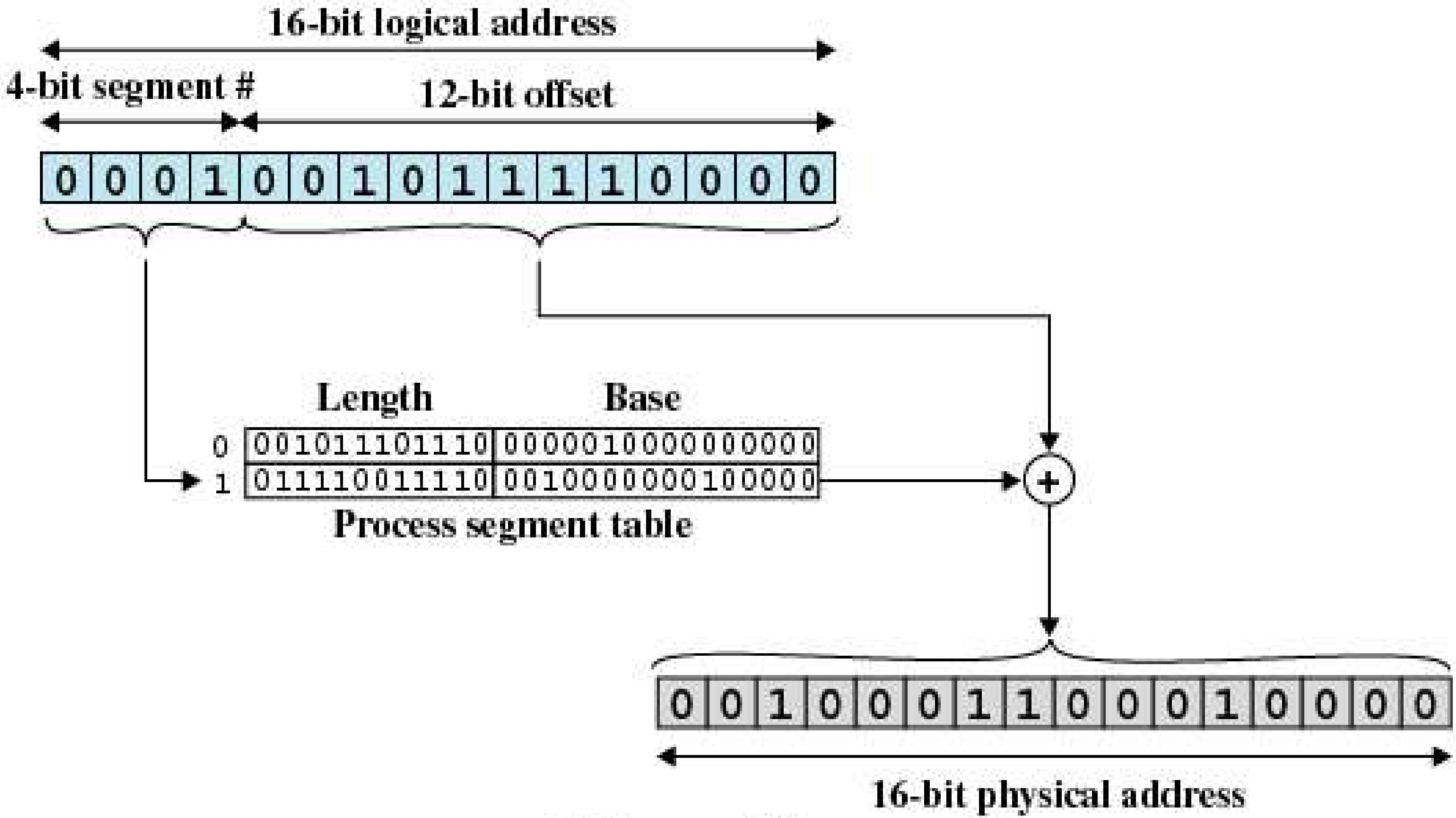


(a) Paging

© 2005 maurizio pizzonia - sistemi operativi a.a. 2004-2005

logical to physical translation

segmentation



(b) Segmentation