

# Operating Systems

# Memory management

Memory allocation and page replacement

Stefano Di Carlo



**Politecnico  
di Torino**

Department of Control and  
Computer Engineering



## 1 Exercise 1 - Contiguous allocation

Consider a system with physical memory of 512MB, in which a management scheme with (contiguous) variable partitions is used with a minimum memory allocation unit of 64B (i.e. the memory space is allocated in multiples of 64 bytes). The first 128MB of memory are permanently allocated to the Operating System.

The process table contains, for each active process, the starting address (ADDR) and the size (SIZE) of the relative partition in memory. The memory is allocated with the Worst-Fit strategy. Free partitions are managed through a linked list sorted by decreasing size, in which each node represents a free partition; the nodes of the list consist of two fields: (pointer to the next partition, and size of the partition, both represented on 4 bytes, size, and addresses represented in Byte, with the value 0 used as a null pointer) and are stored in the first bytes of the partition which represent.

Suppose that at a given instant, the process table and the pointer to the first free partition contain the information shown in Figure 1. Represent the changes to the partitions in memory, the process table, and the Free List, following the activation of 2 new processes, P12 and P13, which require respectively 25MB and 150MB of memory, followed by the termination of the P11 process.

## 2 Exercise 2 - Paging

Consider a virtual memory system with paging, in which the Bytes are addressed. The system has a TLB (Translation Look-aside Buffer), on which a "hit ratio" of 99% is experimentally measured. The page table ("page-table") is created with a two-level scheme, in which a 32-bit logical address is divided (from MSB to LSB) into 3 parts: p1, p2, d, respectively of 10-bit, 11-bit, and 11-bit. No additional data structures (such as hash tables or inverted page tables) are used to speed up access.

- Tell us what is meant by "hit ratio"
- Illustrate the layout of the page table and its overall size for a P1 process having a virtual

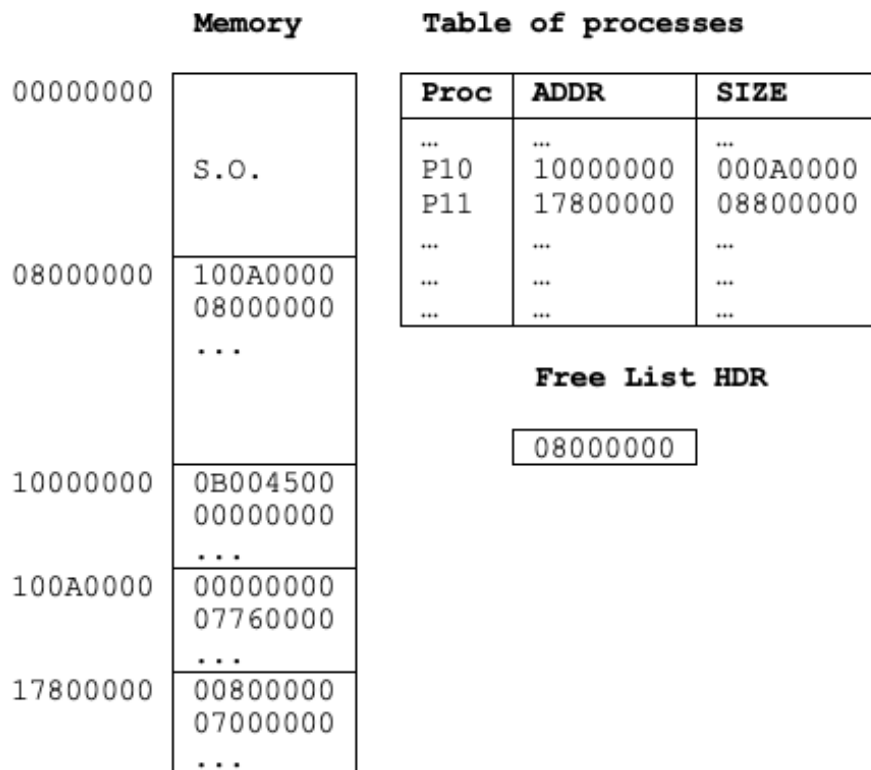


Figure 1: Initial condition

address space of 100MB

- Calculate external and internal fragmentation for process P1 (see the previous point).
- Assuming that the RAM memory has an access time of 300 ns, calculate the effective access time (EAT) for the proposed case (hit ratio = 99%)

### 3 Exercise 3 - Paging

Describe the advantages and disadvantages of an inverted page table (IPT) compared to a standard (possibly hierarchical) page table.

Consider a process having a virtual addressing space of 32 GB, equipped with 8GB of RAM,

on a 64-bit architecture (in which the Byte is addressed), with the management of paged memory (1KB pages/frames). You want to compare a standard page table-based solution (one table for each process) and an IPT-based solution.

Calculate the page table size (at one level only) for the process and the IPT. Let's assume that the pid of a process can be represented on 16 bits. Use 32 bits for page and/or frame indexes. Finally, using the proposed IPT (32 bits for a page/frame index), what is the maximum possible size for the virtual address space of a process?

## 4 Exercise 4 - Page replacement

Consider the following sequence of references in memory in the case of a 1000-word program: 261, 409, 985, 311, 584, 746, 632, 323, 470, 915, 858.

Determine the string of page references, assuming their size is 200 words. Use a second-chance set page replacement algorithm (with a limit of 3 available frames). Suppose the references began when the program started, with the 3 frames allocated to pages n. 4, 3, and 1, and the FIFO queue containing, in order, 40,31,10 (The subscript represents the reference bit). Determine which and how many page faults (accesses to pages not present in the resident set) will occur (the display of the resident set is required after each reference). Suppose the reference bit of a page is initialized to 0 at a page fault.

## 5 Exercise 5 - Page replacement

Consider the following sequence of references in memory in the case of a program in which, for each access (addresses in hexadecimal, the Byte is addressed), it is indicated whether it is reading (R) or writing (W): R 3F5, R 364, W 4D3, W 47E, R 4C8, W 2D1, R 465, W 2A0, R 3BA, W 4E6, R 480, R 294, R 0B8, R 14E.

Assuming that both physical and logical addresses are on 12 bits, that paging with pages of

128 Byte is used, and that the maximum address the program can use is C10, compute how many pages are present in the address space of the program and calculate the internal fragmentation.

Determine the string of the page references (Switching from hexadecimal to binary is recommended to determine the page number correctly and, if necessary, the displacement/offset). An LRU (Least Recently Used) page replacement algorithm is used. Assume that 3 frames are available at physical addresses (expressed in hexadecimal) 780, A00, B00. The computation (after each access) of the resident set (the physical frames containing logical pages) is required.

Determine which and how many page faults (accesses to pages not present in the resident set) will occur. Finally, tell which physical addresses the accesses are made to (among those listed above) R 3F5, W 4D3, R 3BA

## 6 Exercise 6 - Page replacement

Consider the following sequence of references in memory in the case of a 4K word program in which, for each access (addresses in hexadecimal), it is indicated whether it is a read (R) or a write (W): W 3A1, R 3F5, R A64, W BD3, W 57E, R A08, R B85, W 3A0, R A1A, W A36, R B20, R 734, R AB8, R C4E, W B64.

Determine the string of page references, assuming their size is 512 words. An Enhanced Second-Chance page replacement algorithm is used. The modification bit (to be initialized to 0 at the first access to a new page after the relative page fault) is added to the reference bit (modify bit). Assume that a page is always modified in correspondence with a write, that 3 frames are available and that the algorithm operates with the following criterion: given the pointer to the current page (according to the FIFO strategy), a first turn is made, without changing the reference bit on the pages to locate the victim (the order of priority is (reference, modify): (0,0), (0,1), (1,0), (1,1)); once the victim has been determined, a second turn is made to reset the reference bits of the "saved" pages (between the starting position and the victim).

Determine which and how many page faults (accesses to pages not present in the resident set)

will occur. The resident set is requested to be displayed (after each access), indicating the reference and modification bits for each frame. Number the pages starting from 0.

For this question, use the following scheme to carry out the exercise, indicating in the first line the string of references to pages (represented by choice in hexadecimal or decimal), in the second Read or Write, in the following three (representing the 3 frames of the resident set), the pages allocated in the corresponding frames, indicating the bits for each (reference, modify). Also indicate (by underlining, circling or placing an arrow) which page is at the top of the FIFO. In the last line, indicate the presence or absence of a Page Fault.

## 7 Exercise 7 - Page replacement

Consider the following sequence of references to pages in memory: 577517111434123. Use a working-set page replacement algorithm (exact version) with window  $\Delta = 3$ , assuming that a maximum of 3 frames are available. Determine which and how many page faults (accesses to pages not present in the resident set) and page outs (removals of pages from the resident set) will occur. The resident set must be displayed (after each access).

We also want to define a measure of locality of the program carried out, based on the Reuse Distance. The Reuse Distance at time  $T_i$  ( $RD_i$ ), in which the page  $P_i$  is accessed, is defined as the number of pages (distinct and different from  $P_i$ ) accessed starting from the previous access to  $P_i$  (assuming, conventionally, for the first access to a page, the total number of pages accessed up to that moment). For example, at time 3, in which the second access to page 5 is made,  $RD_3 = 1$ , since between the two accesses to page 5 only one page was accessed twice (7). Given the various  $RD_i$ , calculate the average  $RD_{avg}$  value. The locality of the program carried out is defined as  $L = 1 / (1 + RD_{avg})$ . Calculate the  $RD_i$ ,  $RD_{avg}$  and  $L$ .

For this question, use the following scheme to carry out the exercise. The references and the Reuse Distance at times 0 and 3 have already been indicated.

## 8 Exercise 8 - Page replacement

Given the string of references to pages 3, 4, 1, (3, 1, 4, 4, 3, 1, 1) \*10, where the syntax (...) \*n indicates that the string in brackets is repeated / iterated n times (the string can for example derive from an iterative construct).

Use a working-set page replacement algorithm (exact version) with window of duration  $\Delta = 3$ . Suppose you call page-out the removal of a page from the resident set (as it leaves the working set). The references and the resident set are displayed in the following diagram after each reference, indicating the page-faults (accesses to pages not present in the resident set) and the page-outs. Analyze the first two iterations of the substring repeated 10 times. ATTENTION: each row of the resident set represents a frame, therefore a page present in a frame cannot change row when it remains in the resident set. In the case of page-fault without page-out, use the first free frame from above. In the case of page-out and (simultaneous) page-fault, the frame just freed (from page-out) is reused. If page-out and page-fault are related to the same page, the replacement algorithm takes them into account, avoiding both page-out and page-fault.

©2023 Stefano Di Carlo - stefano.dicarlo [at] polito [dot] com.

This work is licensed under a Creative Commons Attribution-NonCommercial 3.0 (CC BY-NC

3.0). To view a copy of this license, visit:

<https://creativecommons.org/licenses/by-nc/3.0/legalcode>.

