
# OPERATING SYSTEMS

**Ebtehal A. Alsaggaf**

**Lab 9&10: Memory Management**

**Cpcs 361 Spring 2013**

# Objectives

- understand some Basic Memory Management Concepts likePaging and Segmentation
- First Fit Best Fit and Worst Fit Algorithms and solve the exercise.
- understand page replacement algorithms, concept of Demand Paging, Converting virtual addresses to their equivalent physical addresses in hexadecimal and solving the exercise

# Paging

- Paging – Computer memory is divided into small partitions that are all the same size and referred to as, page frames.
- Then when a process is loaded it gets divided into pages which are the same size as those previous frames. The process pages are then loaded into the frames.

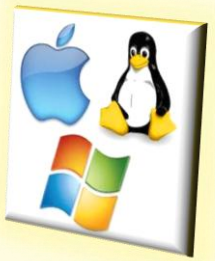
# *Segmentation*

- Segmentation – Computer memory is allocated in various sizes (segments) depending on the need for address space by the process. These segments may be individually protected or shared between processes. Commonly you will see what are called “Segmentation Faults” in programs, this is because the data that’s is about to be read or written is outside the permitted address space of that process..

- Best Fit: The allocator places a process in the smallest block of unallocated memory in which it will fit.

- First Fit: simply scans the free list until a large enough hole is found. Despite the name, first-fit is generally better than best-fit because it leads to less fragmentation.

- Worst Fit: The memory manager places process in the largest block of unallocated memory available. The idea is that this placement will create the largest hole after the allocations, thus increasing the possibility that, compared to best fit, another process can use the hole created as a result of external fragmentation.



# Question # 1

Assume that the main memory has the following 5 fixed partitions with the following sizes: 100KB, 500KB, 200KB, 300KB and 600KB (in order)

- a) Explain the following allocation algorithms: First-fit, Best-fit and Worst-fit.
- b) How would each of the First-fit, Best-fit and Worst-fit algorithms place processes of 212KB, 417KB, 112KB and 426KB (in order)?
- c) Compute the total memory size that is not used for each algorithm.
- d) Which algorithm makes the efficient use of the memory?

# Answer # 1:

| Fisrt-fit | | Best-fit | | Worst-fit | |
|---|---|---|---|---|---|
| 100K | | 100K | | 100K | |
| 500K | 212K | 500K | 417K | 500K | 417K |
| | 288K | | 83K | | 83K |
| 200k | 112k | 200K | 112K | 200K | |
| | 88K | | 88K | | |
| 300K | | 300K | 212K | 300K | 112K |
| | | | 88K | | 188K |
| 600K | 417K | 600K | 426K | 600K | 212K |
| | 183K | | 174K | | 388K |

# Answer # 1:

a. First-fit:

1. 212K is put in 500K partition
2. 417K is put in 600K partition
3. 112K is put in 288K partition (new partition 288K = 500K – 212K)
4. 426K must wait

b. Best-fit:

1. 212K is put in 300K partition
2. 417K is put in 500K partition
3. 112K is put in 200K partition
4. 426K is put in 600K partition

c. Worst-fit:

l. 212K is put in 600K partition
2. 417K is put in 500K partition
3. 112K is put in 388K partition
4. 426K must wait

In this example, best-fit turns out to be the best.

# Converting virtual addresses to their equivalent physical addresses

Consider the page table for a system with 12-bit virtual and physical addresses with 256-byte pages. The list of free page frames is D, E, F (that is, D is at the head of the list, E is second, and F is last).

| Page | Page Frame |
|---|---|
| 0 | - |
| 1 | 2 |
| 2 | C |
| 3 | A |
| 4 | - |
| 5 | 4 |
| 6 | 3 |
| 7 | - |
| 8 | B |
| 9 | 0 |

Convert the following virtual addresses to their equivalent physical addresses in hexadecimal. All numbers are given in hexadecimal. (A dash for a page frame indicates that the page is not in memory.)

Size of virtual/physical address space = $2^{12}$ = 4096 bytes. Page size = 256 bytes = $2^{8}$. Therefore, 12 – 8 = 4 high-order bits of logical address represent page number and 8 low-order bits represent page offset.

- 9EF → corresponds to page 9 → **Physical address = 0EF**
- 111 → corresponds to page 1 → **Physical address = 211**
- 700 → corresponds to page 7 → Read page 7 to free page frame D → **Physical address = D00**
- 0FF → corresponds to page 0 → Read page 0 to free page frame E → **Physical address = EFF**

# Question # 2

Consider the following segment mapping table (SMT)

| *Segment* | *Base* | *Length (limit)* |
|---|---|---|
| **0** | 219 | 600 |
| **1** | 2300 | 14 |
| **2** | 90 | 100 |
| **3** | 1327 | 580 |
| **4** | 1952 | 96 |

What are the physical addresses of the following logical addresses?

a) <0, 430>

b) <1, 10>

c) <2, 500>

d) <3, 500>

e) <4, 122>

f) <6, 1952>

***Answer :***

a) <0, 430> 219 + 430 = 649
b) <1,10> 2300 + 10 = 2310
c) <2,500> Trap (500 >= 100)
d) <3,500> 1327 + 500 = 1827
e) <4,122> Trap (122 >= 96)
f) <6,1952> Trap (6 is an invalid entry)


THANK
YOU!