Fundamentos de los Sistemas Operativos (FSO)

Departamento de Informática de Sistemas y Computadoras (DISCA)
Universitat Politècnica de València

Part 3: Memory management

Unit 7
Memory management

fSO

DISCA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Goals and bibliography

- **Goals**
  - To introduce the basic concepts related to **memory management**
  - To understand the difference between **logical and physical memory**
  - To understand the **contiguous memory allocation** concept
  - To analyse the **fragmentation** problem associated to contiguous memory allocation
  - To study **contiguous allocation strategies**

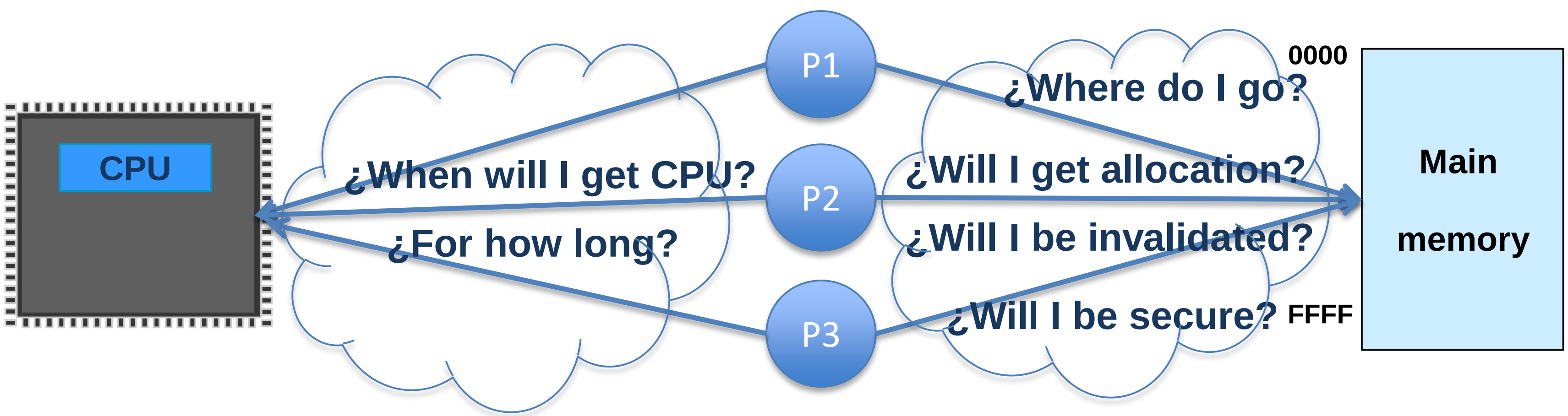
- Bibliography:
  - Silbershatz, chapter 8
  - Carretero, chapter 5

Content

- **Introduction**
- Memory management issues
- Logical vs. physical addresses
- Memory management unit
- Contiguous memory allocation

Introduction

- To execute a program
  - Both **instructions** and **data** must be allocated in **main memory**
- To get more **system efficiency → multiprogramming**
  - Processes in a **multiprogrammed system**
    - Share CPUs → **Process scheduling**
    - Share main memory → **Memory management**



Introduction

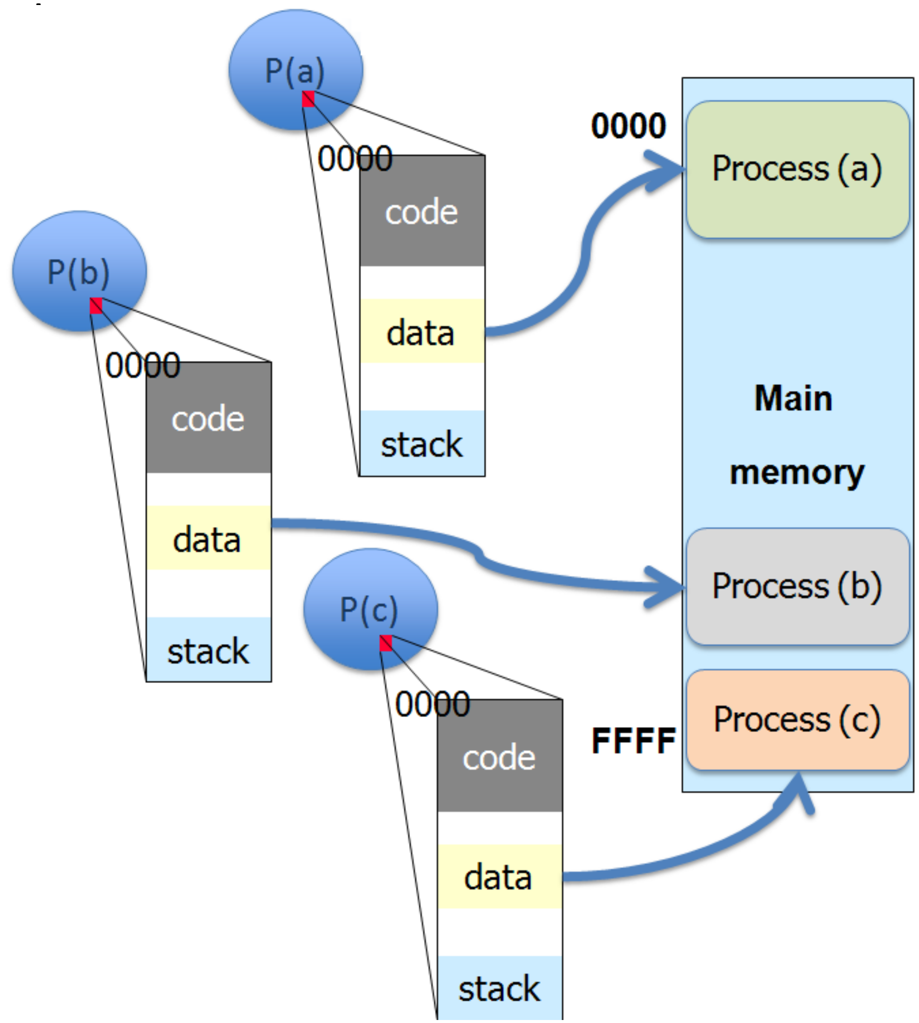
- Computer dynamic storage is available at several levels:
  - CPU registers
  - Cache memory
    - Small buffer that balances memory access and CPU speeds
  - Main memory
- Main memory
  - It is made by a big **binary word or byte vector**, every one with its one **physical address**
  - It is a **critical resource**
    - Its availability is fundamental to system operation because it is accessed continuously -> **instruction execution cycle**
    - It has a **limited allocation capability**

Content

- Introduction
- **Memory management issues**
- Logical vs. physical addresses
- Memory management unit
- Contiguous memory allocation

Memory management issues

- A key OS issue is to offer a good ar efficient memory management, fa the following problems:
  - **Allocation**
  - **Protection**
  - **Shortage**
  - **Relocation**
- Modern OSs own techniques and mechanisms that have evolved an improved to solve former problem
  - Logical address space
  - MMU
  - Dynamic libraries
  - Virtual memory
  - Allocation techniques

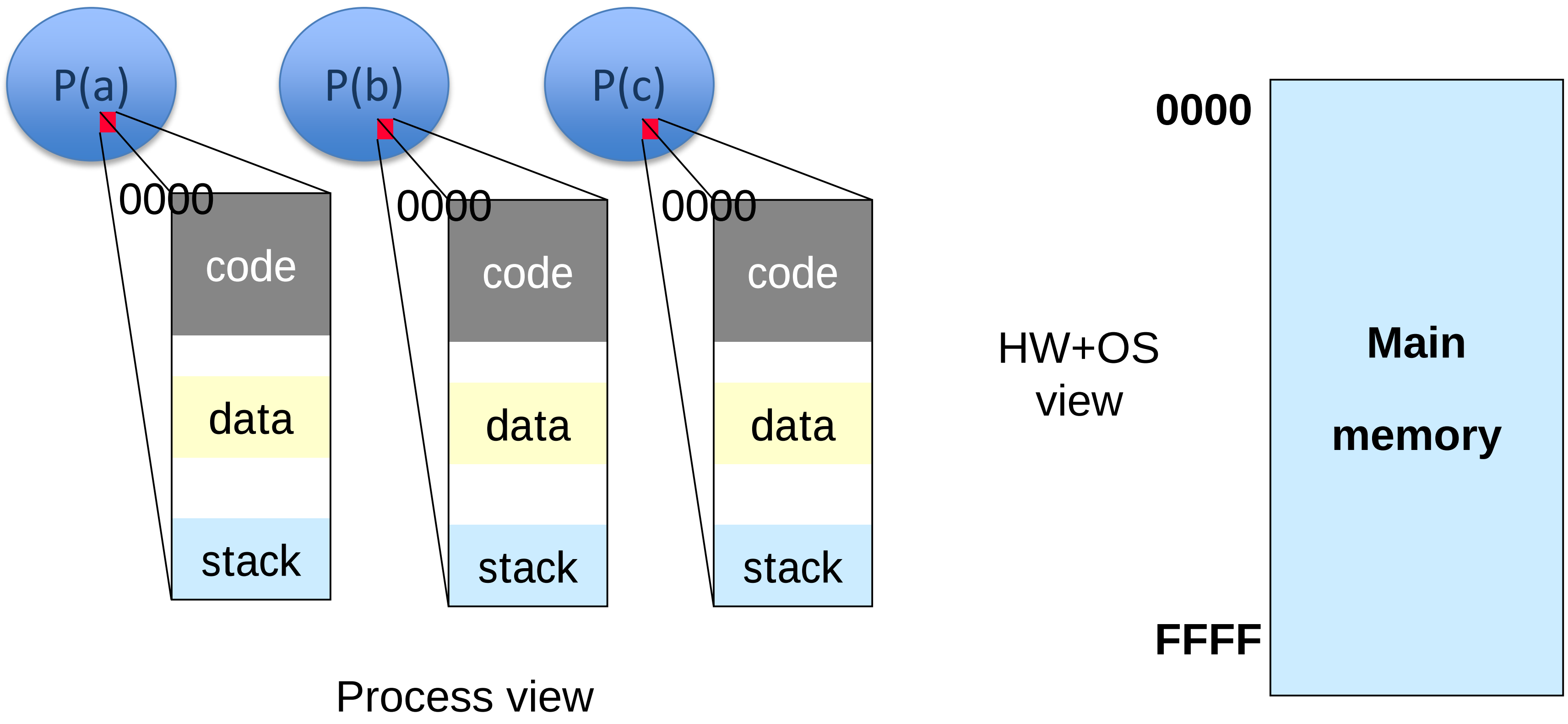


Content

- Introduction
- Memory management issues
- **Logical vs. physical addresses**
- Memory management unit
- Contiguous memory allocation

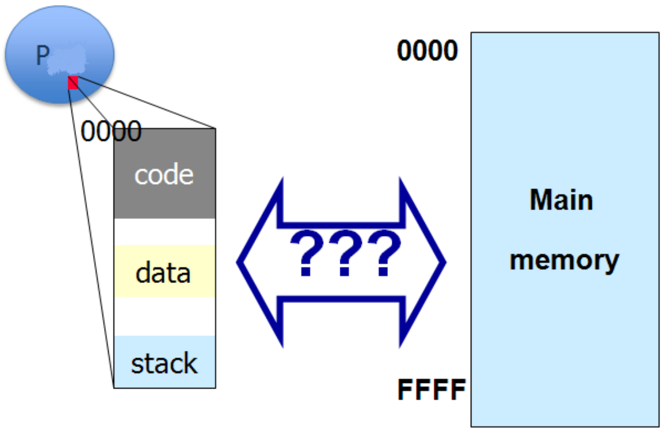
Logical vs. physical addresses

- **Logical addresses**
  - Every process has its own independent address space
  - It allows the code not being involved with machine features -> relocation



Logical vs. physical addresses

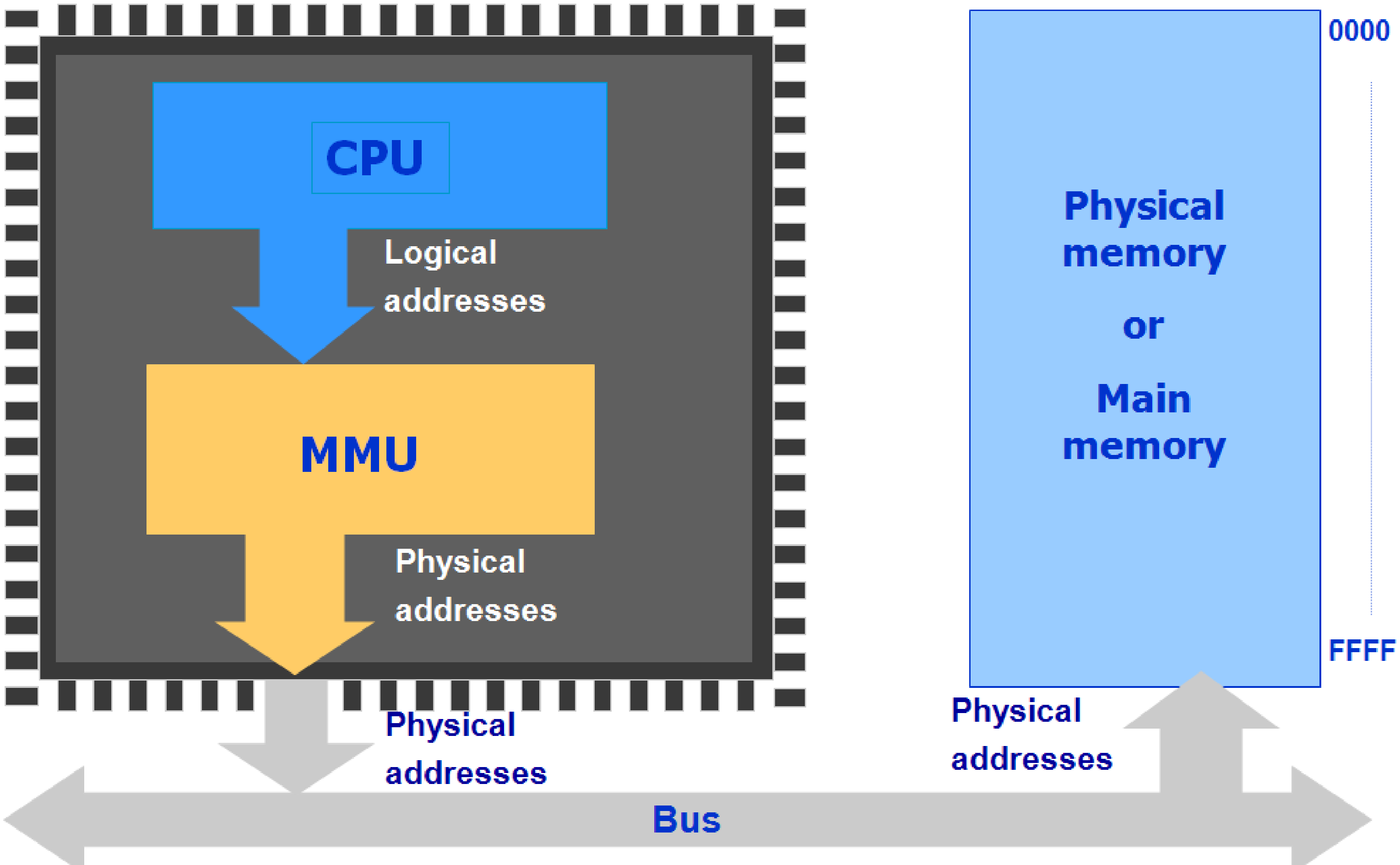
- Need of logical to physical translation
  - **What physical address corresponds to a given logical address?**
  - Using a logical addressing (LA) and a physical addressing (PA) it is required:
    - A translation function from LA to PA
    - To decide how to implement it,
      - Hardware? Software?
    - **When** the translation is done
      - **At compilation time**
        - » Absolute code -> non relocatable
      - **At program load time**
        - » Relocatable code at load time
      - **At execution time**
        - » The process can relocated while it is executed



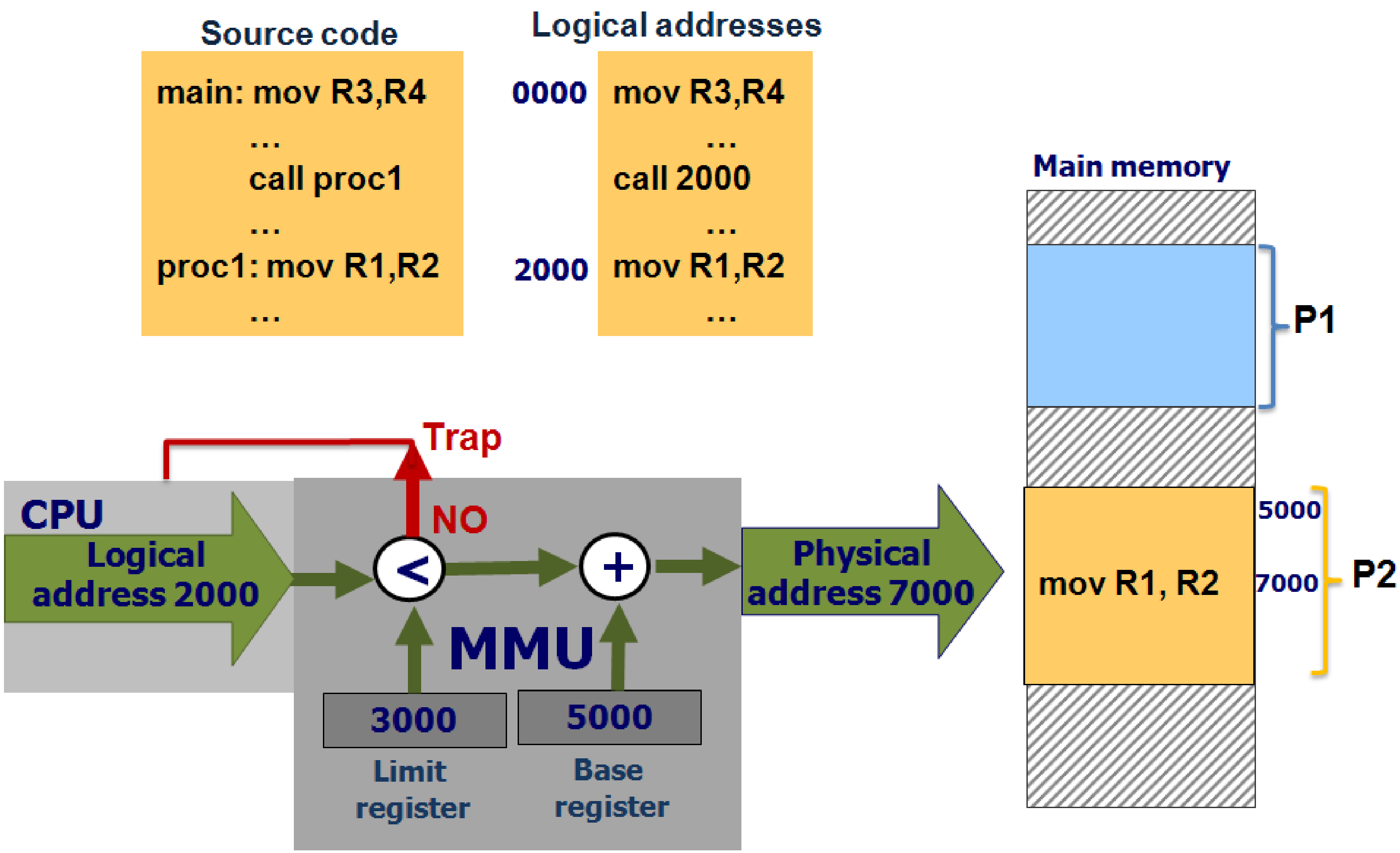
Content

- Introduction
- Memory management issues
- Logical vs. physical addresses
- **Memory management unit**
- Contiguous memory allocation

- **Translating logical addresses into physical addresses** is an overhead
  - **MMU → hardware translation MINIMUM OVERHEAD**



- Simple MMU model
  - Based on base and limit registers



!!!!!! It guaranties relocation at execution time and protection

Content

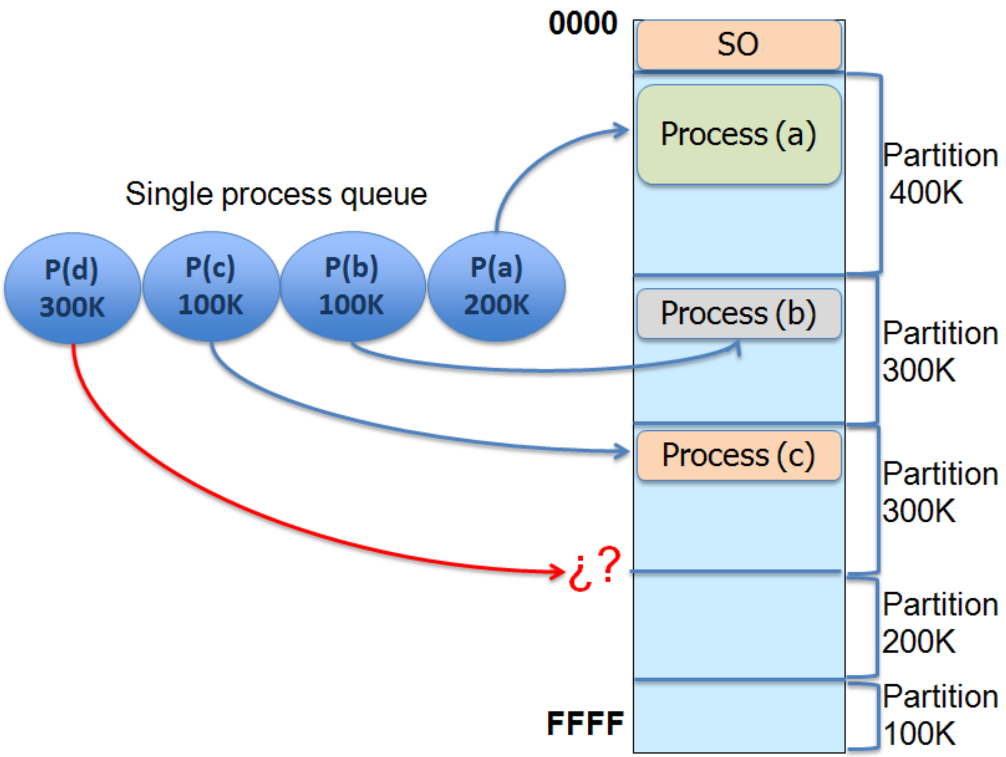
- Introduction
- Memory management issues
- Logical vs. physical addresses
- Memory management unit
- **Contiguous memory allocation**

Contiguous memory allocation

- Contiguous allocation memory management model
  - A **process** is allocated in a unique section in memory with a **contiguous range of physical addresses**
- Memory is usually divided in two parts:
  - The OS allocation area
  - The user process allocation area
- Contiguous allocation strategies:
  - A priory definition (system configuration) of memory section to allocate processes: **fixed partitions**
  - To allow the system allocating processes into available holes: **variable partitions**

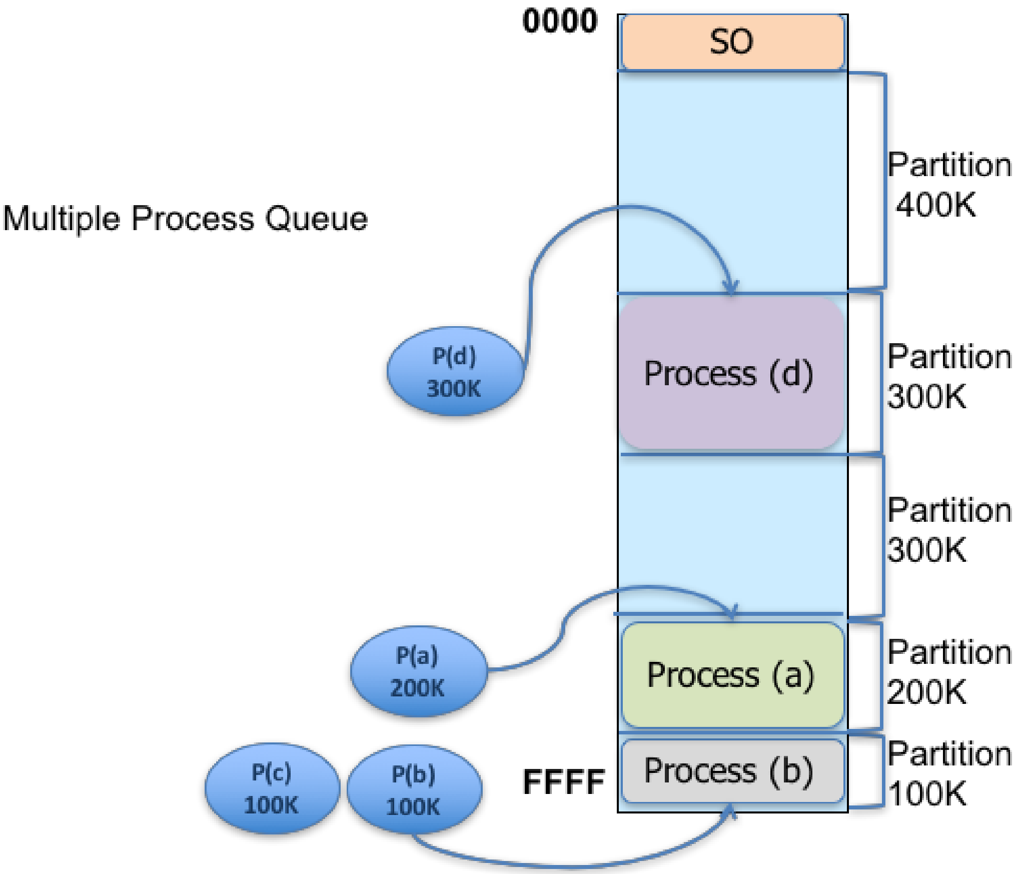
Contiguous memory allocation

- **Fixed partitions**
  - Memory is supposed to be divided into **different fixed size partitions**
  - The OS maintains a **free partitions list**
  - Variations:
    - Single process queue
    - Multiple process queue
  - Problem:
    - **Internal fragmentation**



Contiguous memory allocation

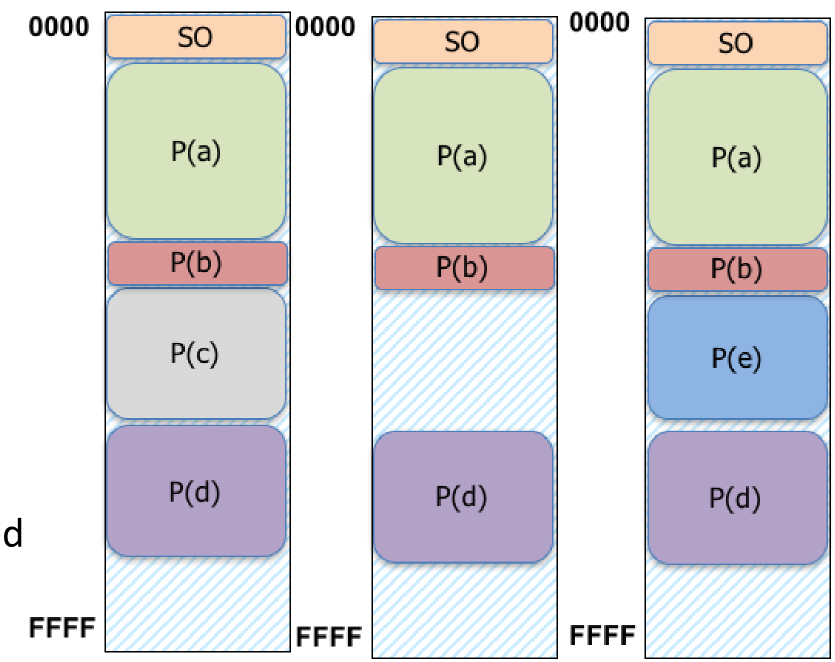
- **Fixed partitions**
  - Memory is supposed to be divided into **different fixed size partitions**
  - The OS maintains a **free partitions list**
  - Variations:
    - Single process queue
    - Multiple process queue
  - Problem:
    - **Internal fragmentation**



Contiguous memory allocation

- **Variable partitions**
  - Initially process available memory is all available into a single hole
  - As process demands arrive memory is allocated
  - The OS maintains a free hole list with hole address and size

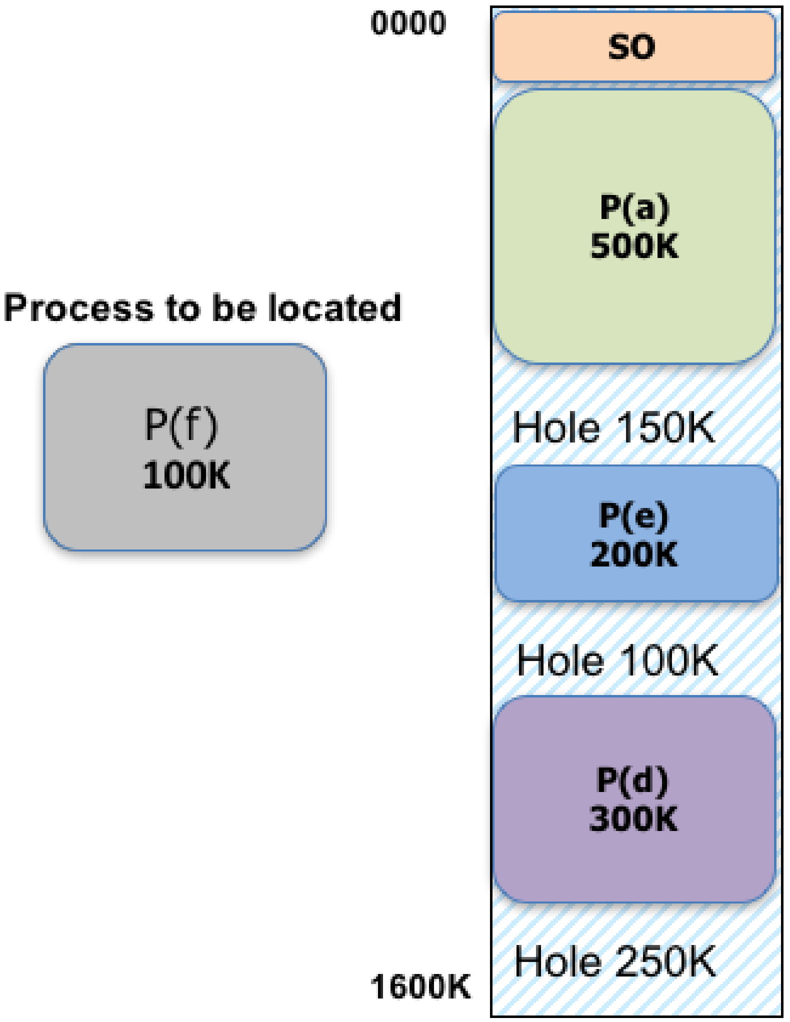
  - Problem:
    - **External fragmentation**



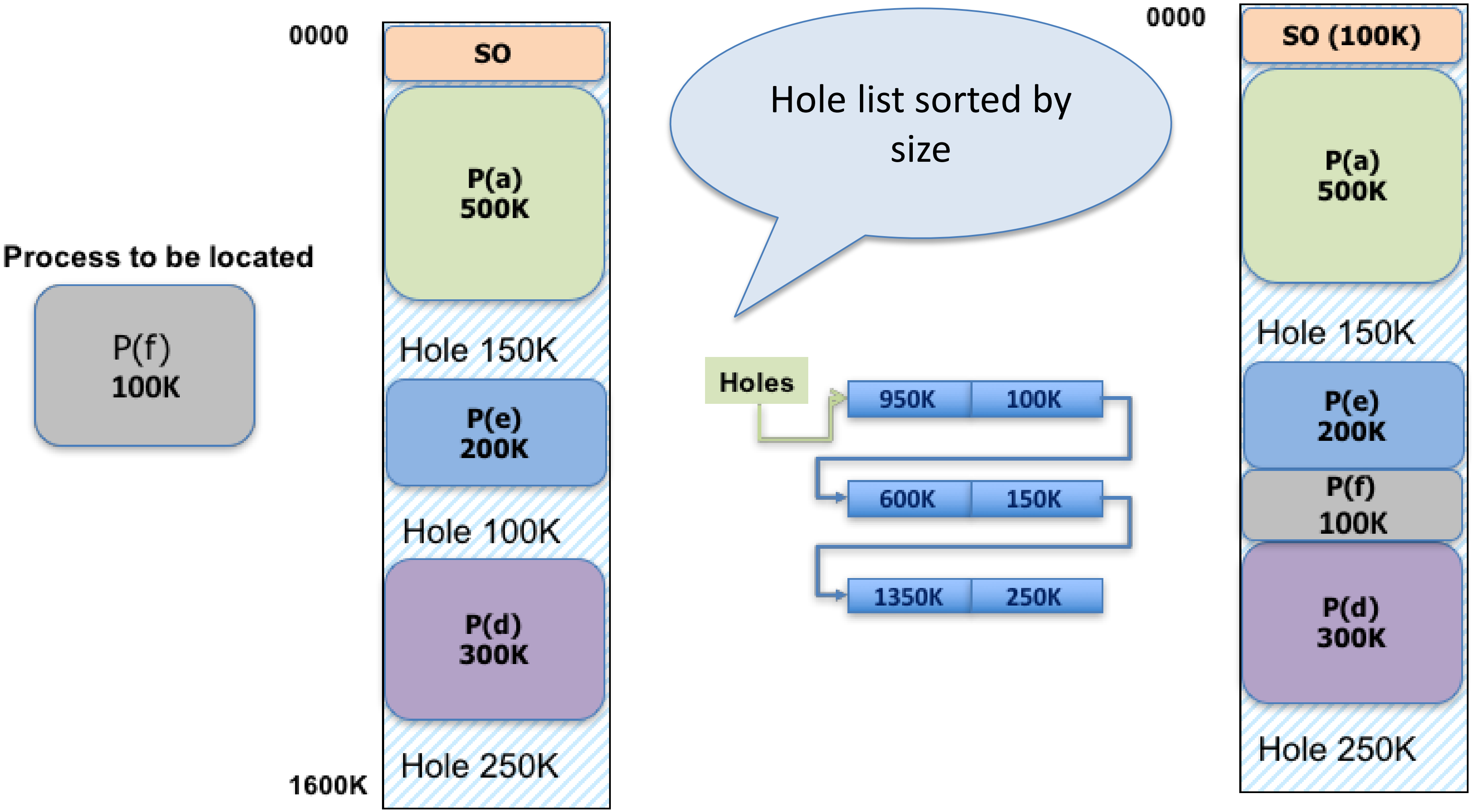
- Hole allocation strategies:
  - **Best Fit**
    - The smallest satisfying hole is allocated
  - **Worst Fit**
    - The biggest satisfying hole is allocated
  - **First Fit**
    - The first found satisfying hole is allocated

- **Variable Partitions**

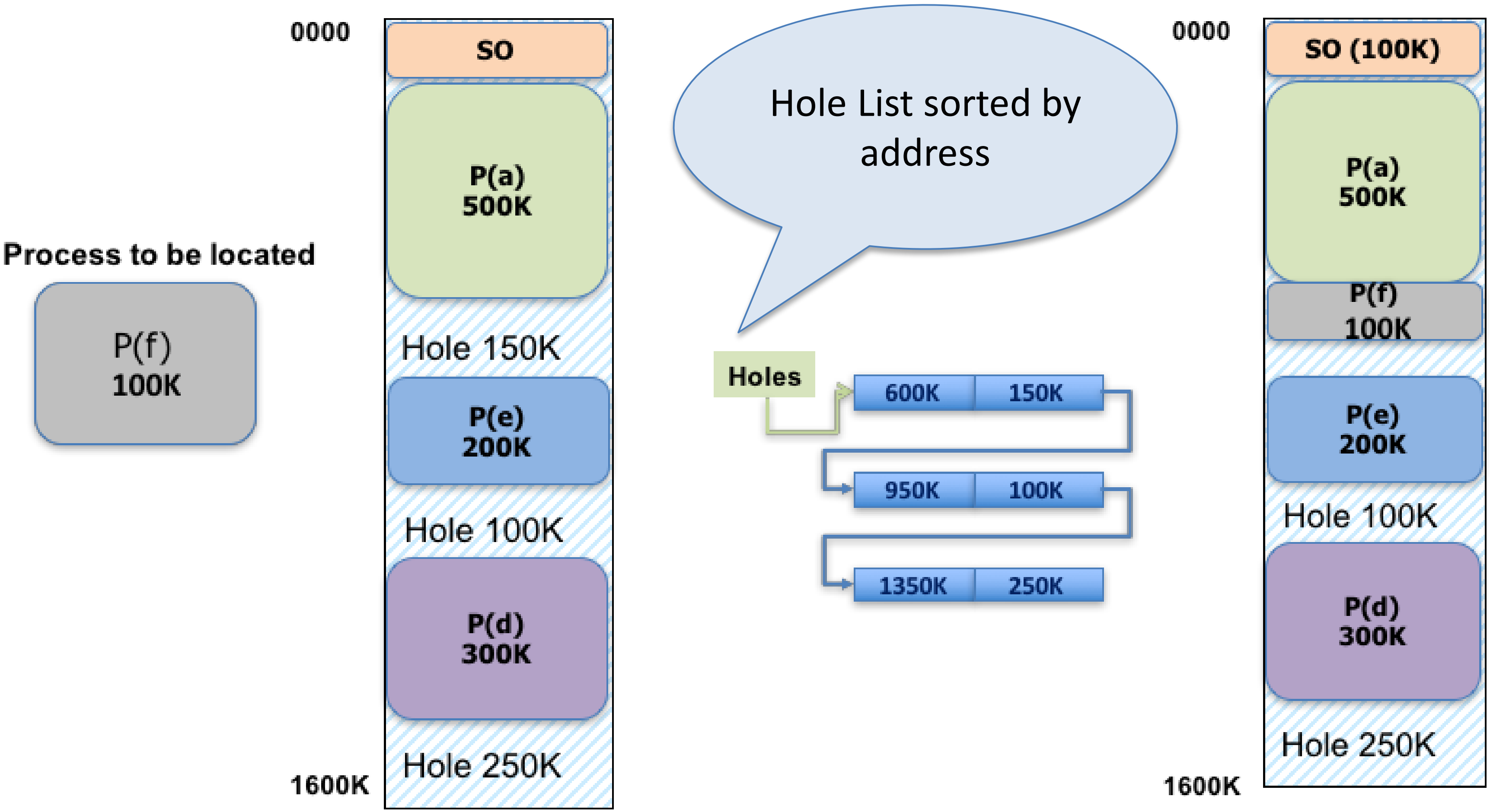
Where the P(f) process would be located in each of the allocation strategies?



- Best Fit

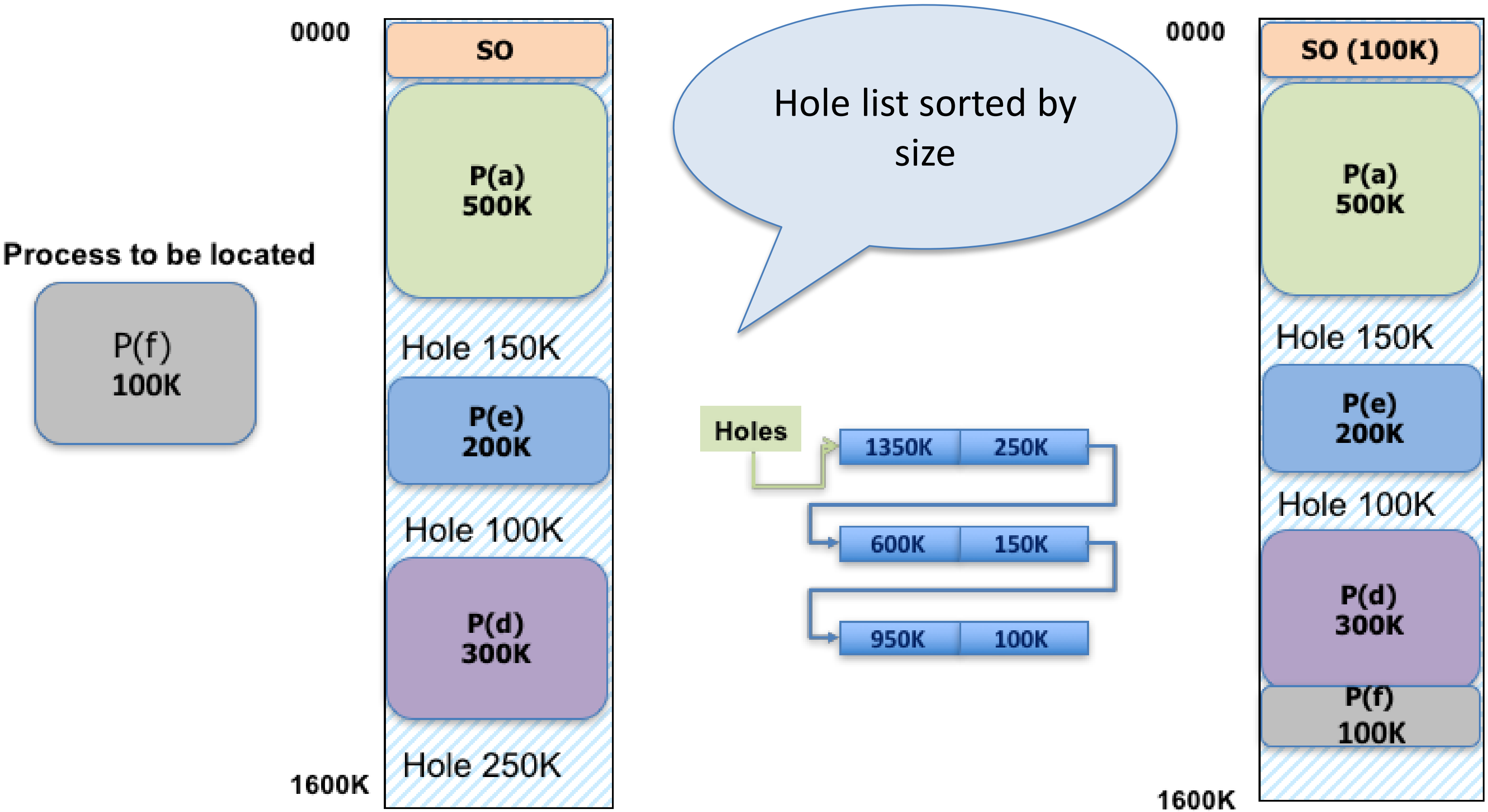

Process to be located
P(f)
100K
0000
SO
P(a)
500K
Hole 150K
P(e)
200K
Hole 100K
P(d)
300K
Hole 250K
1600K
Hole list sorted by size
Holes
950K
100K
600K
150K
1350K
250K
0000
SO (100K)
P(a)
500K
Hole 150K
P(e)
200K
P(f)
100K
P(d)
300K
Hole 250K

- First Fit



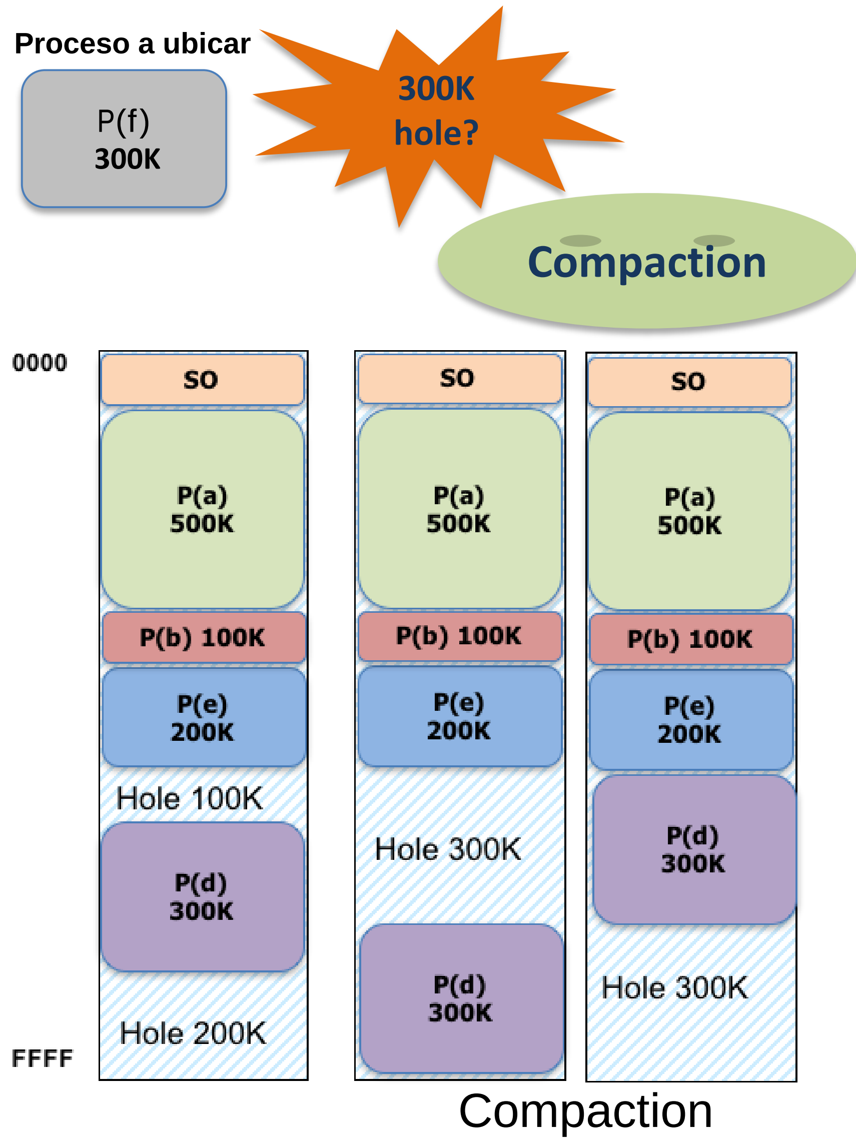
Process to be located
P(f)
100K
0000
SO
P(a)
500K
Hole 150K
P(e)
200K
Hole 100K
P(d)
300K
Hole 250K
1600K
Hole List sorted by address
Holes
600K 150K
950K 100K
1350K 250K
0000
SO (100K)
P(a)
500K
P(f)
100K
P(e)
200K
Hole 100K
P(d)
300K
Hole 250K
1600K

- Worse Fit


Process to be located
P(f)
100K
0000
SO
P(a)
500K
Hole 150K
P(e)
200K
Hole 100K
P(d)
300K
Hole 250K
1600K
Hole list sorted by size
Holes
1350K
250K
600K
150K
950K
100K
0000
SO (100K)
P(a)
500K
Hole 150K
P(e)
200K
Hole 100K
P(d)
300K
P(f)
100K
1600K

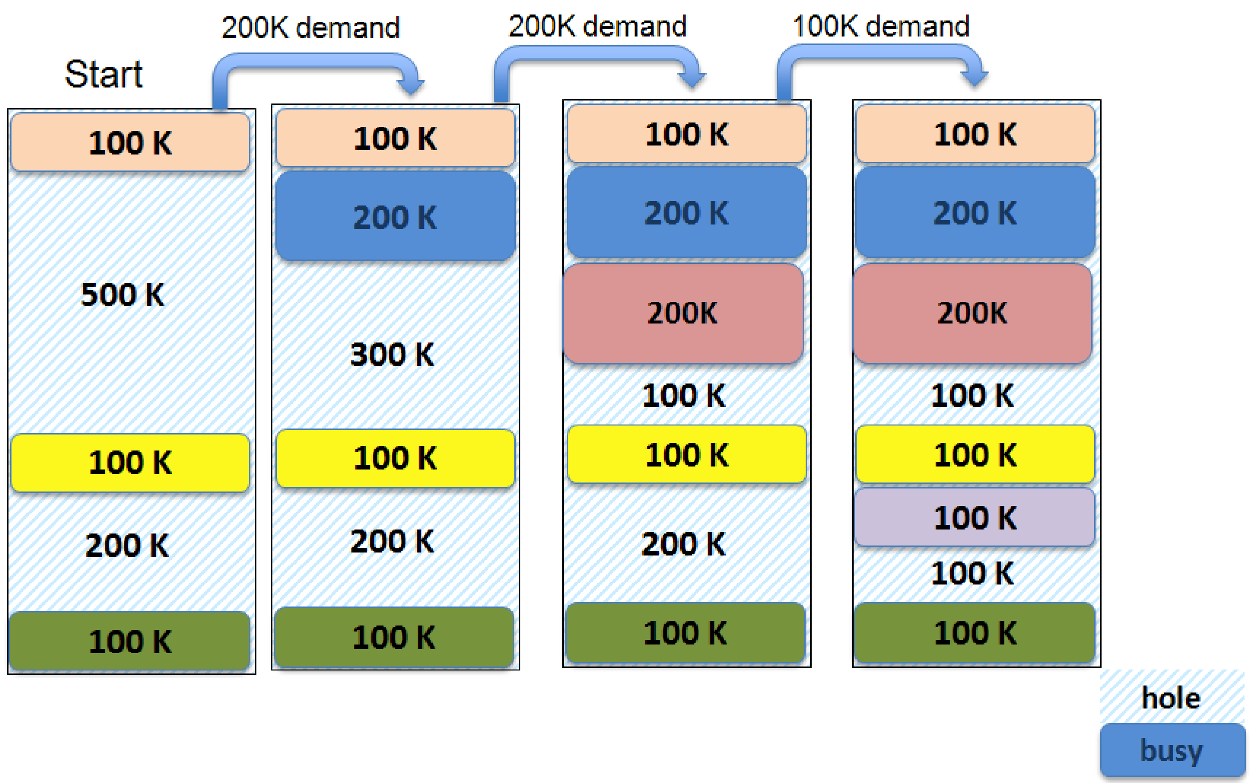
Contiguous memory allocation

- **External fragmentation**
  - The added size of available holes may be enough but it cannot be allocated because it is not contiguous
- **Compaction**
  - External fragmentation solution
  - Processes should be relocated in main memory
  - Relocatable code at execution time required



Example 1

- Consider an OS that manages memory by means of **contiguous allocation with variable partitions**. From memory state "Start" memory occupation has evolved as shown in the following figure:



- What is the allocation algorithm selected between best fit, worst fit or first fit?

Example 2

- An OS manages its 8196 main memory words by means of contiguous allocation with variable partitions. At instant **t** there are three processes in execution with the following sizes:

  Process A 1024 words

  Process B 3072 words

  Process C 3584 words

Are the following independent situations possible?

a) Process **B generates logical address 960** that is translated into **physical address 725**

b) Process **A generates logical address 1500** that is translated into **physical address 1500**

c) Process C at time **t** generates logical address 525, that is translated into physical address 2061 and at time **t**+10 the same logical address is translated into physical address 1549

Contiguous memory allocation

- Interactive learning objects:
  - http://labvirtual.webs.upv.es/Fijas_Multiples_colas.htm
  - http://labvirtual.webs.upv.es/Fijas_Una_cola.htm
  - http://labvirtual.webs.upv.es/Best_Fit.htm
  - http://labvirtual.webs.upv.es/Worst_Fit.htm