Sistemas Operativos

Administración de Memoria I

Curso 2023

Facultad de Ingeniería, URL

Universidad Rafael Landívar
Tradición Jesuita en Guatemala

Agenda

1. Introducción
2. Preparación de un programa para ejecutar
3. Áreas de la memoria de un proceso.
4. Ensamblaje dinámico y bibliotecas compartidas
5. Asociación de direcciones
6. Asignación dinámica de la memoria a nivel de proceso
7. Tipos de direccionamiento
8. Protección de memoria
9. Asignación de memoria
10. Swapping

Introducción

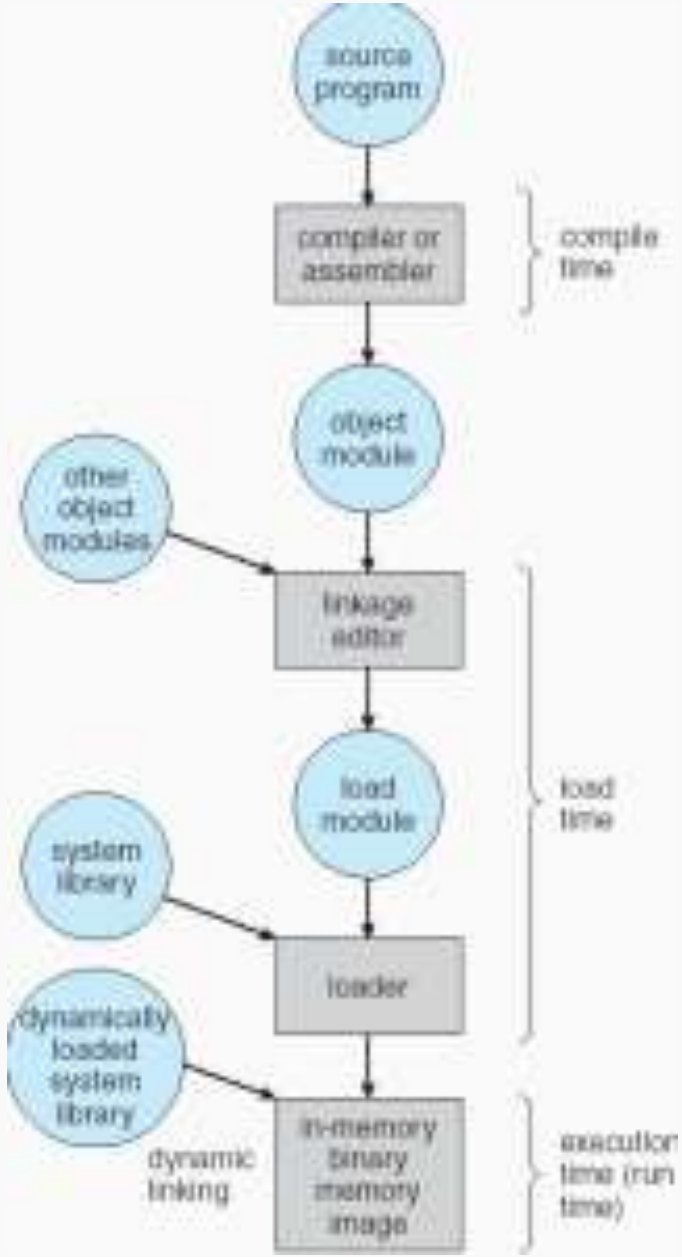
Introducción

- La administración de la memoria es una de las tareas más importantes del sistema operativo.
- En los sistemas operativos multiprogramados es necesario mantener varios programas en memoria al mismo tiempo.
- Existen varios esquemas para la administración de la memoria y requieren distinto soporte del hardware.
- El sistema operativo es responsable de las siguientes tareas:
    - Mantener qué partes de la memoria están siendo utilizadas y por quién.
    - Decidir cuáles procesos serán cargados a memoria cuando exista espacio de memoria disponible.
    - Asignar y quitar espacio de memoria según sea necesario

Conceptos Básicos

- Preparación de un programa para ejecutar.
- Los programas son escritos, por lo general, en lenguajes de alto nivel y deben pasar por distintas etapas antes de ser ejecutados:
  - Compilación (*compile*): Traducción del código fuente del programa a un código objeto.
  - Ensamblaje (*link*): Ensamblaje de varios códigos objetos en un archivo ejecutable.
  - Carga (*load*): Asignación del archivo ejecutable a la memoria principal del sistema.
- Un programa ejecutable consta de secciones de instrucciones y de datos
- El linker surge ante la necesidad de modularizar y reutilizar código. Se resuelven las referencias externas, así como las posiciones relativas de los símbolos en los diferentes módulos, formando uno consolidado.

Conceptos Básicos



Preparación de un programa para ejecutar

Conceptos Básicos

- Cuando un proceso es creado el cargador (loader) del sistema crea en memoria el espacio necesario para la diferentes áreas y la carga con la información.
- El compilador, ensamblador, sistema operativo y bibliotecas dinámicas deben cooperar para administrar la información y realizar la asignación.

Conceptos Básicos

- Compilador: genera un archivo objeto para cada archivo fuente. La información está incompleta, ya que se utilizan información de otros archivos (cómo llamados a funciones externas).
- Ensamblador (*linker*): combina todos los archivos objetos de un programa dentro de un único archivo objeto.
- Sistema operativo: Carga los programa en memoria, permite compartir la memoria entre varios procesos y brinda mecanismos a los procesos para obtener más memoria en forma dinámica.
- Bibliotecas dinámicas: proveen rutinas cargadas en tiempo de ejecución (ejemplo: biblioteca del runtime de C).

Áreas de la memoria de un proceso.

Conceptos Básicos

- La memoria de un proceso cuando ejecuta se estructura en diferentes áreas:

Ensamblaje dinámico y bibliotecas compartidas

Carga dinámica (dynamic loading)

- El tamaño de un proceso en memoria está limitado por la cantidad de memoria física del sistema.
- Con el fin de lograr un mayor aprovechamiento de la memoria se puede utilizar la carga dinámica.
- La carga dinámica implica que una rutina no es cargada en memoria física hasta que no sea invocada.
- La ventaja de la carga dinámica es que las rutinas que no son utilizadas, no son cargadas en memoria física y, por lo tanto, no consumen recursos innecesariamente.

Ensamblaje dinámico (dynamic linking)

- En la etapa de ensamblaje de un programa las bibliotecas compartidas pueden incorporarse al archivo ejecutable generado (ensamblaje estático o static linking).
  - Ej. en Linux: /usr/lib/libc.a
- Otra alternativa es que las bibliotecas compartidas sean cargadas en tiempo de ejecución (ensamblaje dinámico o dynamic linking).
  - Ej. en Linux: /lib/libc.so
  - En Windows: system.dll

Ensamblaje dinámico (dynamic linking)

- En los archivos ejecutables las bibliotecas estáticas son incorporadas, mientras que para las dinámicas se mantiene una referencia.
- Ej. en Linux comando ls:

```
$ ldd /bin/ls
        librt.so.1 =>  /lib/librt.so.1  (0x4001c000)
        libc.so.6 => /lib/libc.so.6 (0x40030000)
        libpthread.so.0 => /lib/libpthread.so.0 (0x40149000)
        /lib/ld-linux.so.2 (0x40000000)
```

- Esto permite, junto con la carga dinámica, hacer un uso más eficiente de la memoria, ya que las bibliotecas dinámicas se cargan una única vez en memoria principal.

Asociación de direcciones

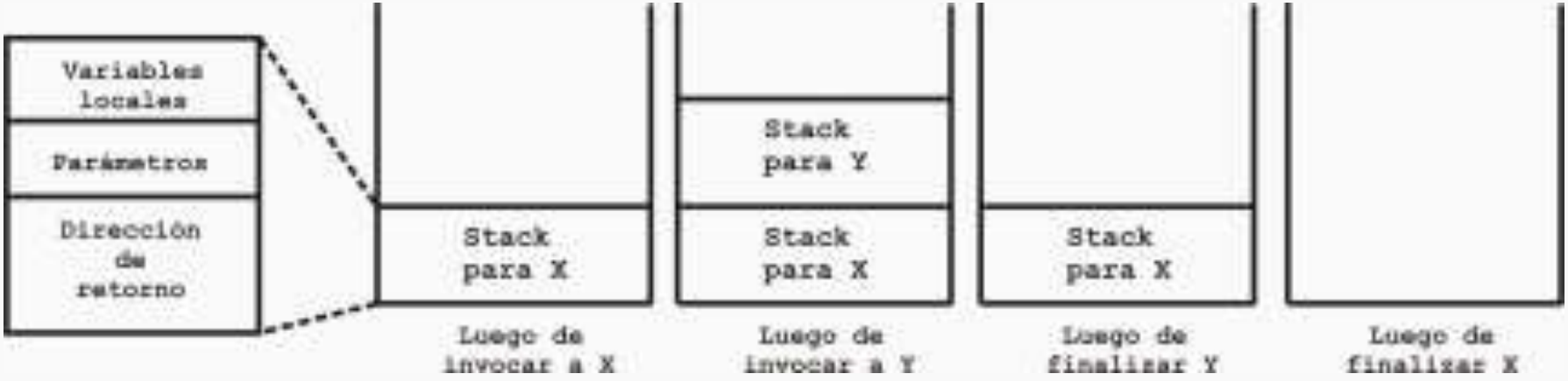
Asociación de direcciones (address binding)

- La asignación de la ubicación de un programa en memoria principal puede ser realizada en varios tiempos:
  - Tiempo de compilación (*compile time*): El programa será asignado a un lugar específico y conocido de la memoria física. La direcciones de memoria son referenciadas en forma absoluta (*static relocation*).
  - Tiempo de carga (*load time*): La asignación del lugar de memoria donde será cargado el programa es hecho al momento de la carga. Las direcciones de memoria deben ser referenciadas en forma relativa (*dynamic reolcation*).
  - Tiempo de ejecución (*execution time*): Un programa puede variar su ubicación en memoria física en el transcurso de la ejecución.

Asignación dinámica de la memoria a nivel de proceso

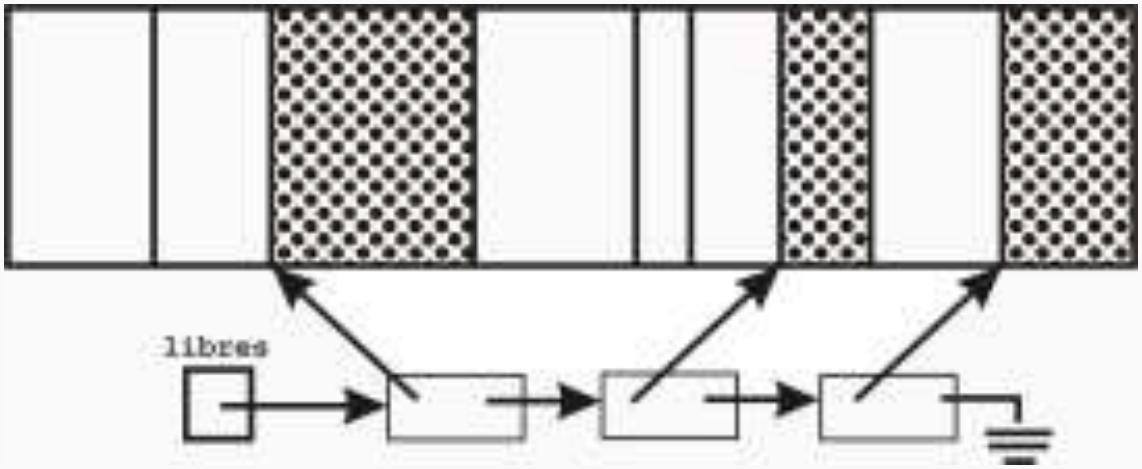
Asignación dinámica a nivel de proceso

- La asignación dinámica en un proceso se da a través de:
  - Asignación en el Stack.
  - Asignación en el Heap.
- A nivel del stack la memoria se comporta en forma más predictiva.



Asignación dinámica a nivel de proceso

- La asignación en el heap no es predictiva como en el caso del stack:



- En este caso se genera fragmentación de la memoria.
- Los sistemas operativos optan por delegar la administración de esta memoria a bibliotecas de usuario.

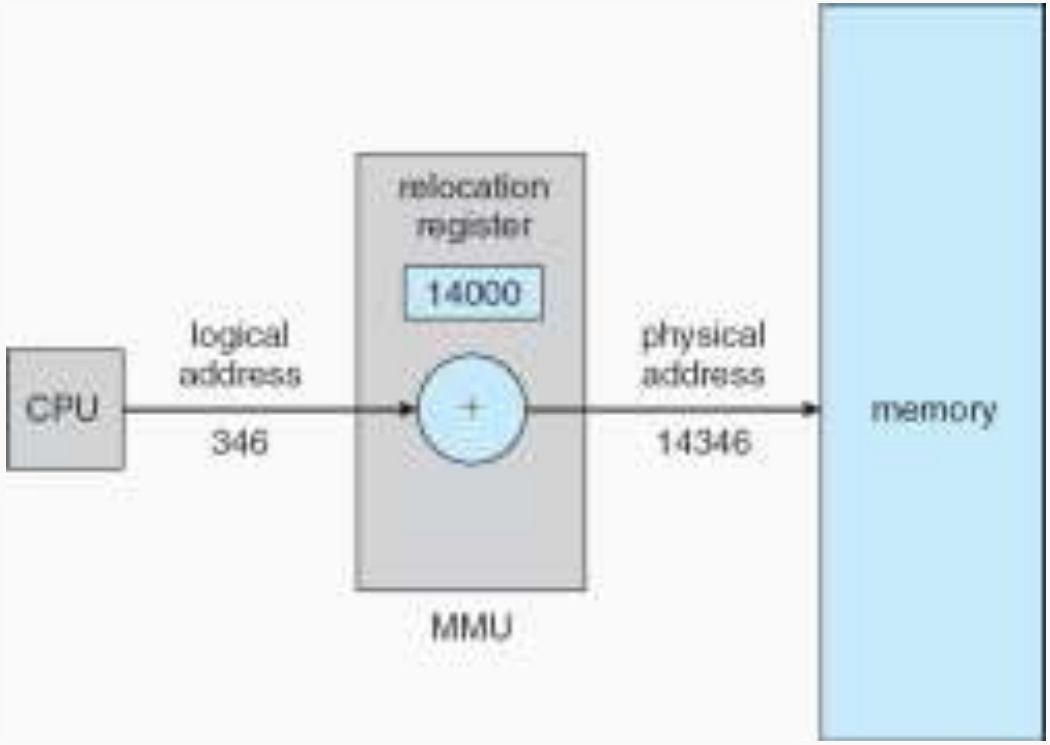
Tipos de direccionamiento

Tipos de direccionamiento

- Se definen varios tipos de direccionamiento:
  - Direccionamiento físico (*physical address*): La unidad de memoria manipula direcciones físicas.
  - Direccionamiento virtual (*virtual address*): Son las direcciones lógicas que se generan cuando existe asociación de direccionamiento en tiempo de ejecución.
- Para la asociación de direcciones en tiempo de compilación o carga, las direcciones lógicas o físicas coinciden. No es así para la asociación en tiempo de ejecución.

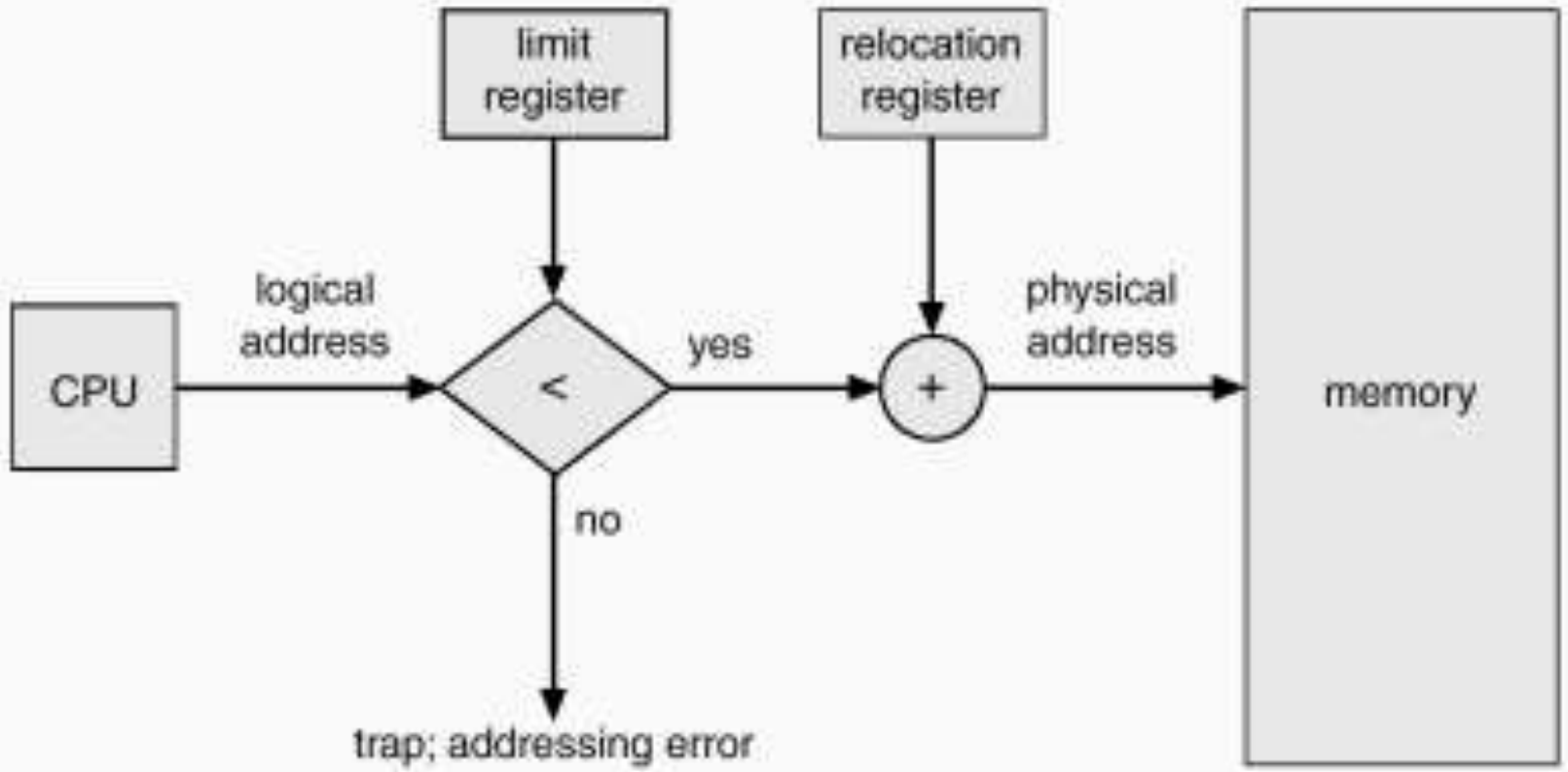
Tipos de direccionamiento

- La traducciones de direcciones lógicas a físicas son hechas por la MMU (Memory Management Unit).
- Los procesos solo manipulan direcciones lógicas y no visualizan las físicas, que solamente son vistas por la MMU.



Protección de memoria

Protección de memoria



Asignación de memoria

Asignación de memoria a nivel del sistema

- La memoria, por lo general, es dividida en dos particiones:
  - Sistema operativo residente.
  - Procesos de usuarios.
- Es necesario un mecanismo de protección de memoria entre los procesos entre sí y el sistema operativo.
- El registro de ubicación (relocation register) y el registro límite son utilizados para realizar la verificación de accesos válidos a la memoria.
- Toda dirección lógica debe ser menor al valor del registro límite.

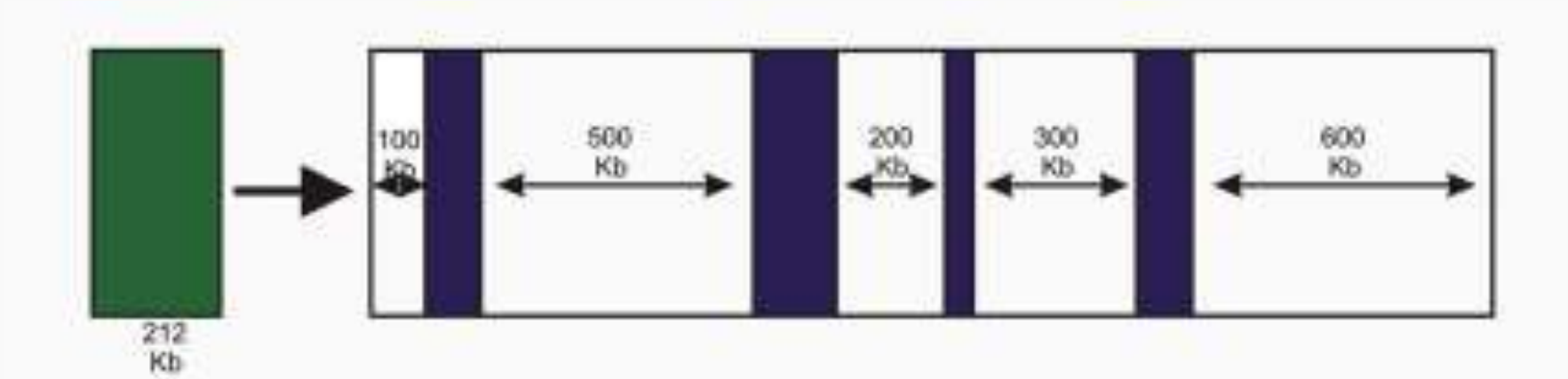
Estructuras para asignación

- El sistema operativo debe llevar cuenta de las particiones ocupadas y libres.
- Los métodos más comunes utilizados son a través de:
    - Mapa de bits.
    - Lista encadenada.

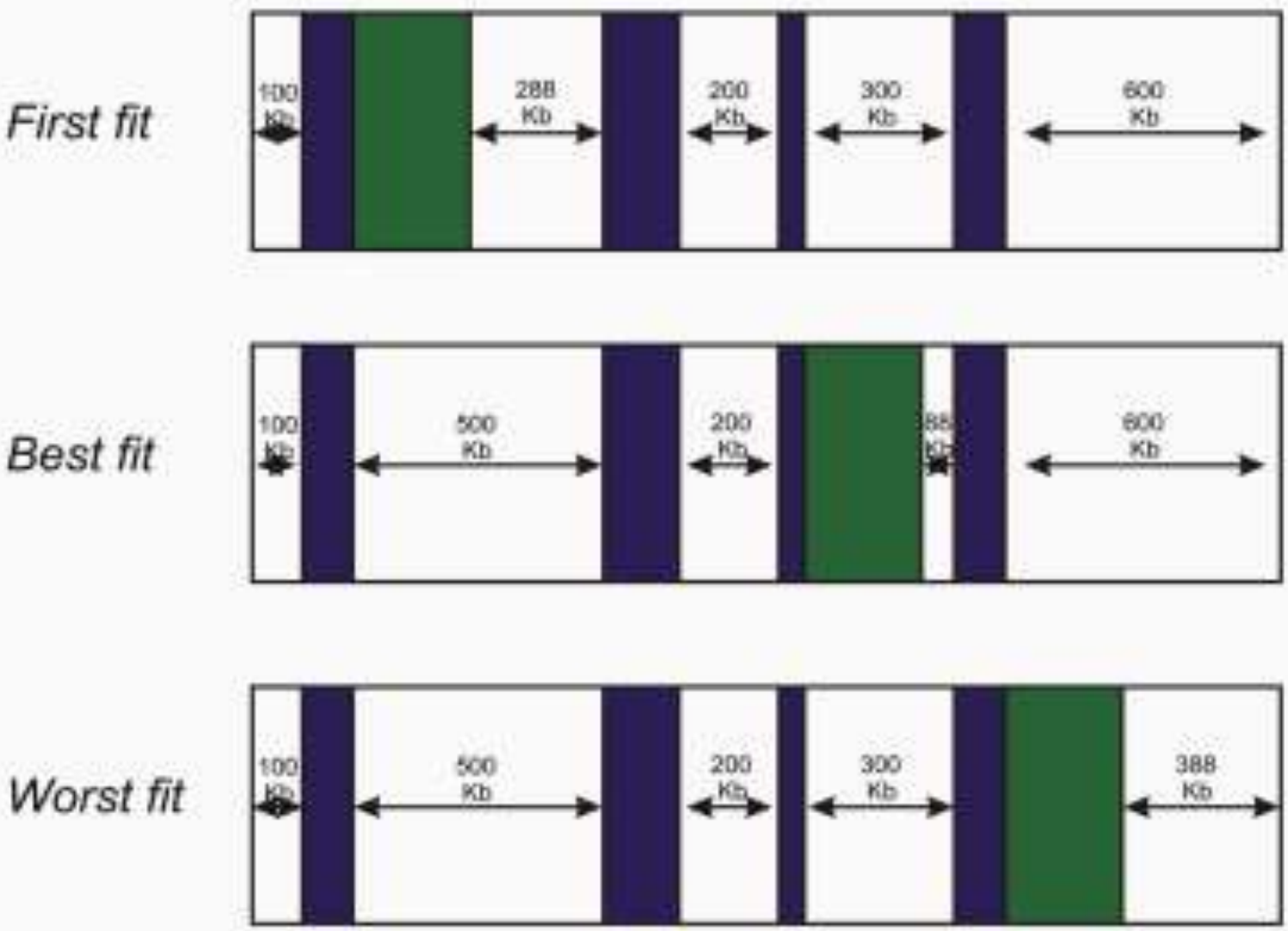
Estrategia de asignación

- En la asignación de memoria a un proceso existen varias estrategias:
  - First fit: Asigna el primer “agujero” de memoria libre que satisface la necesidad.
  - Best fit: Asigna el mejor “agujero” de memoria libre que exista en la memoria principal.
  - Worst fit: Asigna el requerimiento en el “agujero” más grande que exista en la memoria principal.
- Estudios de simulación han mostrado que first fit y best fit lograron mejores rendimientos en tiempo de asignación y utilización de la memoria que la estrategia worst fit.

Estrategia de asignación



Estrategia de asignación



Fragmentación

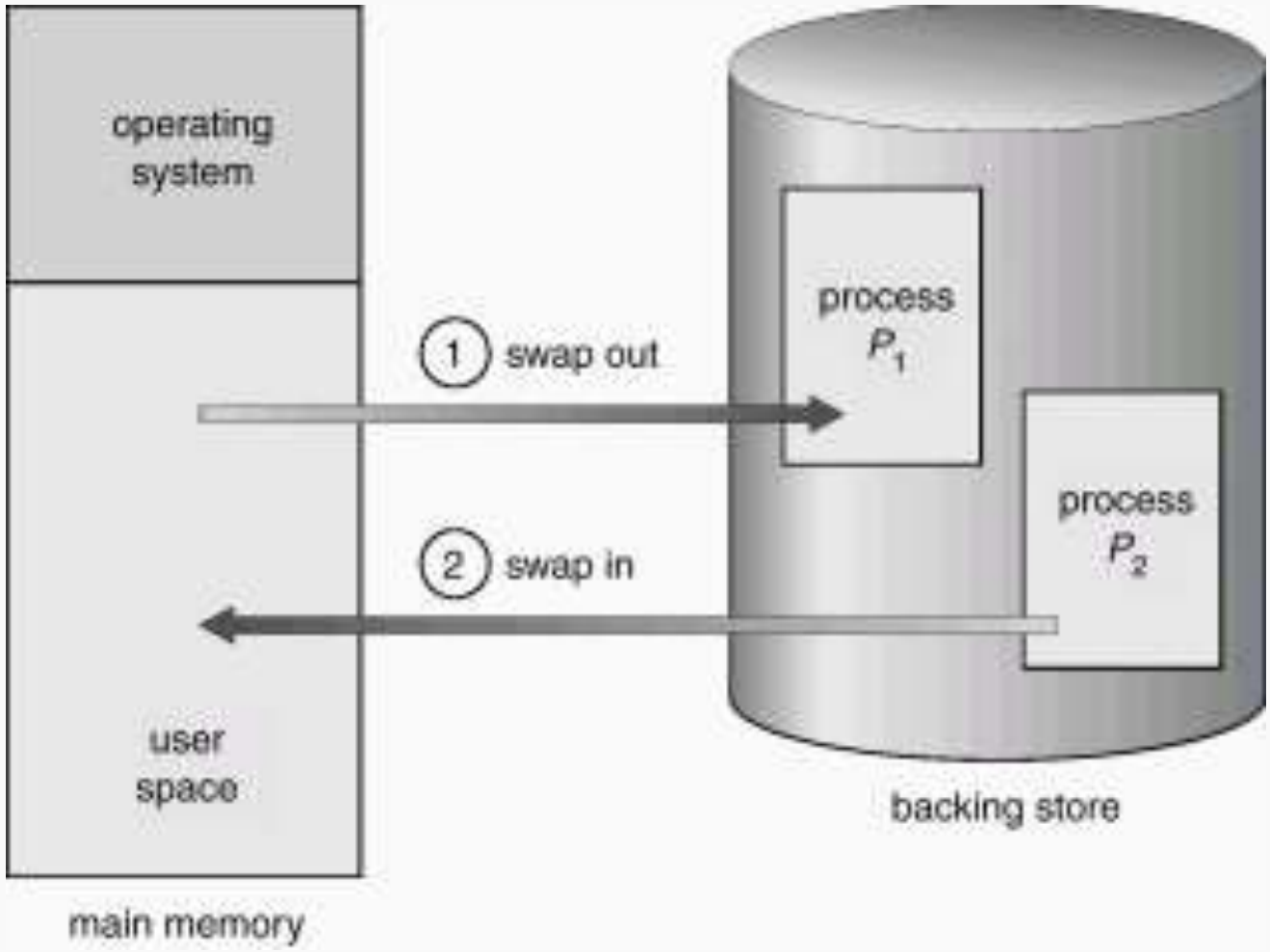
- Las estrategias de asignación presentadas muestran problemas de fragmentación externa.
- En la memoria van quedando una gran cantidad de *agujeros* chicos, que no son asignados. La memoria libre está fragmentada en una gran cantidad *agujeros* chicos.
- La fragmentación externa existe cuando existe suficiente memoria libre en el sistema para satisfacer un requerimiento de memoria, pero no es posible asignarlo debido a que no es contiguo.

Swapping

Swapping

- En sistemas multiprogramados más de un proceso está cargado en memoria principal. Para obtener un mayor nivel de multiprogramación, los procesos que no están ejecutando pueden ser llevados a disco temporalmente.
- El disco (*backing store*) es un espacio donde se dispondrán las imagen de memoria de los procesos.
- Al mecanismo de llevar un proceso desde memoria principal a disco se le denomina swap-out. Al inverso se le denomina swap-in.
- El mayor tiempo consumido en el swapping es el tiempo de transferencia.

Swapping


operating system
1 swap out
2 swap in
user space
main memory
process P_1
process P_2
backing store

Swapping

- El lugar de memoria donde será asignado un proceso en el momento de swap-in depende del método de asociación de direccionamiento (*address binding*) utilizado.
- En la asociación en tiempo de compilación o de carga (*compile*, *load time*) debe ser el mismo lugar, mientras que si la asociación es en tiempo de ejecución la asignación del lugar es libre.