Tema 3. Gestión de la memoria

Competencias:

- ✓ Comprender las funciones que debe desempeñar todo mecanismo que gestiona la memoria, identificando las ventajas e inconvenientes de los distintos esquemas de organización y su ámbito de aplicabilidad.
- ✓ Ser capaz de analizar el rendimiento de un sistema según la organización de memoria que disponga.
- ✓ Ser capaz de llevar a cabo un razonamiento crítico y lógico.
- ✓ Capacidad dialéctica, siendo capaz de argumentar y defender sus decisiones.
- ✓ Conseguir capacidad de abstracción.
- ✓ Ser capaz de enfrentarse a problemas nuevos recurriendo conscientemente a estrategias que han sido útiles en problemas resueltos anteriormente.

Tema 3. Gestión de la memoria

1. Visión general.
    1. Conceptos relacionados con la organización y gestión de la memoria.
    2. Estrategias de administración del almacenamiento.
2. Organización y gestión en sistemas monoprogramados.
    1. Organización y gestión en entornos de monoprogramación.
    2. Protección.
    3. Overlays.
3. Organización y gestión en sistemas multiprogramados.
    1. Introducción. Grado de multiprogramación.
        1. Gestión de memoria con particiones fijas.
        2. Gestión de memoria con particiones variables.
    2. Multiprogramación con intercambio a disco (Swapping).
4. Organización de la memoria virtual.
    1. Conceptos básicos sobre memoria virtual.
    2. Paginación.
    3. Segmentación.
    4. Paginación/segmentación.

    } **(Actividad 3)**
5. Gestión de la memoria virtual: Paginación.
    1. Algoritmos de sustitución de página.
    2. Estrategias de búsqueda.
    3. Evaluación de los sistemas paginados.
        1. Liberación de página.
        2. Tamaño de página.
        3. Localidad.

1. Visión general (I)

1.1. Conceptos relacionados con la organización y gestión de la memoria.

Funciones:

1. *Reubicación*. El espacio de memoria asignado a un trabajo puede cambiar durante la vida del mismo, y ello conlleva realizar en cada momento la transformación de las direcciones lógicas indicadas por el programador a direcciones físicas.
2. *Protección*. Evitar que un proceso altere el contenido de las posiciones asignadas a otros trabajos.
3. *Utilización compartida*. Permitir que varios procesos puedan acceder a la misma zona de memoria de forma controlada.
4. *Organización lógica*. Reflejar la estructura lógica de los programas mediante una división en bloques del espacio de direcciones. Ventajas:
   - codificar independientemente los diferentes bloques que componen los programas;
   - proporcionar fácilmente distintos grados de protección a los diferentes bloques;
   - introducir mecanismos que permitan que diferentes procesos compartan bloques de memoria.
5. *Organización física*. Organizar el almacenamiento a dos niveles.

1. Visión general (II)

1.2. Estrategias de administración de almacenamiento.

1. *Estrategias de búsqueda:* deciden **cuándo** obtener el siguiente fragmento de programa, o de datos, para insertarlo en la memoria principal.
   - Estrategias de búsqueda por demanda: el momento viene dado cuando algún programa en ejecución hace referencia al citado fragmento;
   - Estrategias de búsqueda anticipada: predicen por donde irá el control del programa en el paso siguiente para decidir el momento.

2. *Estrategias de colocación:* deciden **dónde** va a ser colocado un nuevo programa, o un trozo de programa o de datos.

2. *Estrategias de reposición:* deciden **qué** fragmento de programa, o de datos, se va a desplazar, para dejar sitio a los programas nuevos o a otros fragmentos.

2. Organización y gestión en sistemas monoprogramados (I)

2.1. Organización y gestión en entornos de monoprogramación.

Memoria Principal

El tamaño del programa está limitado por la cantidad de memoria principal menos el espacio ocupado por el S. O..

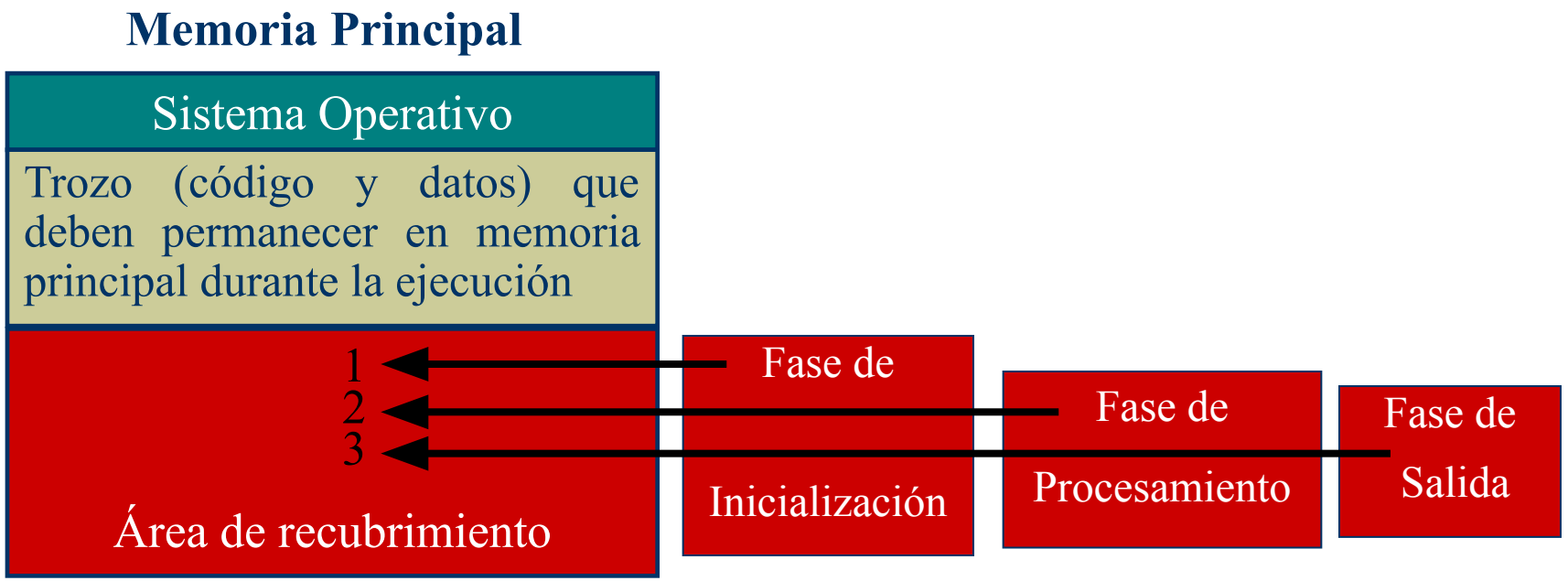
2.2. Protección.

La protección del S. O. se realiza mediante un *registro de límites,* perteneciente a la CPU, que contiene la dirección más alta (o más baja) utilizada por el S. O..

2. Organización y gestión en sistemas monoprogramados (II)

2.3. Overlays.

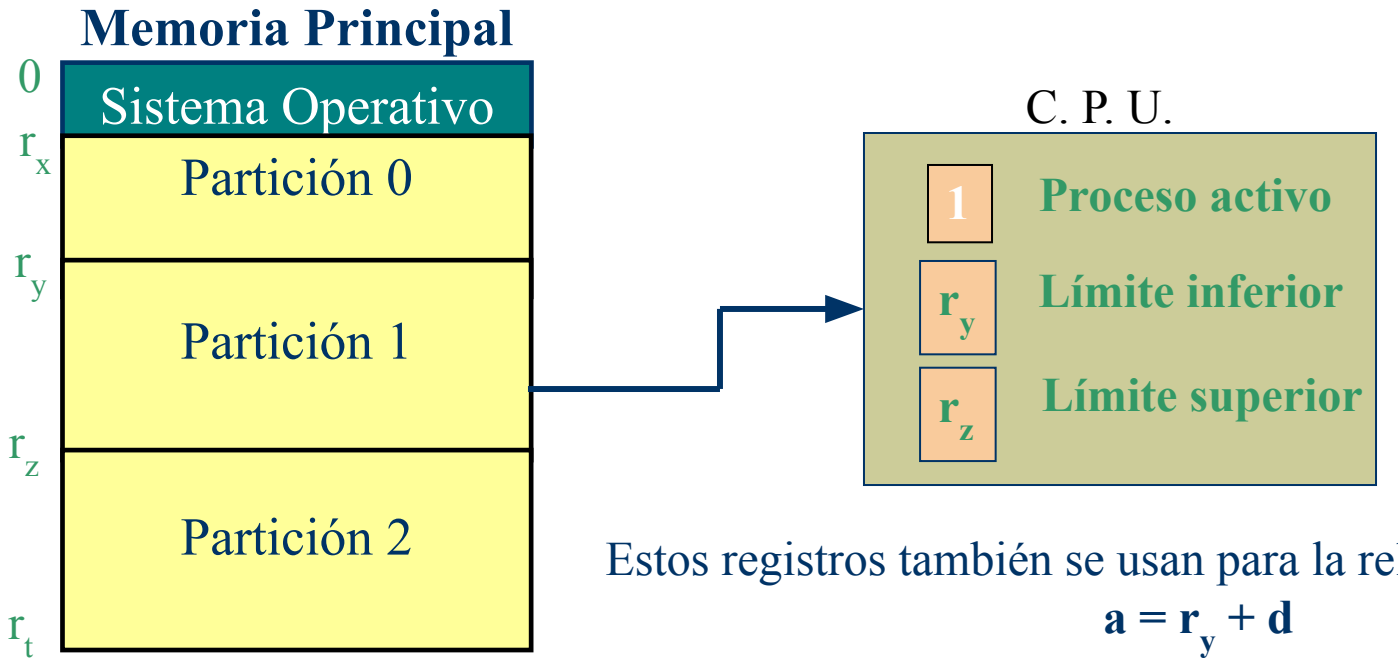
Si se desea ejecutar un trabajo cuyo tamaño es mayor que el espacio disponible de memoria principal, será necesario usar *programación con overlays* (recubrimientos), que consiste en dividir el trabajo en capas independientes, de forma que en la memoria principal solo se tenga la capa que se está ejecutando en ese momento.



3. Organización y gestión en sistemas multiprogramados (I)

3.1. Introducción. Grado de multiprogramación.

La memoria principal se divide en una serie de particiones, conteniendo cada partición un trabajo completo. Estas particiones se protegen mediante los *registros de límites*.



Estos registros también se usan para la relocalización:

$$\mathbf{a} = \mathbf{r}_y + \mathbf{d}$$

a: dirección absoluta; $\mathbf{r}_y$: base; **d**: dirección relativa

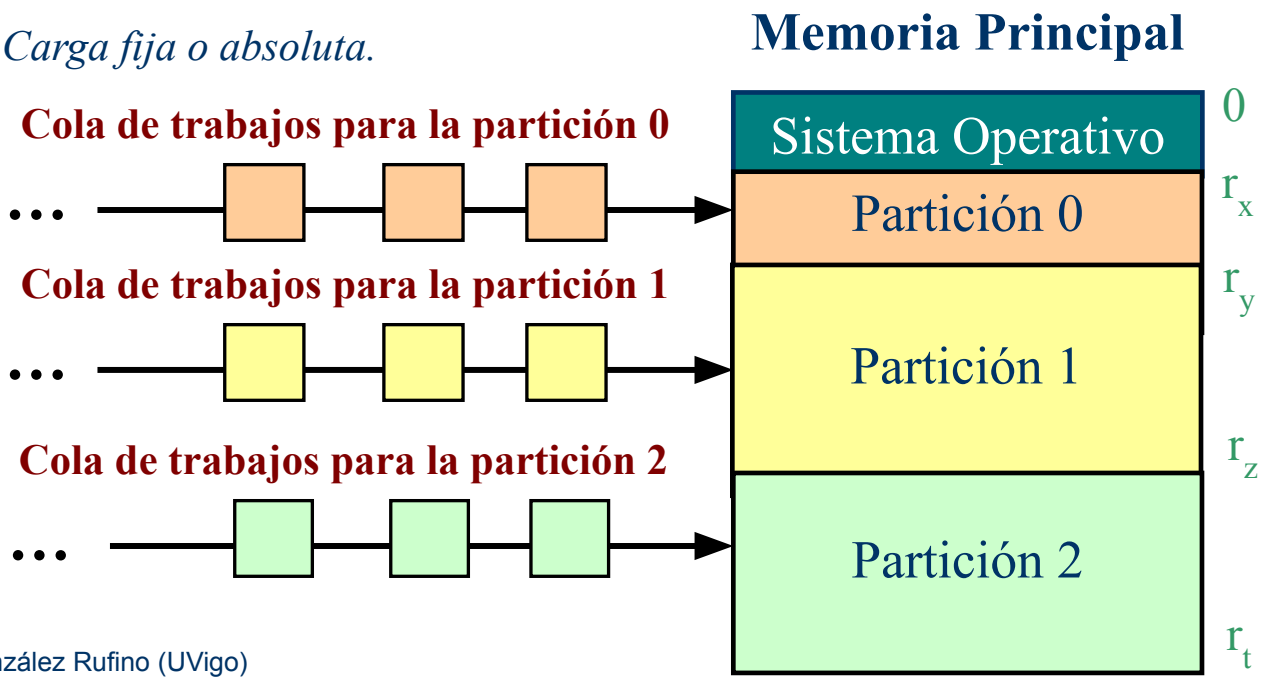
3. Organización y gestión en sistemas multiprogramados (II)

3.1.1. Gestión de memoria con particiones fijas. (I)

Las particiones son de tamaño fijo, conteniendo cada partición un sólo trabajo completo.

Modos de carga:

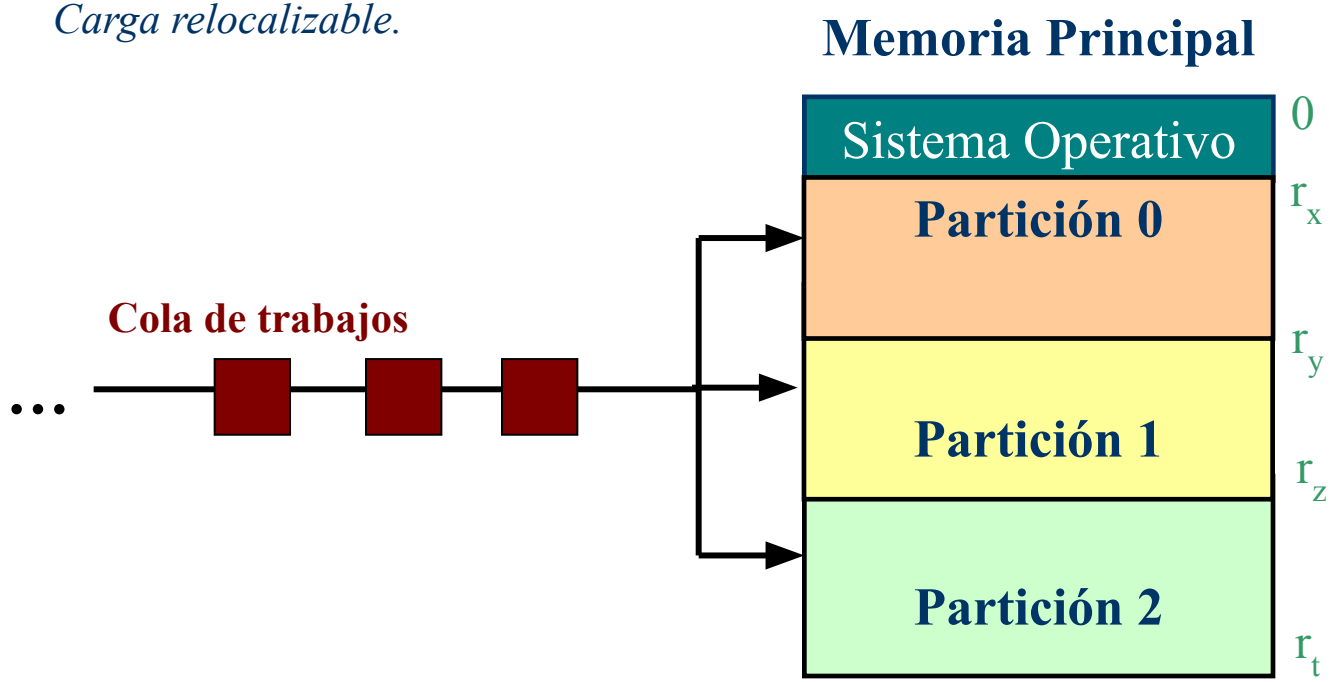
- *Carga fija o absoluta.*



3. Organización y gestión en sistemas multiprogramados (III)

3.1.1. Gestión de memoria con particiones fijas. (II)

- *Carga relocalizable.*

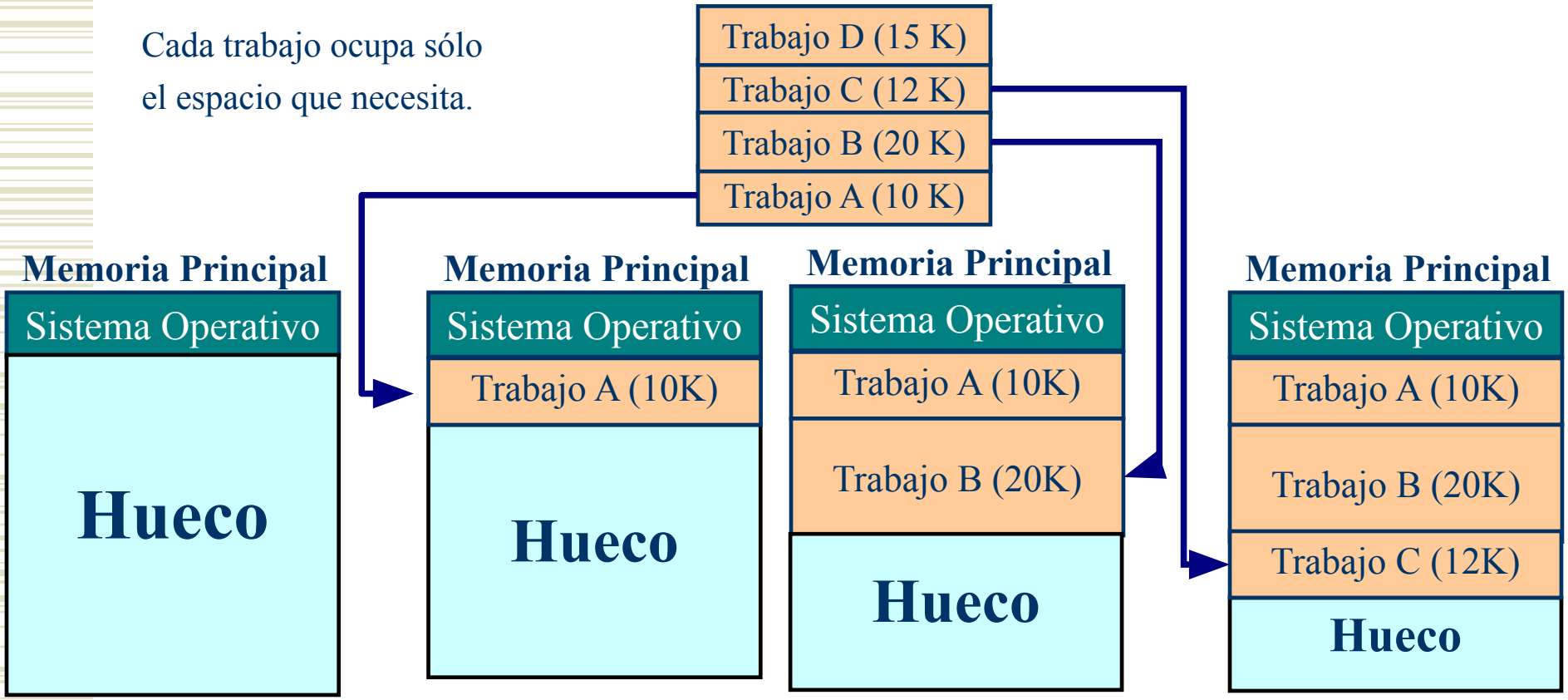


Problema: *desperdicio de memoria* debido a que el tamaño de las particiones esta establecido a priori (*fragmentación interna*).

3. Organización y gestión en sistemas multiprogramados (IV)

3.1.2. Gestión de memoria con particiones variables. (I)

Cada trabajo ocupa sólo el espacio que necesita.

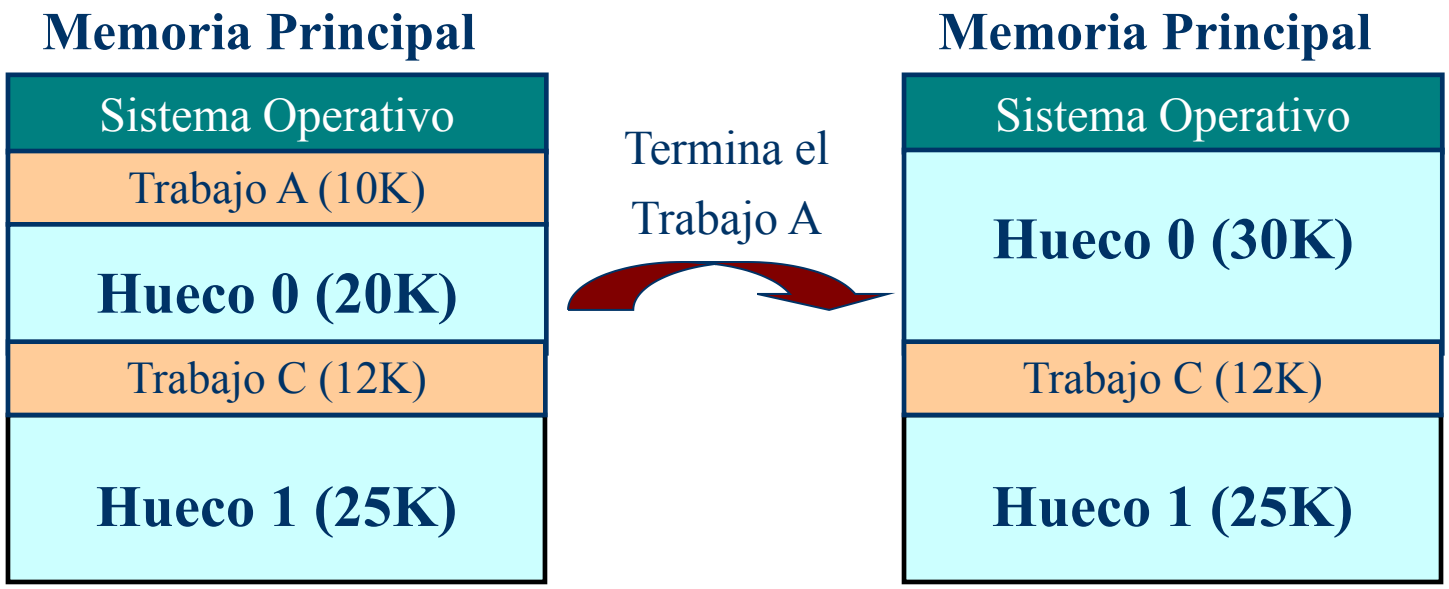


3. Organización y gestión en sistemas multiprogramados (V)

3.1.2. Gestión de memoria con particiones variables. (II)

Técnicas:

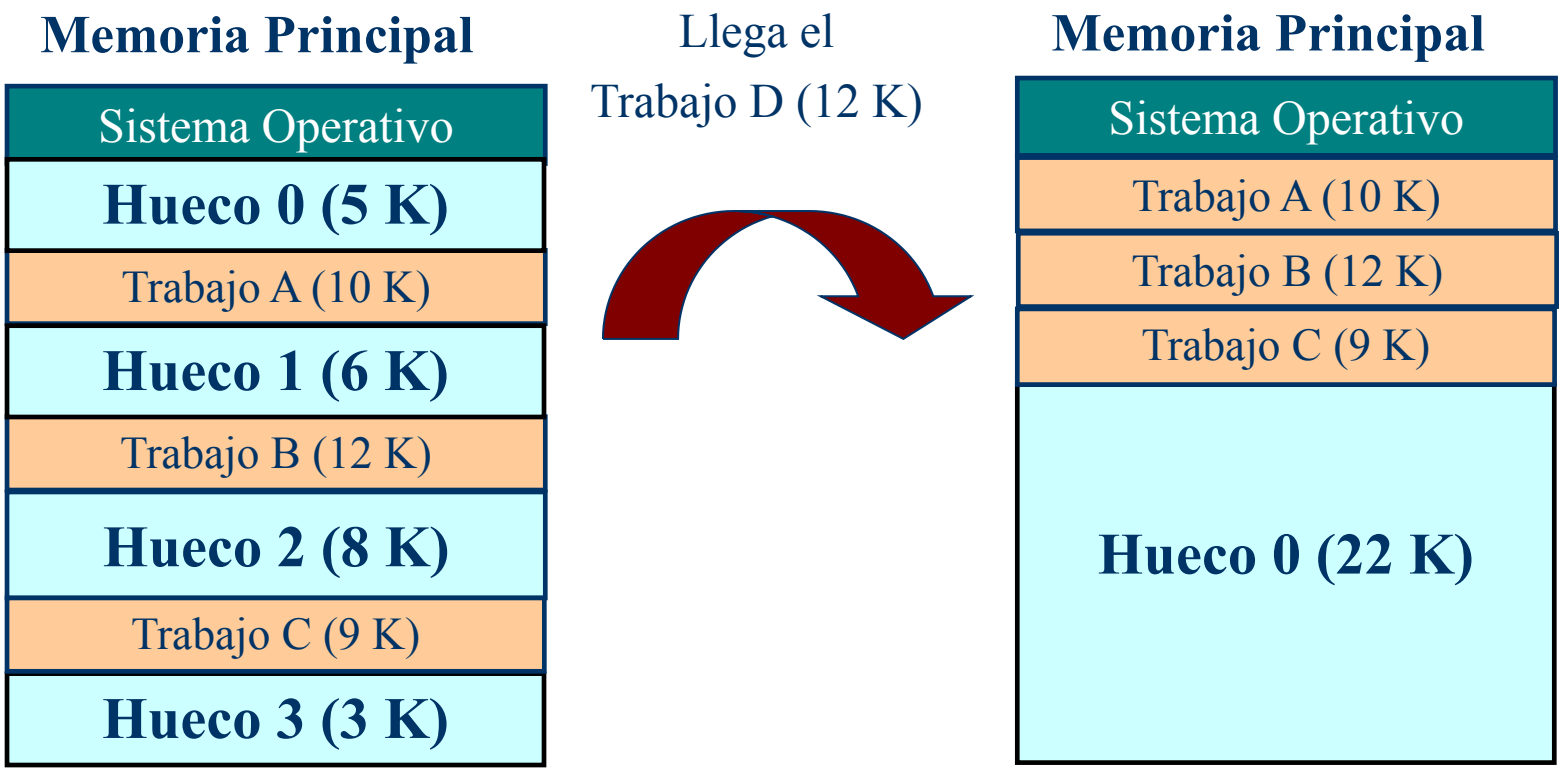
➢ *Combinación de huecos:*



3. Organización y gestión en sistemas multiprogramados (VI)

3.1.2. Gestión de memoria con particiones variables. (III)

➢ *Compactación de memoria:*

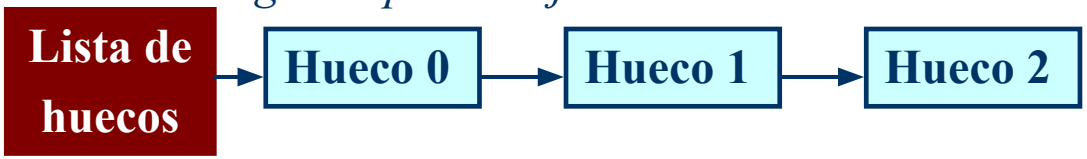


3. Organización y gestión en sistemas multiprogramados (VII)

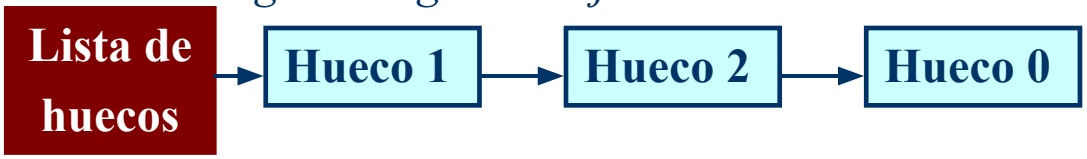
3.1.2. Gestión de memoria con particiones variables. (IV)

Estrategias de colocación:

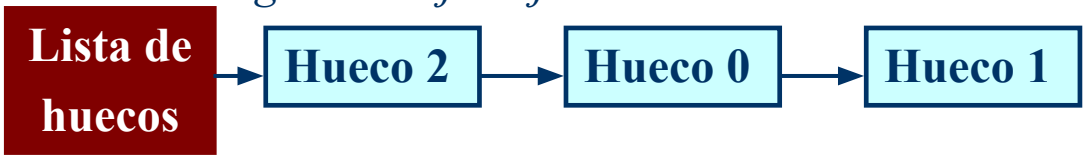
1. *Estrategia de primer ajuste*:



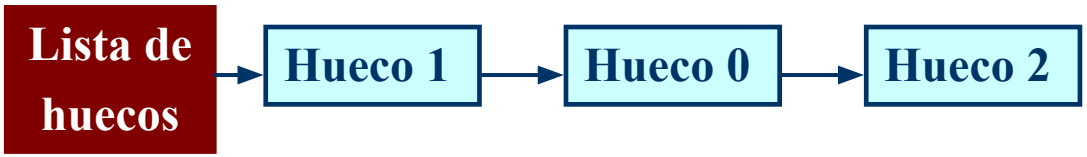
2. *Estrategia de siguiente ajuste*:



3. *Estrategia de mejor ajuste*:



4. *Estrategia de peor ajuste*:



3. Organización y gestión en sistemas multiprogramados (VIII)

3.2. Multiprogramación con intercambio a disco (Swapping).

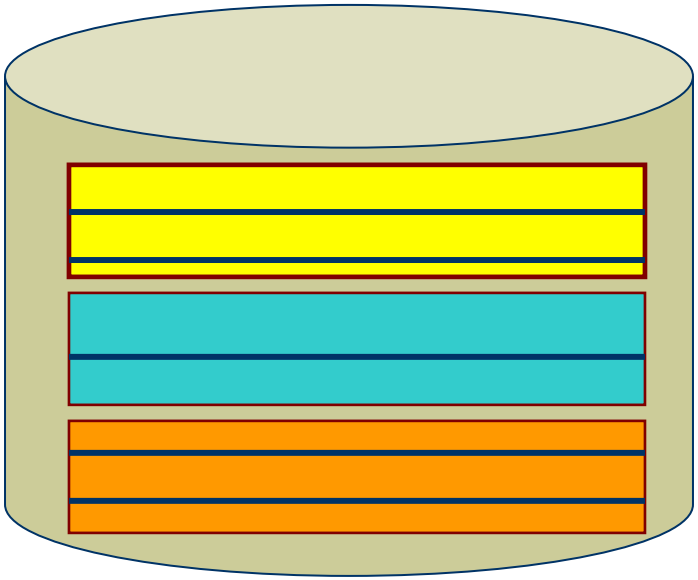
Se trabaja con los dos niveles de almacenamiento para no limitar el número de procesos, manteniendo los trabajos de los procesos que no están activos en memoria secundaria.

Para conseguir esto, se catalogan los procesos entre los que deben ser residentes en memoria principal y los que pueden pasar a disco. Entre estos últimos se puede usar prioridades; de forma que un trabajo no pasará a disco si existe otro cuya prioridad le obliga a que pase antes.

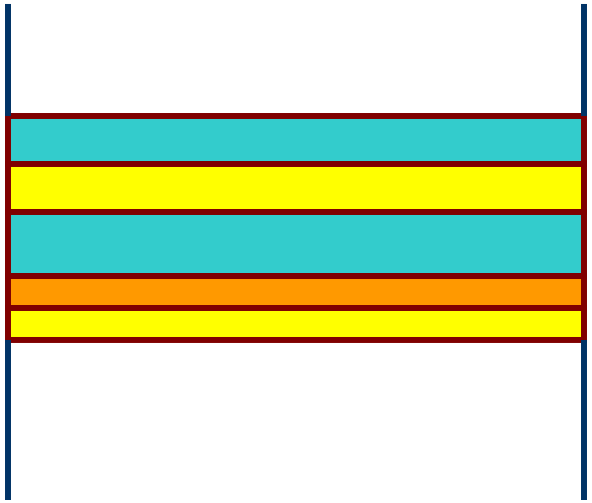
4. Organización de la memoria virtual (I)

4.1. Conceptos básicos sobre memoria virtual (I).

Definición: *memoria virtual* (Fotheringham, 1961) es un método que permite direccionar un espacio de almacenamiento mucho mayor que el disponible en la memoria principal. Para ello, el S. O. usa memoria a dos niveles y particiona los trabajos en bloques, manteniendo en memoria principal aquellos bloques que se están usando.



Memoria secundaria



Memoria principal

4. Organización de la memoria virtual (II)

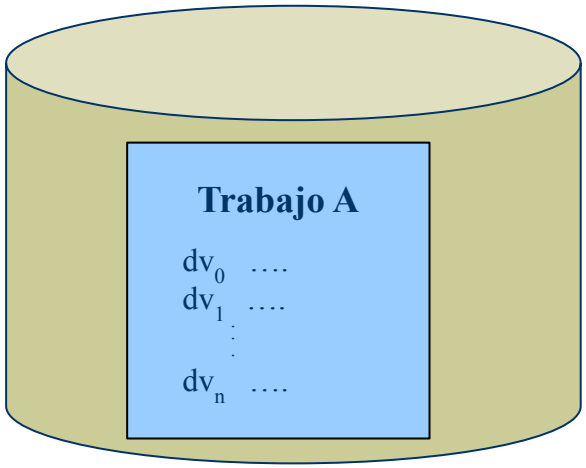
4.1. Conceptos básicos sobre memoria virtual (II).

Tamaño de los bloques:

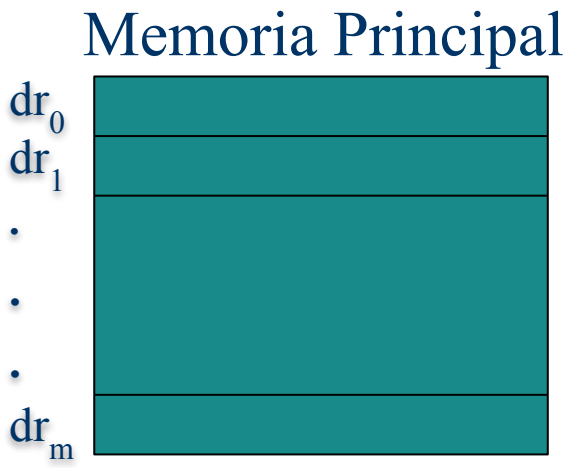
- con el mismo tamaño: páginas. ***Paginación.***
- con diferentes tamaños: segmentos. ***Segmentación.***

Segmentos de tamaño variable de páginas de tamaño fijo. ***Paginación/Segmentación.***

Direcciones virtuales: las direcciones a las que se refiere un proceso durante su ejecución. *Espacio de direcciones virtuales*.



Direcciones reales: las direcciones que tiene la memoria principal. *Espacio de direcciones reales*.



4. Organización de la memoria virtual (II)

4.1. Conceptos básicos sobre memoria virtual (III).

Los mecanismos de *traducción dinámica de direcciones* (DAT) se encargan de transformar las direcciones virtuales a direcciones reales, basándose en que direcciones contiguas dentro del espacio de direcciones virtuales, no tienen porqué ser contiguas dentro del almacenamiento real.

$$dv_y \longrightarrow dr_x$$

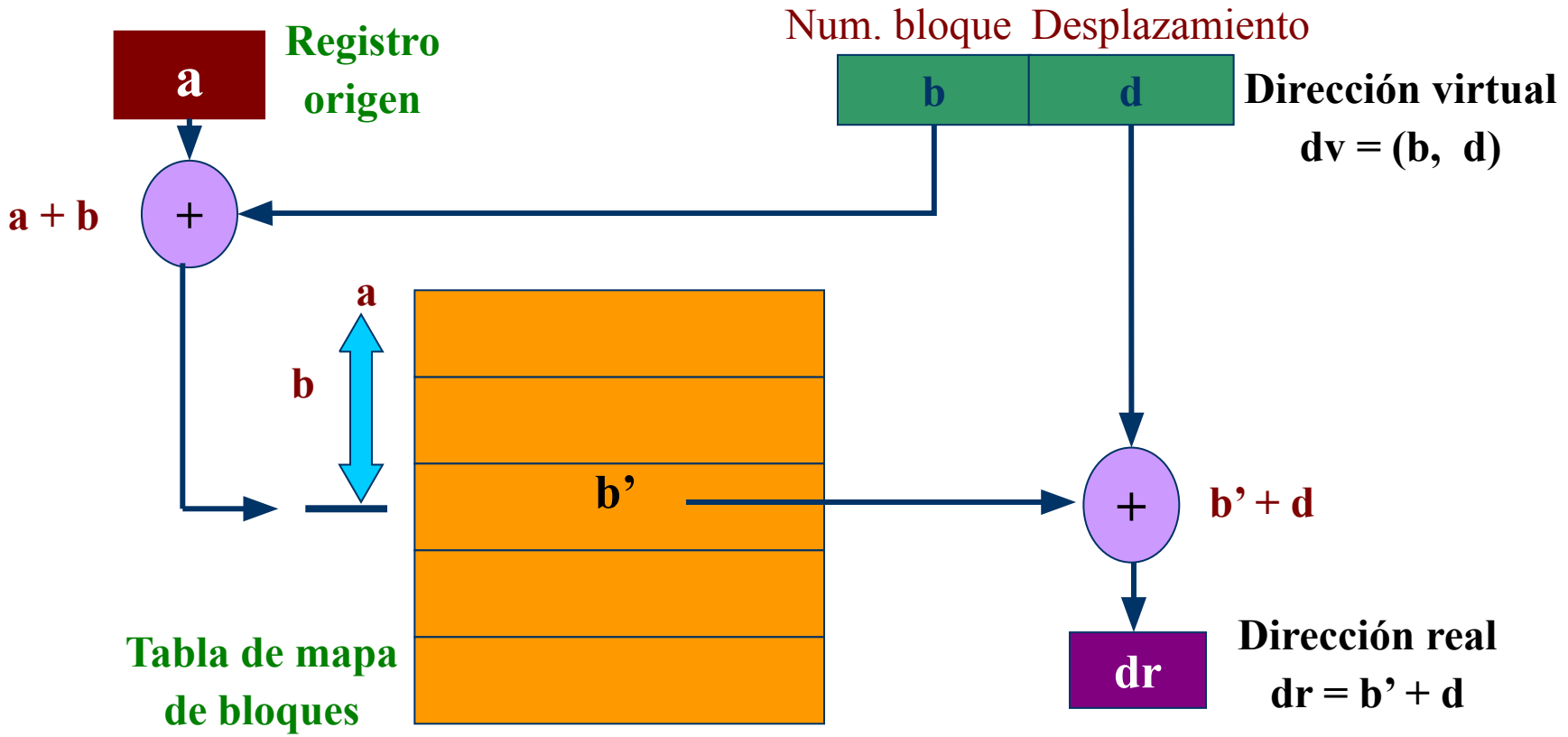
$$dv_{(y+1)} \longrightarrow dr_z \qquad dr_z \neq dr_{(x+1)}$$

Para realizar esta transformación, los DAT mantienen por cada trabajo un mapa que indica qué direcciones virtuales se encuentran en memoria principal y dónde. Para que este mapa no sea muy grande las transformaciones se realizan a nivel de bloque.

4. Organización de la memoria virtual (III)

4.1. Conceptos básicos sobre memoria virtual (IV).

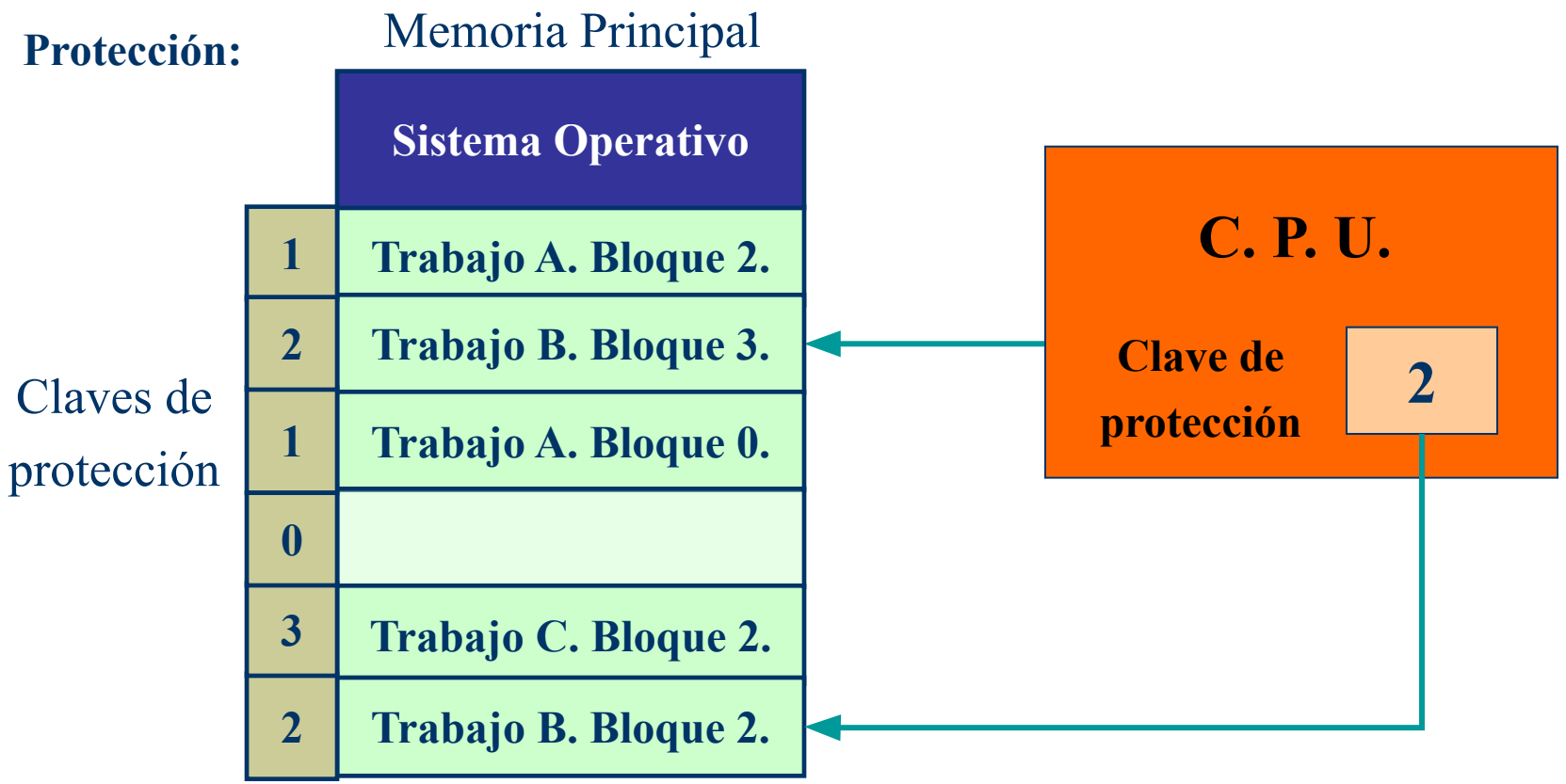
Traducción de una dirección virtual a una dirección real:



4. Organización de la memoria virtual (IV)

4.1. Conceptos básicos sobre memoria virtual (V).

Protección:



5. Gestión de la memoria virtual: Paginación (I)

5.1. Algoritmos de sustitución de página (I).

Principio de optimización: para obtener un rendimiento óptimo se debe desplazar aquella página que no se va a usar en el futuro durante el período de tiempo más largo.

Estrategias:

A) *Reposición de página al azar*.

Se elige al azar la página a desplazar, por lo tanto supone poca sobrecarga y no es discriminatoria.

B) *Reposición por el sistema de primero en entrar – primero en salir (FIFO).*

Se desplaza la página que lleva más tiempo almacenada en memoria principal. Se basa en que la página que lleva menos tiempo es porque se ha referenciado hace poco, y por tanto la que lleva más tiempo es porque ya no se usa.

C) *Reposición de página menos – recientemente – usada (LRU).*

Se desplaza la página que no ha sido usada durante el mayor período de tiempo. Para ello, es necesario ponerle un sello de tiempo a la página cada vez que sea referenciada, suponiendo una sobrecarga adicional importante.

5. Gestión de la memoria virtual: Paginación (II)

5.1. Algoritmos de sustitución de página (II).

D) *Reposición de página menos – frecuentemente – usada (LFU).*

Se desplaza la página que se ha usado con menos frecuencia, por lo tanto cada página posee un contador de utilización.

E) *Reposición de página no usada – recientemente (NUR).*

Se basa en que:

- ✓ la página que no ha tenido un uso reciente tiene poca probabilidad de ser usada en el futuro y puede por tanto ser reemplazada;
- ✓ la sobrecarga es menor si se reemplaza una página que no ha cambiado desde que se cargó en memoria principal, ya que no es necesario llevarla al almacenamiento secundario.

Para llevar a cabo esta estrategia cada página va a tener dos bits hardware: *bit referenciado* y *bit modificado* o *bit sucio*. La estrategia NUR elige la página a desplazar siguiendo el orden de los siguientes grupos:

1º *no referenciado no modificado*
2º *referenciado no modificado*
3º *referenciado modificado*

Cada cierto tiempo se ajusta todos los bits de referencia a 0, apareciendo un nuevo grupo *no referenciado modificado.*

5. Gestión de la memoria virtual: Paginación (III)

5.2. Estrategias de búsqueda (I).

1. *Paginación por demanda*.

   Ninguna página debe ser traída del almacenamiento secundario al principal hasta que no sea referenciada de forma explícita por el proceso en ejecución.

   Razones:

   - el camino que tomará la ejecución de un programa no se puede predecir con exactitud,
   - se garantiza que las páginas que se llevan a la memoria principal son realmente las únicas que los procesos necesitan;
   - la sobrecarga es inexistente.

   Inconveniente: el proceso debe esperar a que se transfiera a la memoria principal una a una cada página nueva que referencia.

5. Gestión de la memoria virtual: Paginación (IV)

5.2. Estrategias de búsqueda (II).

2. *Paginación anticipada.*

   El S. O. intenta predecir las páginas que un proceso va a necesitar y precargar estas páginas cuando hay espacio disponible, de forma que mientras que el proceso ejecuta sus páginas actuales, el sistema carga nuevas páginas que estarán disponibles cuando el proceso las pida. Se intenta reducir el tiempo de espera de los procesos.

5.3. Evaluación de los sistemas paginados (I).

5.3.1. Liberación de página.

Cuando se hace evidente que una página ya no se necesita se debe liberar el marco de página donde está contenida.

Formas:

a) de forma voluntaria: la liberación es realizada por el usuario mediante mandatos;

b) de forma automática: la responsabilidad recae en los compiladores y sistemas operativos.

5. Gestión de la memoria virtual: Paginación (V)

5.3. Evaluación de los sistemas paginados (II).

5.3.2. Tamaño de página.

	Fragmentación de tablas	Fragmentación interna	Número de transferencias	Desperdicio (información no usada)
Páginas pequeñas				
Páginas grandes				

5. Gestión de la memoria virtual: Paginación (VI)

5.3. Evaluación de los sistemas paginados (III).

5.3.3. Localidad.

Definición: la *localidad* se refiere al hecho de que los procesos tienden a hacer referencias en patrones no uniformes pero si muy localizados.

Tipos:

1. *localidad temporal*: la zona de memoria referenciada recientemente tiene una alta probabilidad de ser referenciada en un futuro próximo;
2. *localidad en el espacio*: una vez hecha una referencia a una zona determinada, es probable que las zonas cercanas también sean referenciadas.

Consecuencia: un programa puede ser ejecutado eficientemente, si se tiene en memoria principal un subconjunto de páginas de dicho programa que son las preferidas.

Teoría del conjunto de trabajo del comportamiento de un programa (Denning): es el conjunto de páginas a las cuales el proceso hace referencia activamente.

Este conjunto de trabajo cambia durante la ejecución del proceso, por lo tanto cualquier suposición sobre el tamaño y el contenido del conjunto de trabajo inicial de un proceso no se puede aplicar a los sucesivos conjuntos de trabajo de dicho proceso.