

Memoria virtual

E. Campo

M. Knoblauch

Ó. López

J. Clemente

Departamento de Automática
Universidad de Alcalá



```
/gso>
```

Índice

1 Introducción a la memoria virtual (MV)

- ¿Qué es la MV?
- Ventajas de la MV

2 Conceptos relacionados con la MV

- Requisitos hardware de la MV
- Carga dinámica
- Paginadores
- Hiperpaginación

3 Algoritmos de gestión de la MV

- Algoritmos de gestión de memoria
- Políticas de asignación
- Políticas de ubicación
- Políticas de búsqueda o de lectura
- Políticas de reemplazo

4 Casos de estudio

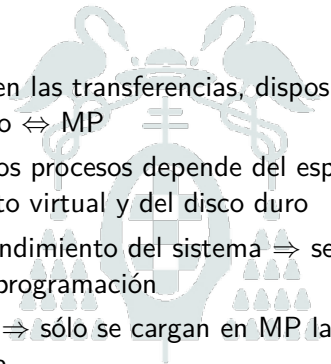
- Mach 3.0
- Windows
- Linux



¿Qué es la MV?

- Es un esquema de gestión de memoria en el que los procesos:
 - Se ejecutan sin estar completamente cargados en memoria principal (MP)
 - Pueden tener mayor tamaño que la MP disponible
- Permite un desacoplamiento entre el espacio de direcciones físicas y el espacio de direcciones virtual
- Utiliza el almacenamiento secundario como extensión de la MP
- Basada en el principio de localidad de las referencias $\Rightarrow$ Sólo se mantiene en MP la información necesaria en cada momento
- El control lo realiza el sistema operativo (SO) con la ayuda imprescindible del hardware

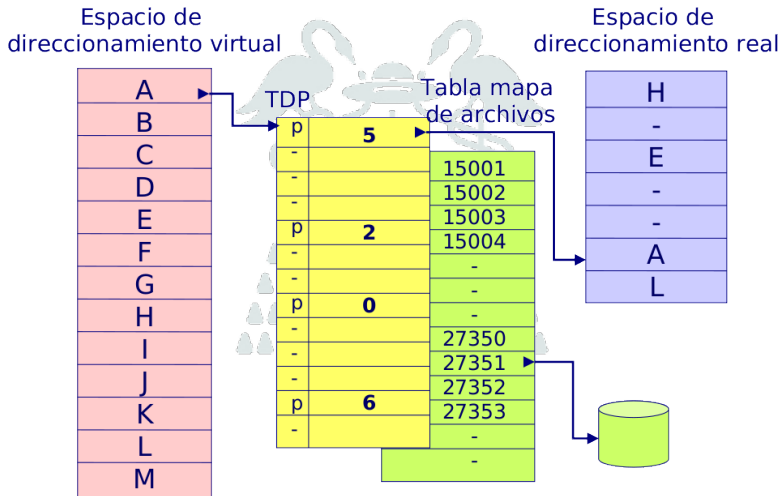
Ventajas de la MV

- 
- 1 Transparencia en las transferencias, dispositivo de almacenamiento $\Leftrightarrow$ MP
 - 2 El tamaño de los procesos depende del espacio de direccionamiento virtual y del disco duro
 - 3 Se mejora el rendimiento del sistema $\Rightarrow$ se incrementa el grado de multiprogramación
 - 4 Reduce la E/S $\Rightarrow$ sólo se cargan en MP las partes necesarias de un programa

Requisitos hardware de la MV

- Se utilizan mecanismos de paginación y segmentación
- Ventaja de la paginación:
 - Transferencias más simples con bloques de tamaño fijo
Disco $\Leftrightarrow$ MP
- Hardware de paginación:
 - Unidad de gestión de la tabla del mapa de páginas
 - Bits en los descriptores de páginas de: presencia, modificación (*dirty bit*) y referencia
 - Almacenamiento auxiliar para las páginas del proceso
 - Soporte para interrumpir instrucciones

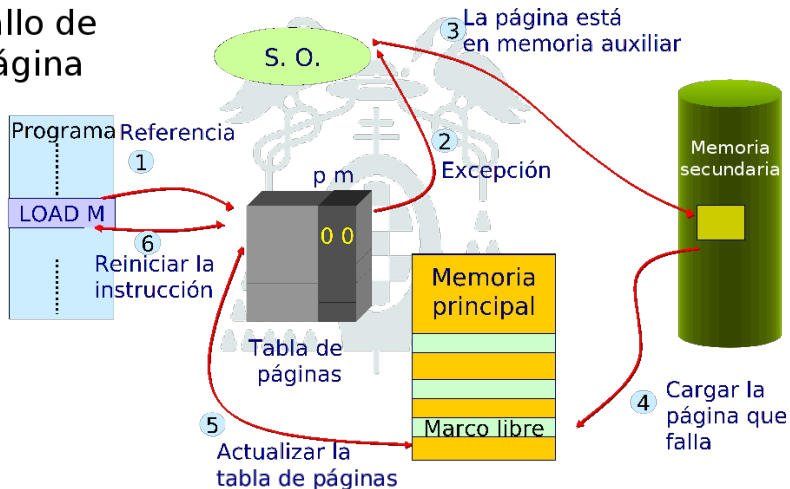
Requisitos hardware de la MV



Carga dinámica

- Transferencia de páginas de dispositivo de almacenamiento a MP
- Si la página referenciada no está disponible en MP $\Rightarrow$ Fallo de página (FP)
 - La tasa de fallos de página disminuye al aumentar el número de marcos
- Los tiempos que más afectan a la carga dinámica son:
 - Cambios de contexto
 - Guardar una página modificada en disco (*page out*)
 - Cargar una página referenciada en MP (*page in*)
- Mientras se realiza, el proceso está bloqueado

Fallo de página

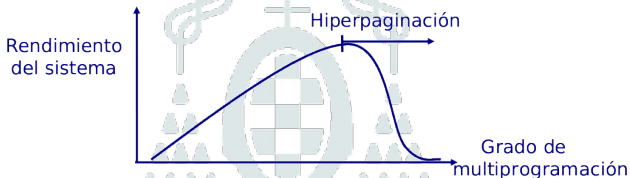


Paginadores

- Parte del SO que mueve páginas entre disco y MP
- Rutinas para hacer la transferencia cuando se produce un FP
- Se crean y destruyen con el objeto proyectado en MV
- Tipos de paginadores
 - De archivos
 - P.ej.: `mmap`, `exec`
 - De objetos anónimos o *swap pager*
Aquellos que no tienen una imagen en el sistema de archivos
 - Gestión del área de *swap*
(área de memoria persistente para los objetos anónimos)
 - De dispositivos
 - P.ej.: gestión del *frame buffer*
 - Se proyecta la zona de memoria utilizada por el dispositivo (no en el disco)

Hiperpaginación, vapuleo o *thrashing*

- Caída de rendimiento al aumentar el grado de multiprogramación
- Cuando los procesos intercambian páginas disco $\Leftrightarrow$ MP

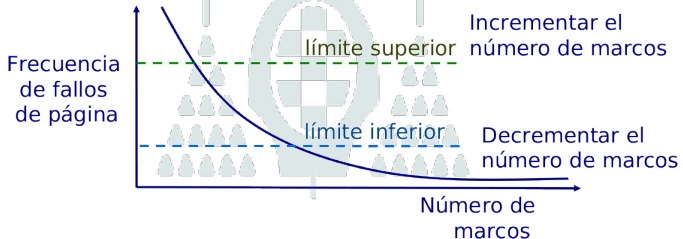


- Soluciones:
 - Reducir grado de multiprogramación
 - Utilizar un algoritmo de reemplazo local
 - Suministrar a cada proceso el número de marcos que necesite
 - Modelo del conjunto de trabajo
 - Frecuencia de fallos de página

Soluciones a la hiperpaginación

Frecuencia de fallos de página

- Se establece un umbral superior y otro inferior, para quitar o asignar marcos de página a un proceso



Algoritmos de gestión de memoria (i)

- Objetivo $\Rightarrow$ Minimizar el porcentaje de FPs, con mínima sobrecarga y máximo aprovechamiento de la MP
- Políticas de asignación
 - ¿Qué cantidad de MP se asigna a un determinado proceso?
 - Asignación fija y asignación variable
 - Alcance del reemplazo: global y local
 - Gestión del espacio libre
- Políticas de ubicación
 - ¿Dónde se ubica un bloque en MP?
- Políticas de búsqueda o lectura
 - ¿Cuándo y qué páginas se cargan en MP?
 - Paginación por demanda
 - Paginación anticipada o prepaginación

Algoritmos de gestión de memoria (ii)

- Políticas de reemplazo
¿Qué páginas deben sustituirse en MP?
 - Algoritmo óptimo
 - Algoritmo primera página en llegar/primera en salir *First In-First Out (FIFO)*
 - Algoritmo página menos usada recientemente - *Least Recently Used (LRU)*
 - Algoritmos de aproximación al LRU: reloj global, FIFO segunda oportunidad, página no usada frecuentemente - *Not Frequently Used (NFU)*

Políticas de asignación (i)

- Determinan qué cantidad de MP se asigna a un proceso según sus necesidades
- Asignación fija
El número de marcos se decide en la carga inicial y está determinado por el tipo de proceso
- Asignación variable
El número de marcos cambia a lo largo de la vida de un proceso
 - Modelo del conjunto de trabajo
 - Frecuencia de FPs

Políticas de asignación (ii)

- Alcance del reemplazo
 - Global: Considera todos los marcos de MP como candidatos, independientemente del proceso al que pertenezcan
 - Ventaja: Mejor aprovechamiento de la MP
 - Inconveniente: Hiperpaginación
 - Ejemplo: Unix
 - Local: Considera los marcos del proceso que originó el FP
 - Ventaja: El número de FPs es más determinista
 - Inconveniente: Mayor sobrecarga $\Rightarrow$ Cálculo de los marcos a asignar en cada instante
 - Ejemplos: VMS y Windows

Políticas de asignación (iii)

- Gestión del espacio libre
 - El SO mantiene el estado de los marcos libres o asignados
 - Mapa de bits
 - Listas enlazadas (Windows)
 - Sistema de colegas o *buddy* (Linux, Unix)

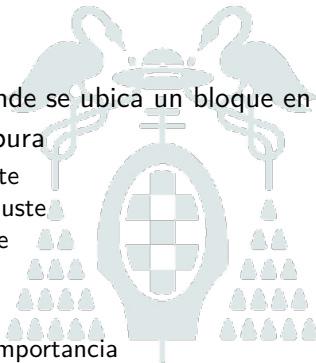
Bloques libres

Inicial	1024					1
P1 pide 70	P1	128	256	Kb	512	3
P2 pide 35	P1	P2	64	256	512	3
P3 pide 80	P1	P2	64	P3	128	3
Devuelve P1	128	P2	64	P3	128	4
P4 pide 60	128	P2	P4	P3	128	3
Devuelve P2	128	64	P4	P3	128	4
Devuelve P4	256		P3	128	512	3
Devuelve P3	1024					1

Kb

Políticas de ubicación

- Determinan dónde se ubica un bloque en MP
- Segmentación pura
 - Primer ajuste
 - Siguiendo ajuste
 - Mejor ajuste
 - Peor ajuste
- Paginación
 - Carece de importancia



Políticas de búsqueda o de lectura

- Determinan cuándo y qué páginas se cargan en MP
- Paginación por demanda
Sólo se carga en MP cuando se ha referenciado
 - Ventajas
 - En MP sólo hay lo que se necesita
 - La sobrecarga es mínima
- Paginación anticipada o prepaginación
Se cargan en MP según una predicción
 - Ventajas
 - Si la predicción es buena, el tiempo de ejecución de los procesos se reduce
 - Útil cuando se accede secuencialmente a dispositivos de almacenamiento

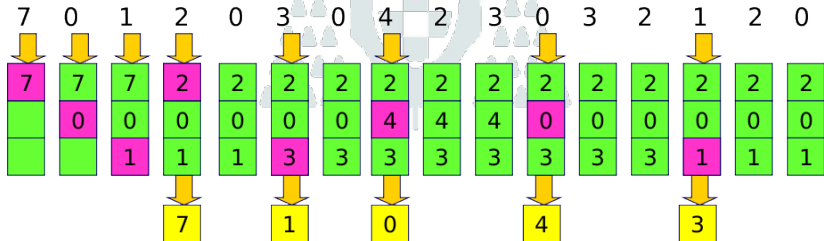
Políticas de reemplazo

- Deciden qué páginas deben sustituirse en MP
- Criterios
 - 1 Baja sobrecarga
 - 2 Sin ajustes en máquinas con distintas configuraciones (*no tuning*)
 - 3 Aproximación al LRU
- Cadena de referencias
 - Lista de referencias a páginas para evaluar la calidad de estos algoritmos
 - Se eliminan los accesos contiguos a la misma página
 - Obtención
 - Artificialmente, de forma pseudoaleatoria
 - A partir de una traza de ejecución

Políticas de reemplazo

Algoritmo óptimo

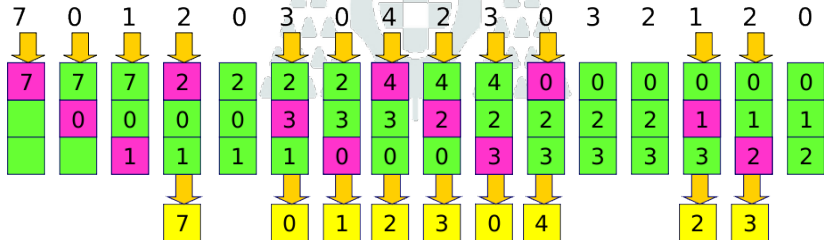
- Se reemplaza la página que va a tardar más tiempo en usarse
- Ventaja: Ofrece la tasa de fallos más baja posible
Establece un criterio de evaluación del resto de algoritmos
- Inconveniente: No implementable



Políticas de reemplazo

Algoritmo FIFO

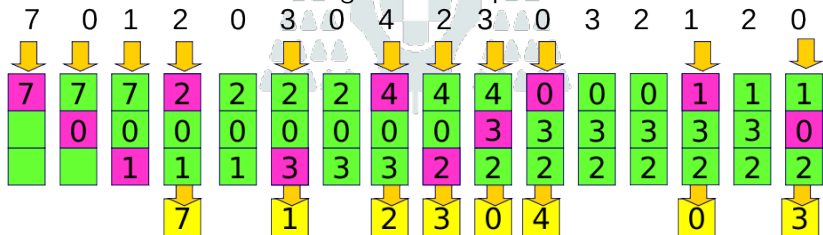
- Se reemplaza la página que lleva más tiempo cargada en MP
 - Ventaja: Sencillo de implementar
 - Inconvenientes
 - Bajo rendimiento
 - Anomalía de Belady $\Rightarrow$ Incremento FPs al aumentar nº marcos
- Ejemplo: Calcular FPs en 1 2 3 4 1 2 5 1 2 3 4 5 con 3 y 4 marcos



Políticas de reemplazo

Algoritmo LRU

- Aproximación al algoritmo óptimo. Utiliza el pasado reciente para predecir el futuro
- Sustituye la página menos usada en el pasado
- Inconvenientes
 - Dificultad de implementación
 - Alta sobrecarga
 - Solución: Utilizar algoritmos de aproximación al LRU



Políticas de reemplazo

Algoritmos de aproximación al LRU - **Reloj global**

- Se crea una lista circular y se emplea un bit R asociado a cada página
 - 1 Mientras la lista no esté llena, cargar páginas con bit de referencia (R) igual a 0
 - 2 Si se referencia una página, se pone bit $R = 1$
 - 3 Cada cierto periodo, con un puntero giratorio poner bits $R = 0$
 - 4 Si la lista está llena:
 - Si bit $R = 0$, reemplazar página y avanzar puntero
 - Si bit $R = 1$, poner bit $R = 0$ y avanzar el puntero
- Ejemplo: 4.3 BSD

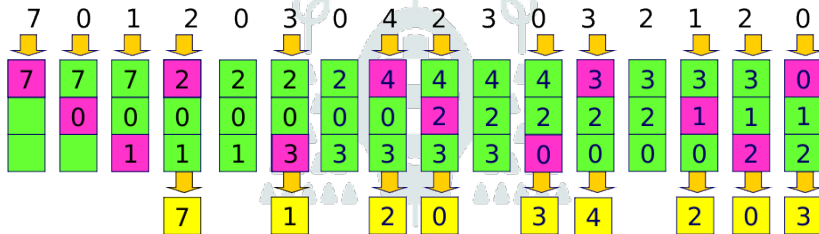
Políticas de reemplazo

Algoritmos de aproximación al LRU - **FIFO con 2ª oportunidad**

- Se emplea un bit R asociado a cada página
 - 1 Elegir una página con criterio FIFO
 - Si $R = 1$, poner $R = 0$
 - Si $R = 0$, sustituir la página
 - 2 Avanzar puntero e ir a 1
- Generalmente se implementa con una cola FIFO circular
- Ventaja: Baja sobrecarga
- Inconveniente: Puede degenerar en un FIFO si todas la páginas tienen $R = 1$

Políticas de reemplazo

Algoritmos de aproximación al LRU - **FIFO** con 2ª oportunidad



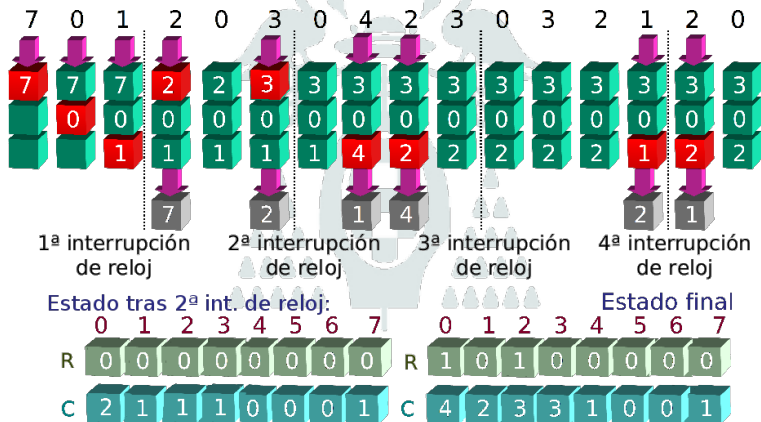
Políticas de reemplazo

Algoritmos de aproximación al LRU - **NFU**

- Se controla la interrupción de un reloj y se emplea un contador y un bit R asociado a cada página
 - Por cada interrupción del reloj
 - Si $R = 1$, incrementar el contador
 - Poner todos los bits R a 0
 - Por cada FP, se reemplaza la página con el menor valor en el contador
- Inconveniente: Si una página se usó mucho, no se reemplazará aunque ya no se acceda a ella
- Solución: Emplear mecanismos que envejezcan los contadores. En vez de incrementarlos
 - Desplazar el contador 1 bit a la derecha
 - Añadir bit de ref. en el extremo izquierdo del contador

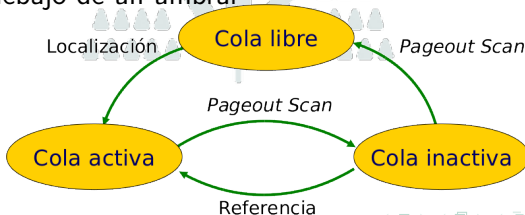
Políticas de reemplazo

Algoritmos de aproximación al LRU - **NFU**



Mach 3.0

- Aproximación al LRU con FIFO segunda oportunidad
- Utiliza tres colas de páginas
 - Cola de páginas libres, puestas a 0. Se reemplazan cada vez que hay un FP
 - Cola de páginas activas. Mantienen el conjunto de trabajo
 - Cola de páginas inactivas. *Buffer* de páginas no referenciadas
- El demonio *pageout* libera páginas, moviéndolas de la cola de activas a la de inactivas cuando el número de páginas libres cae por debajo de un umbral



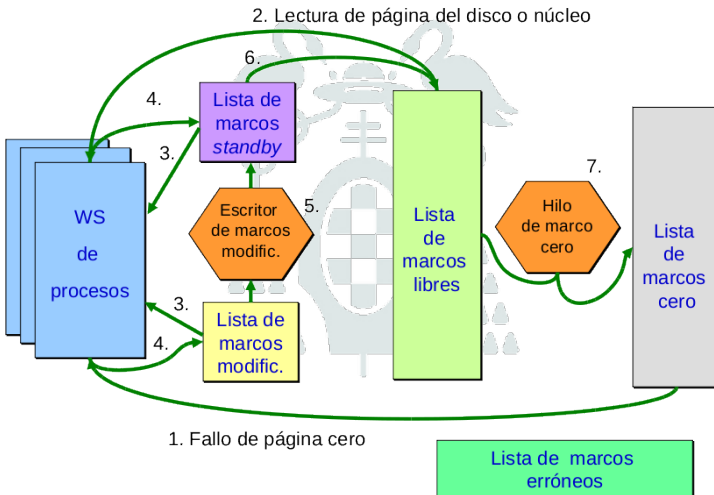
W2K

- Mecanismo de *clustering*
 - Si hay FP, realiza paginación por demanda anticipada o prepaginación, trayendo varias páginas contiguas
 - Depende de la cantidad de MP y tipo de objeto que produjo el FP
 - En sistemas de sobremesa el valor del *clustering* es 8 marcos para código, 4 para datos y 8 para el resto
- Gestión del conjunto de trabajo (WS) $\Rightarrow$ Número de páginas cargadas en MP del proceso en ejecución
 - Asignación variable de número de marcos y alcance local
 - Al iniciar un proceso se asigna un tamaño mínimo del WS
 - El tamaño de los WS varía dinámicamente entre 50 y 345 páginas
 - Reajusta el WS de los procesos y decide qué cantidad de páginas se pueden liberar
 - Incrementa el tamaño del WS si el proceso genera FPs

W2K

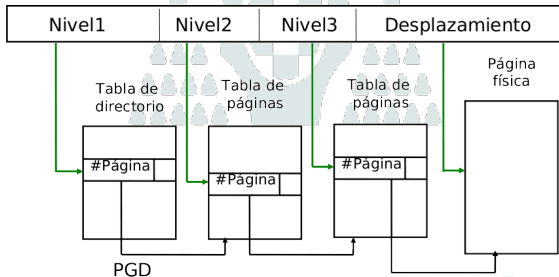
- Estado de los marcos
 - Erróneo - Con daños físicos
 - Libre - Sin asignar, pero no puede asignarse hasta poner a 0s
 - Vacío - Iniciado a 0s
 - Activo - Presente en un WS o bloqueado en memoria
 - Transición - Está actualizándose con información del disco
 - Reposo - Ha dejado de pertenecer a un WS, pero contiene información y está apuntado desde la TMP
 - Modificado - La información que contiene no se ha salvado en disco
 - Modificado sin escritura - Es necesario esperar para actualizar su información en disco

W2K



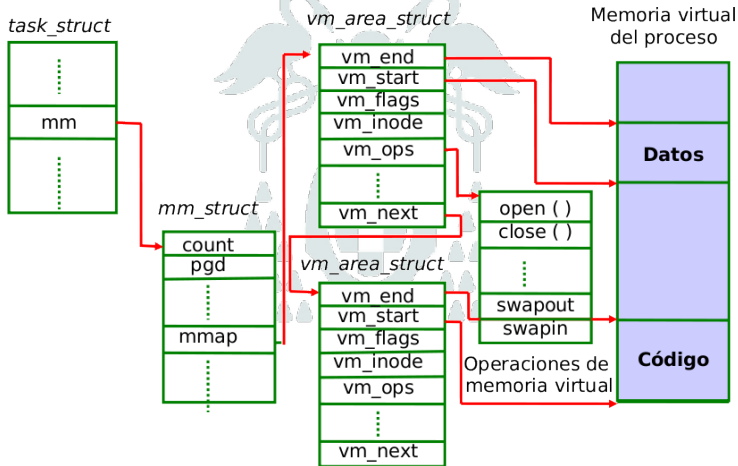
Linux

- Direccionamiento de la memoria virtual
 - Tabla de páginas con tres niveles y cuatro campos: directorio de páginas, directorio intermedio, tabla de páginas y desplazamiento
 - Cada tabla de páginas ocupa 1 página
 - Diseñada para Alpha 64 bits. Se adapta a x86 (32 bits) definiendo tamaño del directorio intermedio = 1



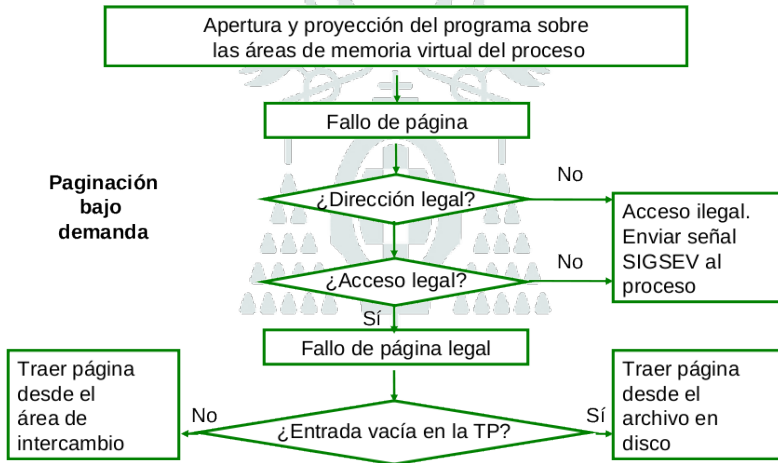
Linux

• Estructuras de datos



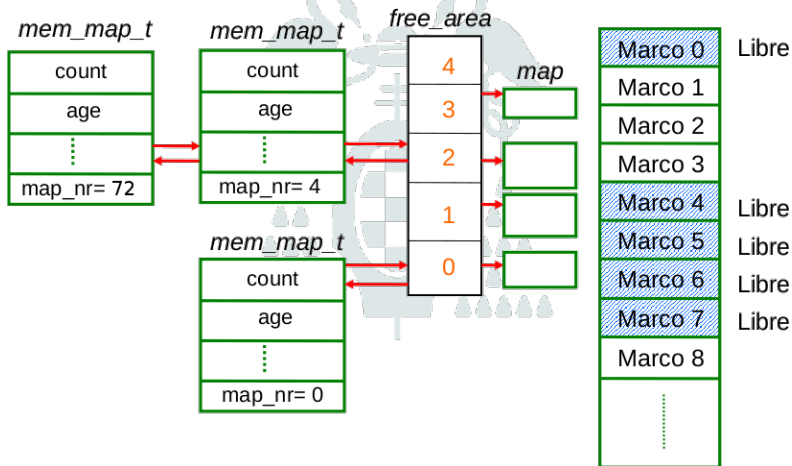
Linux

- Fallo de páginas



Linux

- Asignación de páginas utilizando el sistema de colegas



Linux

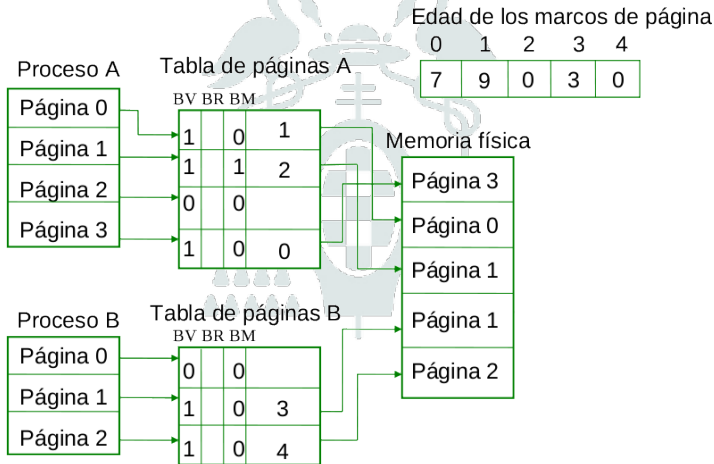
- Algoritmo de reemplazo
 - Alcance del reemplazo global
 - La gestión del área de intercambio se realiza con el demonio kswapd
 - Cada segundo comprueba el número de marcos libres y busca los que pueden ser reemplazados
 - Es un algoritmo de aproximación al LRU con envejecimiento
 - Basado en la edad de las páginas

Linux

- Algoritmo de reemplazo
 - Técnica de envejecimiento de páginas
 - Todas las páginas se inician con edad 3
 - Si se accede a una página, $R = 1$, se incrementa en 3 la edad de la página, hasta un máximo de 20
 - Si se ejecuta kswapd, se decrementa en 1 la edad de la páginas que no se usan (bit $R = 0$)
 - Si una página modificada se lleva a disco
 - Se marca como inválida la entrada en la TMP
 - Se incluye la información para su recuperación posterior
 - Se libera el marco, añadiéndolo a lista de marcos libres
 - Las páginas no modificadas se marcan como libres y sus marcos se añaden a la lista de marcos libres
 - Si se recuperan suficientes marcos para el proceso, el demonio dormirá de nuevo
 - Si no, se continúa con el siguiente proceso

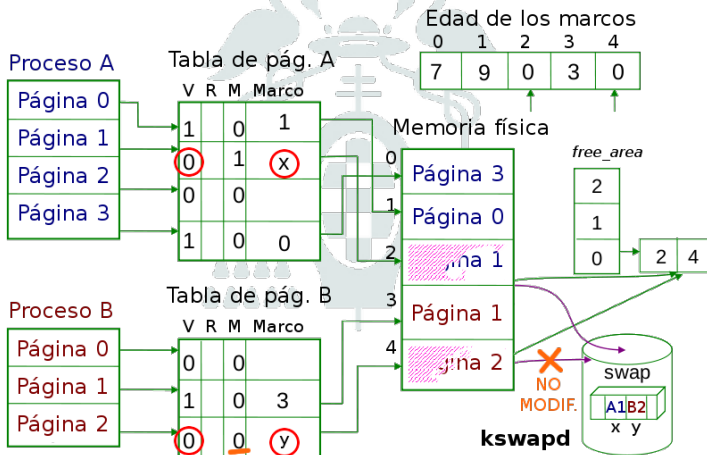
Linux

● Algoritmo de reemplazo



Linux

● Algoritmo de reemplazo



Referencias bibliográficas

-  [Sánchez, 2005] S. Sánchez Prieto.
Sistemas Operativos.
Servicio de Publicaciones de la UA. 2005.
-  [Stallings, 1999] W. Stallings.
Organización y arquitectura de Computadores.
Ed. Prentice Hall, 1999.
-  [Silberschatz, 2006] A. Silberschatz, P. B. Galván y G. Gagne
Fundamentos de Sistemas Operativos.
McGraw Hill. 2006
-  [Tanenbaum, 2009] A. Tanenbaum.
Sistemas Operativos Modernos.
Ed. Pearson Education, 2009.