

Fundamentos de los Sistemas Operativos (FSO)

Departamento de Informática de Sistemas y Computadoras (DISCA)

Universitat Politècnica de València

Bloque Temático 4: Gestión de Memoria

Unidad Temática 11 Memoria Virtual (I)

f SO

DISCA



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

- **Objetivos**

- Describir las **ventajas** de un sistema de Memoria Virtual y sus efectos sobre el **rendimiento** del Sistema
- Entender el concepto de **paginación bajo demanda**
- Estudiar las **técnicas de reemplazo de página**

Bibliografía

- A. Silberschatz, P. B. Galvin. “Sistemas Operativos”. 7^a ed. Capítulo 9

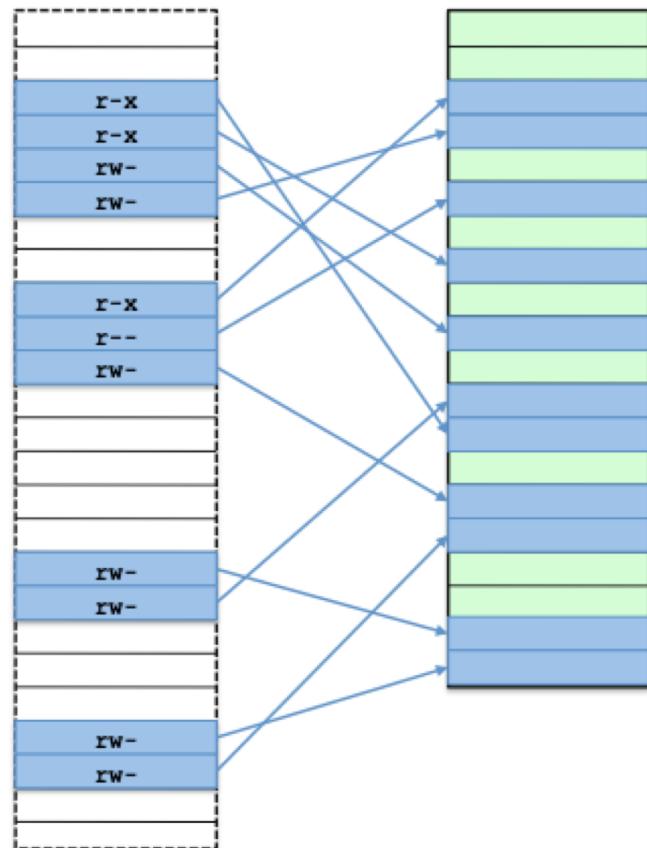
- Contenido
 - **Objetivo de la Memoria Virtual**
 - Concepto de Memoria Virtual
 - Soporte a la Memoria Virtual
 - Paginación por Demanda
 - Memoria Virtual y Gestión de Procesos
 - Serie de Referencias
 - Algoritmo de Reemplazo FIFO
 - Algoritmo de Reemplazo Óptimo
 - Algoritmo de Reemplazo de LRU

Objetivo de la Memoria Virtual

fso

- Gestión de memoria sin MV
 - El SO reserva para el proceso **toda la memoria** prevista en su mapa, aunque no la utilice
 - Todos los accesos a la memoria principal duran el mismo tiempo
 - Cuando el proceso termina, el SO reutiliza todos los marcos liberados

Espacio lógico Memoria física

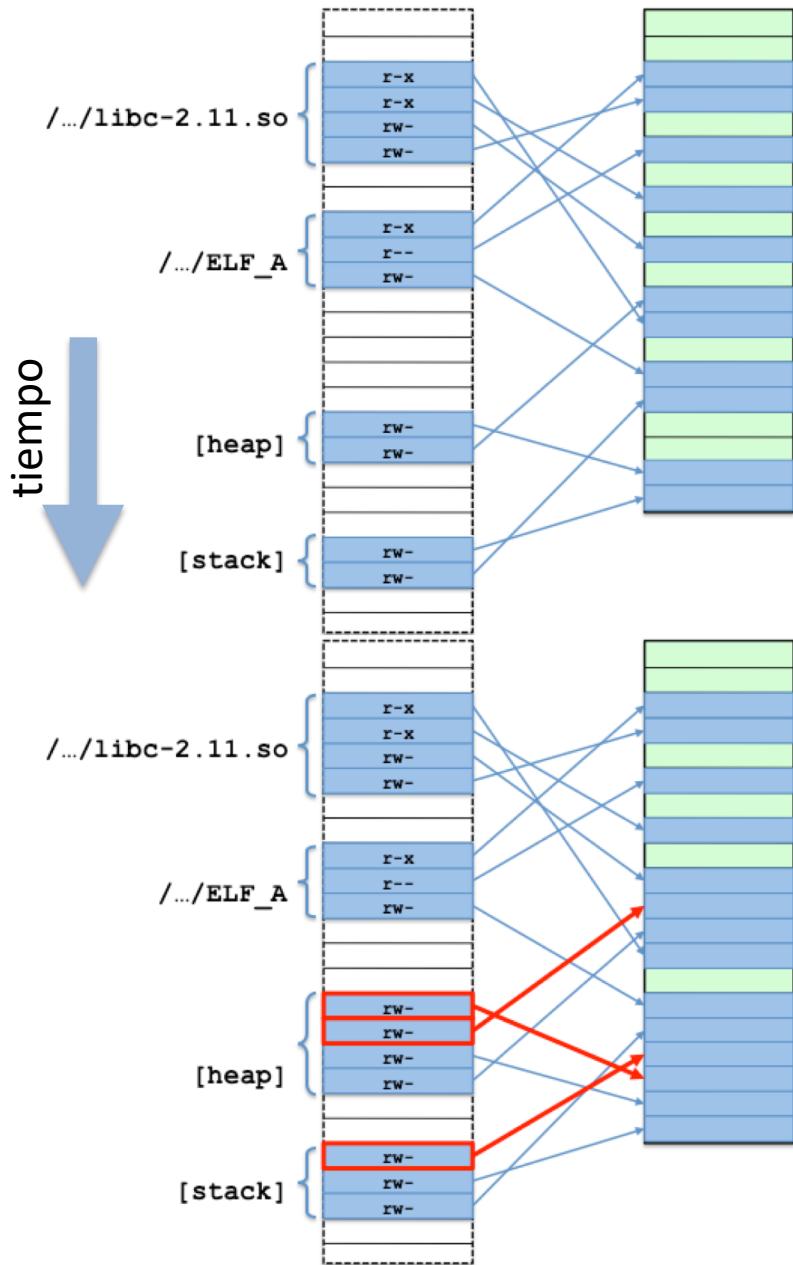


Marco libre,
disponible para
otros procesos

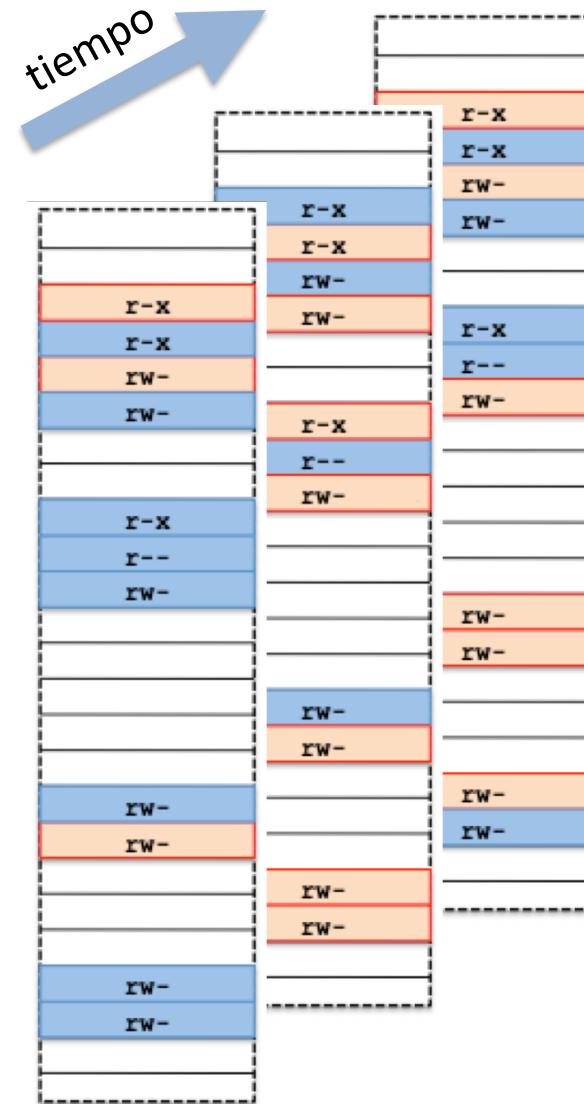
Objetivo de la Memoria Virtual

fso

- El mapa de memoria de un proceso evoluciona
 - Los procesos tienen requerimientos de memoria cambiantes
 - El compilador sólo puede prever las instrucciones y las variables globales del programa
 - El número de regiones crece y también su tamaño
 - La pila crece cuando las funciones se llaman unas a otras
 - Las regiones de *heap* aparecen y crecen por efecto de las llamadas de reserva de memoria dinámica
 - La creación de nuevos hilos exige memoria para sus variables locales

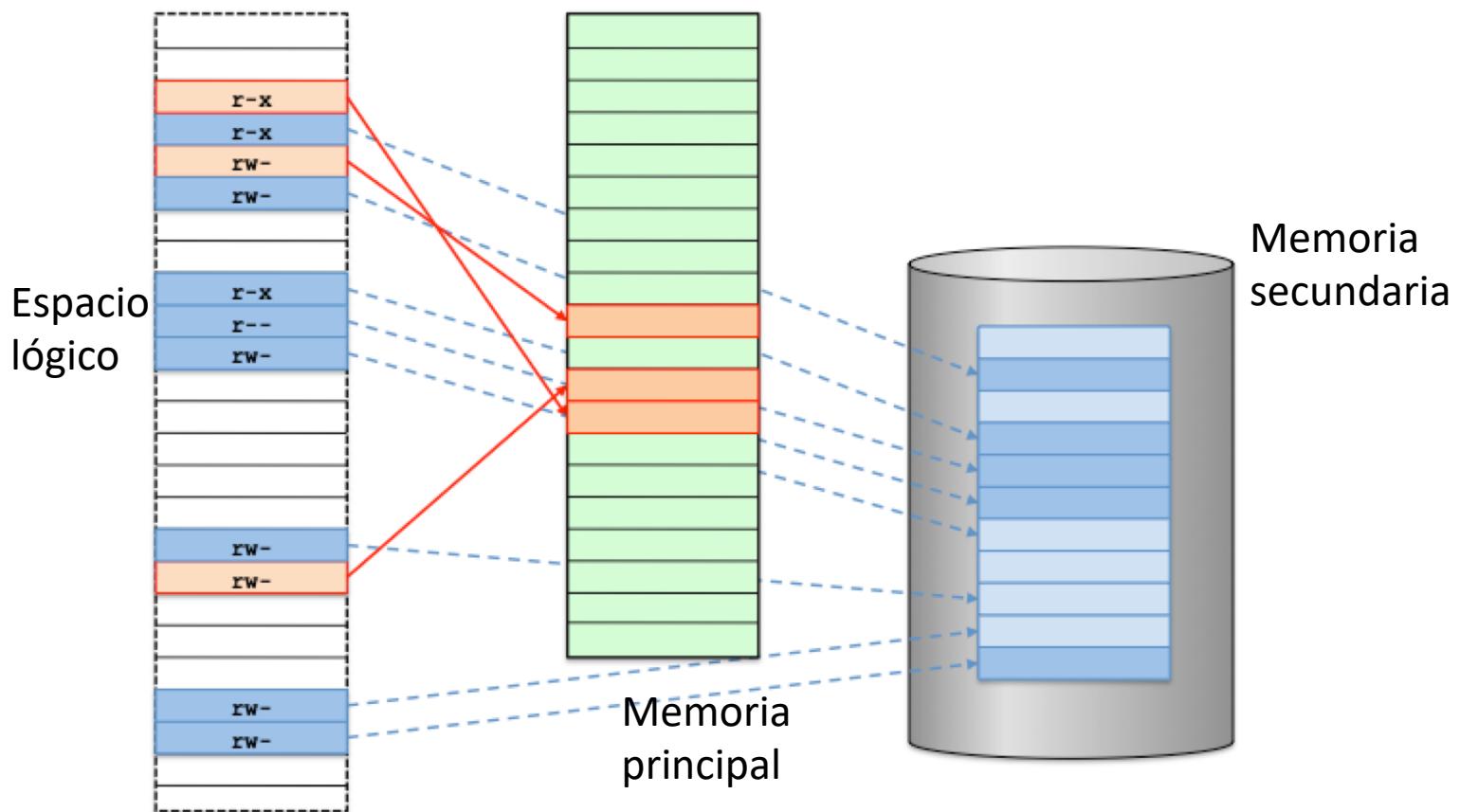


- Principio de localidad de referencia
 - Un proceso puede ocupar mucha memoria lógica, pero los accesos tienden a concentrarse en zonas *calientes*
 - En un intervalo de tiempo dado, se leen las instrucciones de unos pocos métodos/funciones, se leen y se escriben reiteradamente unas pocas variables
 - Conforme el proceso evoluciona, los puntos calientes cambiarán de lugar (otras funciones, otras variables)



- **Gestión de memoria con MV**

- El SO gestiona la asignación de memoria del proceso, para que solo las zonas calientes se encuentren en la memoria física
- El resto del espacio lógico del proceso se encuentra en la memoria secundaria (habitualmente el disco): área de *swap*



- Contenido
 - Objetivo de la Memoria Virtual
 - **Concepto de Memoria Virtual**
 - Soporte a la Memoria Virtual
 - Paginación por Demanda
 - Memoria Virtual y Gestión de Procesos
 - Serie de Referencias
 - Algoritmo de Reemplazo FIFO
 - Algoritmo de Reemplazo Óptimo
 - Algoritmo de Reemplazo de LRU

- **La tecnología base de la MV**

- La MV combina memoria principal con memoria secundaria (habitualmente disco)
- La memoria principal está formada por palabras de 2, 4 u 8 bytes direccionables por el procesador, y su tiempo de acceso se mide en ns
 - Cada ciclo de instrucción supone uno o más accesos a la memoria principal
 - Cada acceso (lectura de una instrucción, lectura o escritura de un dato) trasfiere una palabra entre la memoria y el procesador
- La memoria secundaria está organizada en bloques (512, 4096 bytes, etc) accesibles a través de un adaptador y su tiempo de acceso se mide en ms ($1 \text{ ms} = 10^6 \text{ ns}$)
 - Para una operación con la memoria secundaria, el procesador ha de ejecutar muchas instrucciones
 - La transferencia de una página entre la memoria principal y la secundaria se realiza en una única operación de E/S

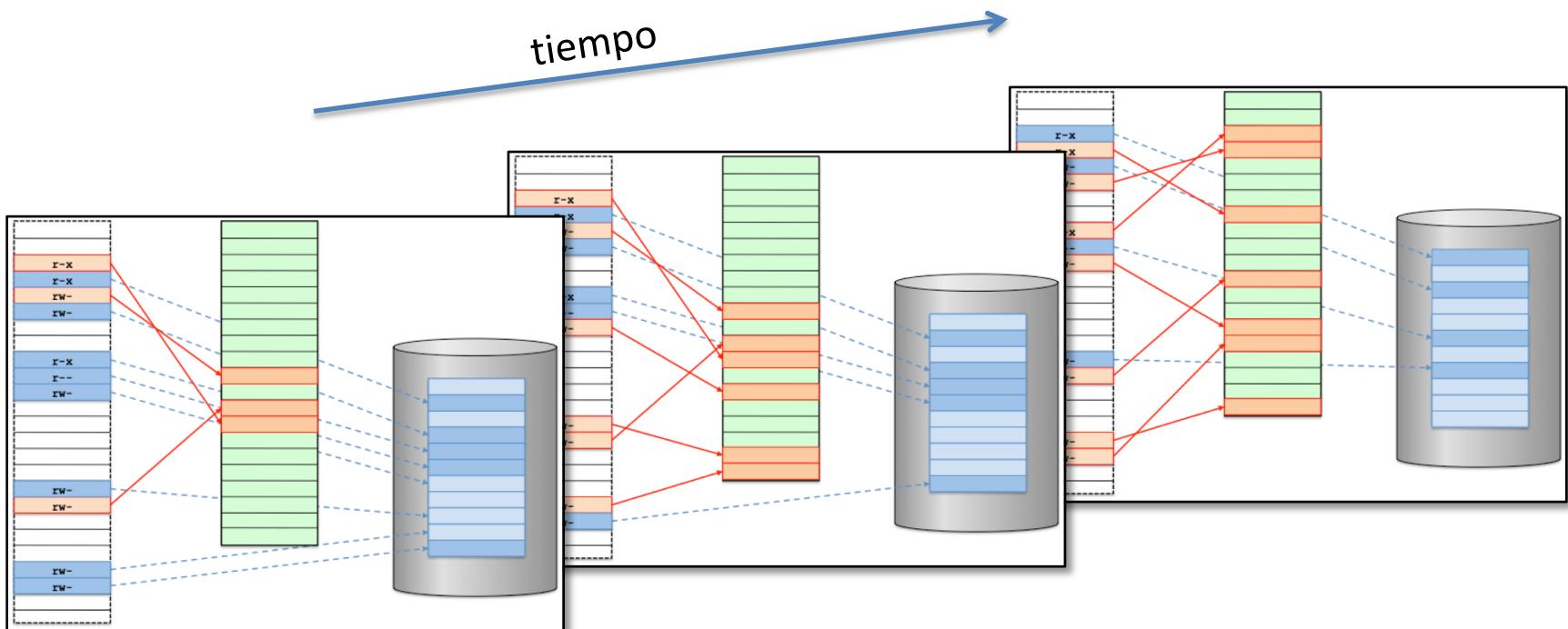
- **Esquema de la MV**

- El SO gestiona una asignación dispersa de memoria (habitualmente paginación)
- El SO gestiona el área de swap.
 - Según el caso, en una partición dedicada en el disco duro o un archivo en el sistema de archivos común
- Cada una de las páginas del espacio lógico de un proceso puede encontrarse en dos estados
 - Válida: la página se encuentra en un marco
 - Inválida: la página está en el área de swap
- Dos situaciones durante cada acceso a la memoria principal:
 - Acierto (situación más frecuente): referencia a página válida.
 - Fallo: referencia a página inválida. Es necesario cambiar la tabla de asignación y, según el caso, transferir una o más páginas entre la memoria física y el disco.

Concepto de Memoria Virtual

fso

- Paginación por demanda = **Paginación + Intercambios** (swapping) entre memoria principal y secundaria
 - Las **páginas** se cargan en memoria cuando se hace referencia a ellas durante la ejecución del proceso
 - Cada página cargada en memoria tiene asignado un marco
 - Cuando un proceso deja de acceder a una página, el SO terminará transfiriéndola a la memoria secundaria para liberar espacio
 - Las páginas que nunca se acceden no llegan a cargarse

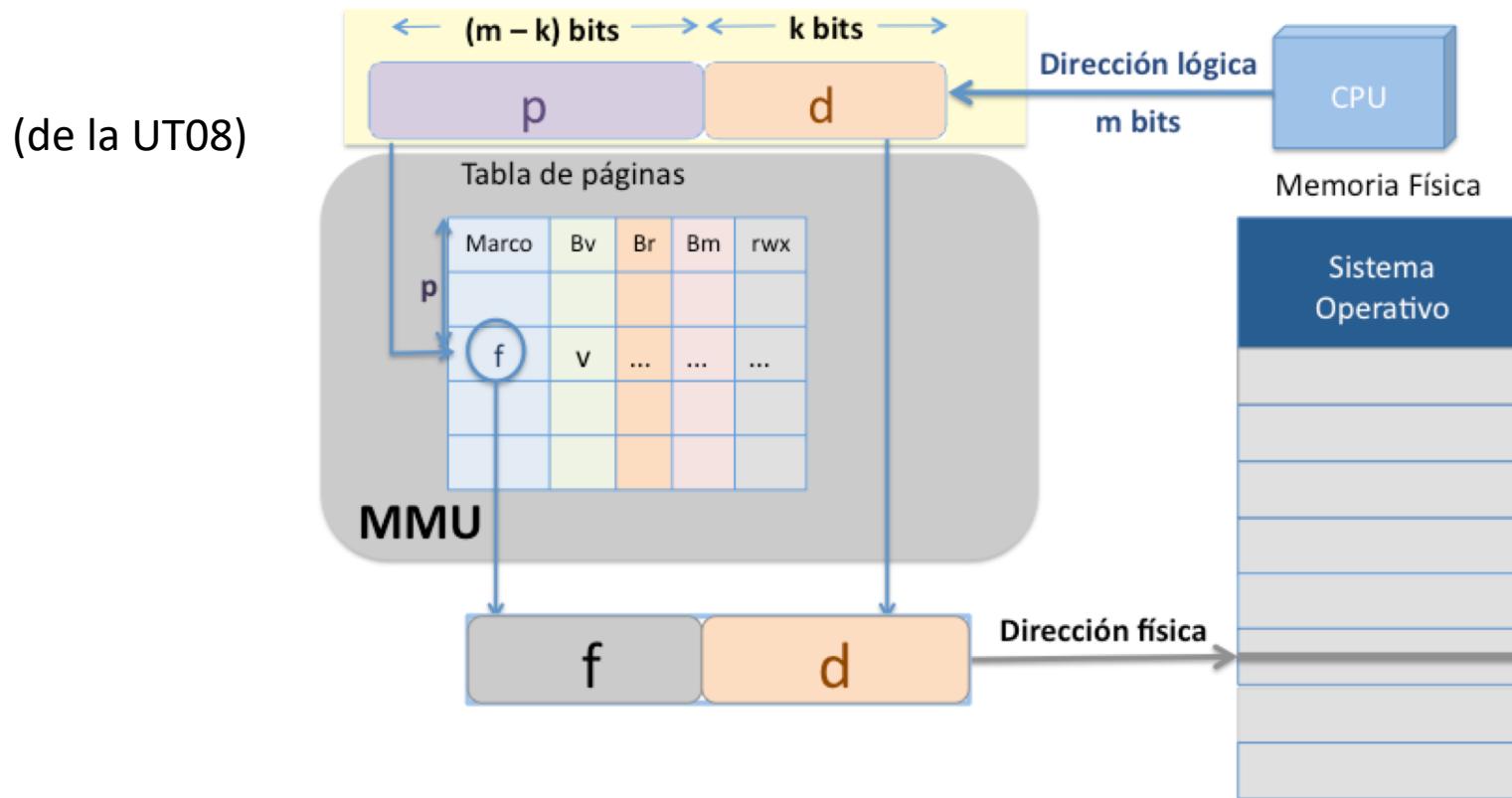


- Beneficios de la paginación (con o sin MV)
 - Permite proteger los accesos a cada página
 - Se detectan errores de programa y accesos maliciosos
 - Permite compartir páginas entre procesos:
 - Por ejemplo, las instrucciones de las bibliotecas comunes
 - Permite compartir variables entre procesos
- Beneficios de la Memoria Virtual
 - Ahorra memoria:
 - Aumenta el grado de multiprogramación
 - Permite procesos de mayor tamaño
 - Facilita la gestión de la memoria dinámica
- Coste de la Memoria Virtual
 - Crece el tiempo de retorno de los procesos porque el tratamiento de algunos fallos de página los suspende
 - El área de swap ocupa espacio en el disco
 - Aumenta la carga de trabajo de la memoria secundaria
 - Mayor complejidad en el diseño del sistema operativo
 - El PCB ha de proteger la tabla de asignación de cada proceso

- Contenido
 - Objetivo de la Memoria Virtual
 - Concepto de Memoria Virtual
 - **Soporte a la Memoria Virtual**
 - Paginación por Demanda
 - Memoria Virtual y Gestión de Procesos
 - Serie de Referencias
 - Algoritmo de Reemplazo FIFO
 - Algoritmo de Reemplazo Óptimo
 - Algoritmo de Reemplazo de LRU

Soporte a la MV

- La organización de la tabla de páginas no cambia con la MV
 - Cada entrada de la tabla consiste en un descriptor de página
 - La MMU utiliza el número de página para indexar la tabla

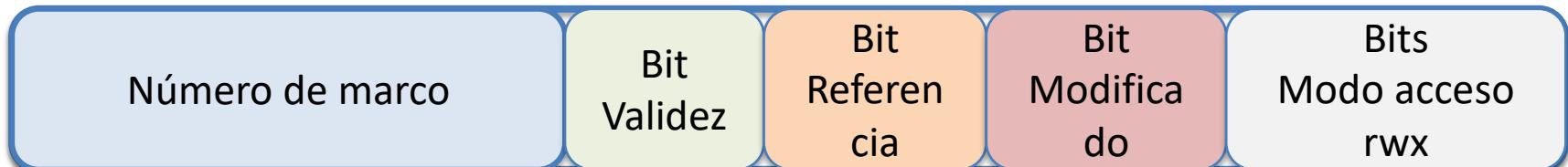


- Los descriptores han de tener más información:

Soporte a la MV

- Descriptor de página para la MV:
 - Cada descriptor de página tiene
 - **Número de marco** donde está ubicada la página en memoria (si es válida)
 - **Bit de Validez**: Indica si una página está mapeada en memoria. Soporta la paginación por demanda.
 - **Bit de Referencia**: Indica si la página ha sido accedida. Necesario para algoritmo de segunda oportunidad.
 - **Bit de Modificación**: Indica si la página ha sido accedida para escritura.
 - **Bits de Modo de Acceso**: sólo lectura, lectura-escritura, ejecución, ..

Descriptor de página

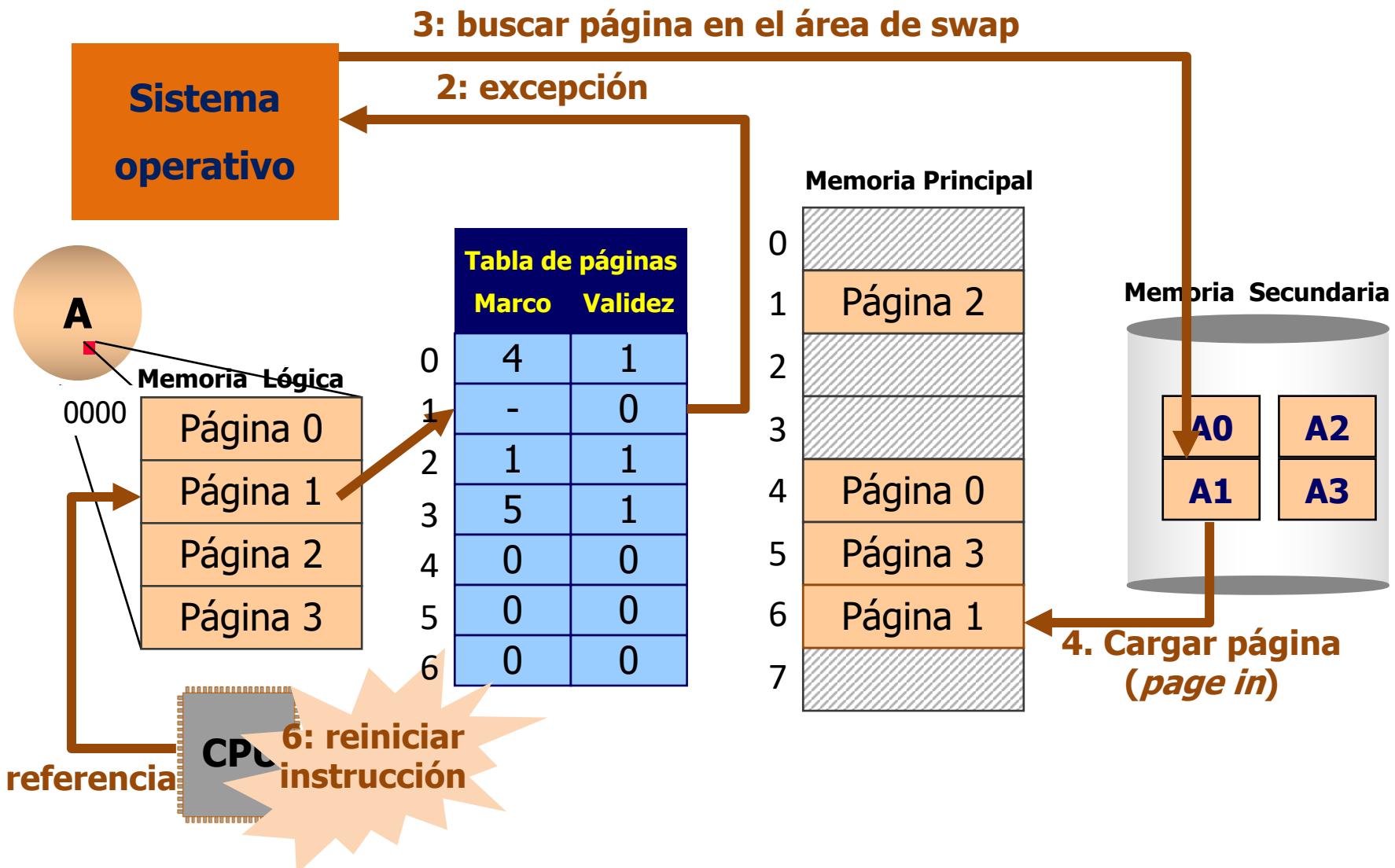


- **Las excepciones de la MMU**
 - Acceso sin incidentes (el más habitual)
 - Acceso correcto a página válida: el descriptor suministra la dirección física
 - El proceso en ejecución no cambia de estado
 - La MMU provocará excepción de **Fallo de página** (y el SO intervendrá) en, al menos, dos casos:
 - **Acceso a página inválida.** Es un acceso legal. El SO trata la excepción como señal de demanda
 - Las transferencias entre disco y memoria para resolver fallos de página suspenden el proceso
 - **Violación de acceso o Acceso ilegal (*segmentation fault*):** acceso a una página fuera del mapa o acceso no permitido a una página
 - Normalmente, el SO termina el proceso

- Contenido
 - Objetivo de la Memoria Virtual
 - Concepto de Memoria Virtual
 - Soporte a la Memoria Virtual
 - **Paginación por Demanda**
 - Memoria Virtual y Gestión de Procesos
 - Serie de Referencias
 - Algoritmo de Reemplazo FIFO
 - Algoritmo de Reemplazo Óptimo
 - Algoritmo de Reemplazo de LRU

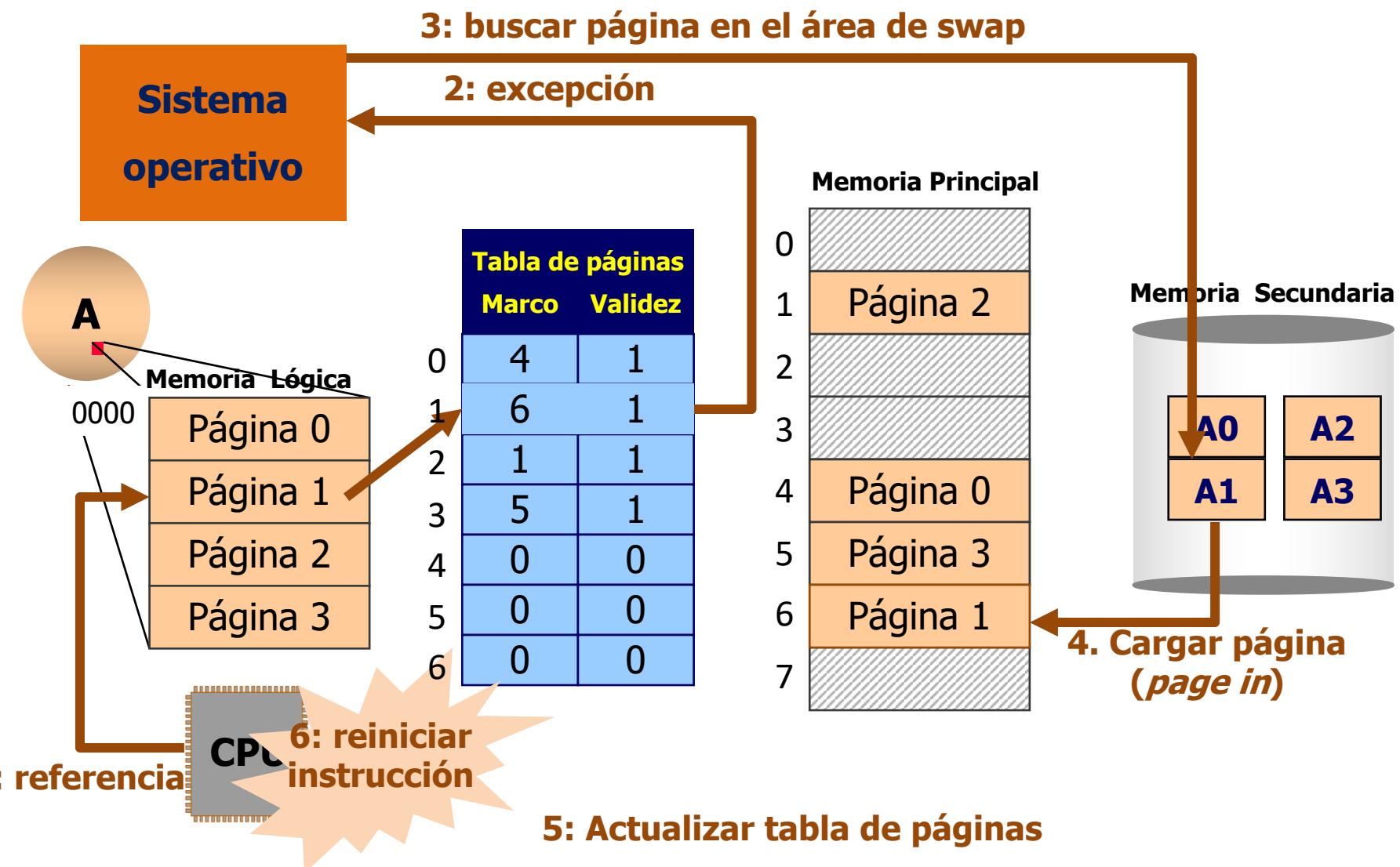
Paginación por demanda

- Fallo de página: caso de Página en Disco



Paginación por demanda

- Fallo de página: caso de Página en Disco

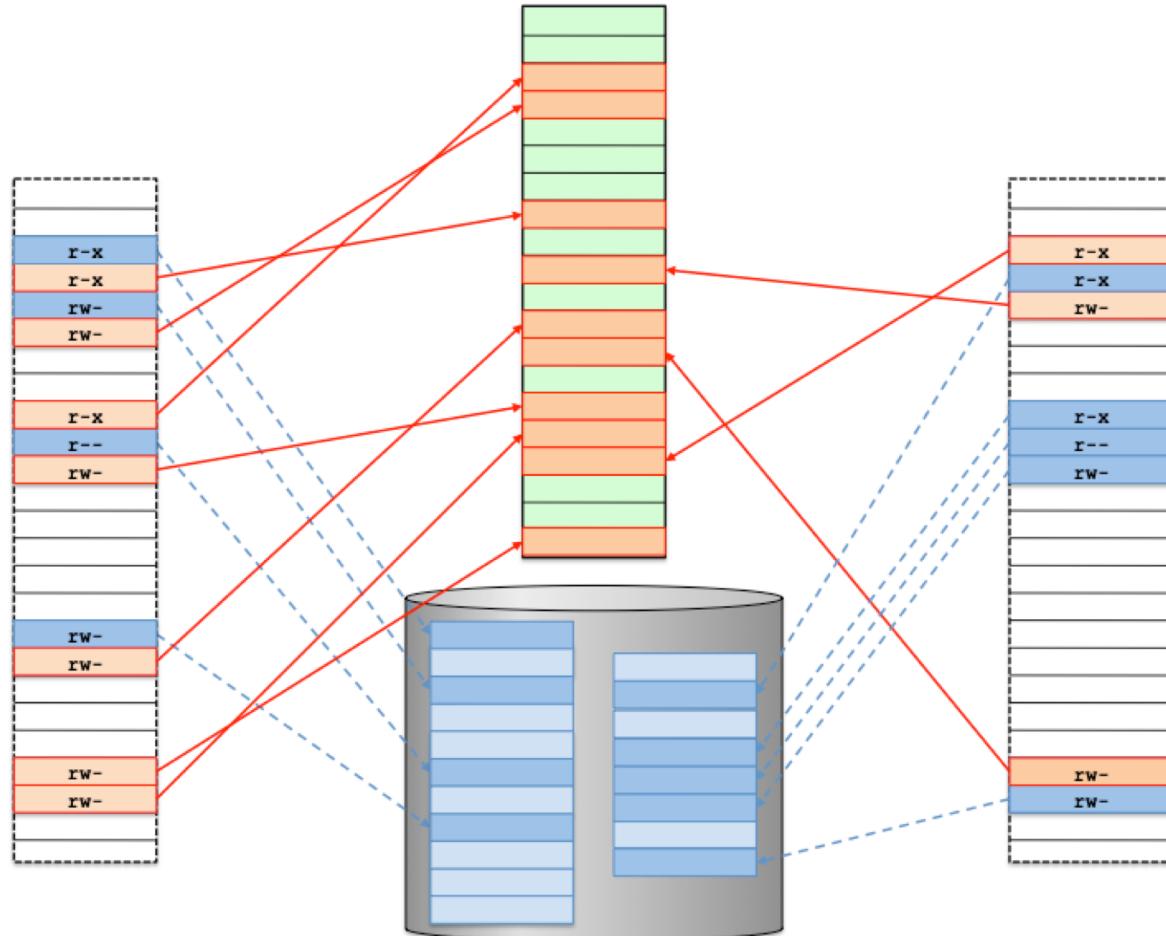


- Algoritmo de fallo de página: caso de página en disco
 - Encontrar la **página demandada** en disco
 - Encontrar un **marco libre**:
 - Si existe un marco libre, utilizarlo.
 - Si no existe marco libre,
 - Ejecutar algoritmo de reemplazo de páginas y seleccionar víctima
 - Si la víctima tiene el bit de modificación a 1, escribir la víctima en disco (page out).
 - En la víctima: Actualizar tabla de páginas, con bit de validez = 0
 - Actualizar la tabla de marcos.
 - **Leer la página demandada en disco** (page in) y ubicarla en el marco libre
 - **Actualizar tabla de páginas** del proceso que genera el fallo
 - Actualizar la tabla de marcos libres
 - El proceso de usuario pasa a preparado
 - cuando obtenga de nuevo la CPU, continuará su ejecución en la instrucción que provocó el fallo de página.

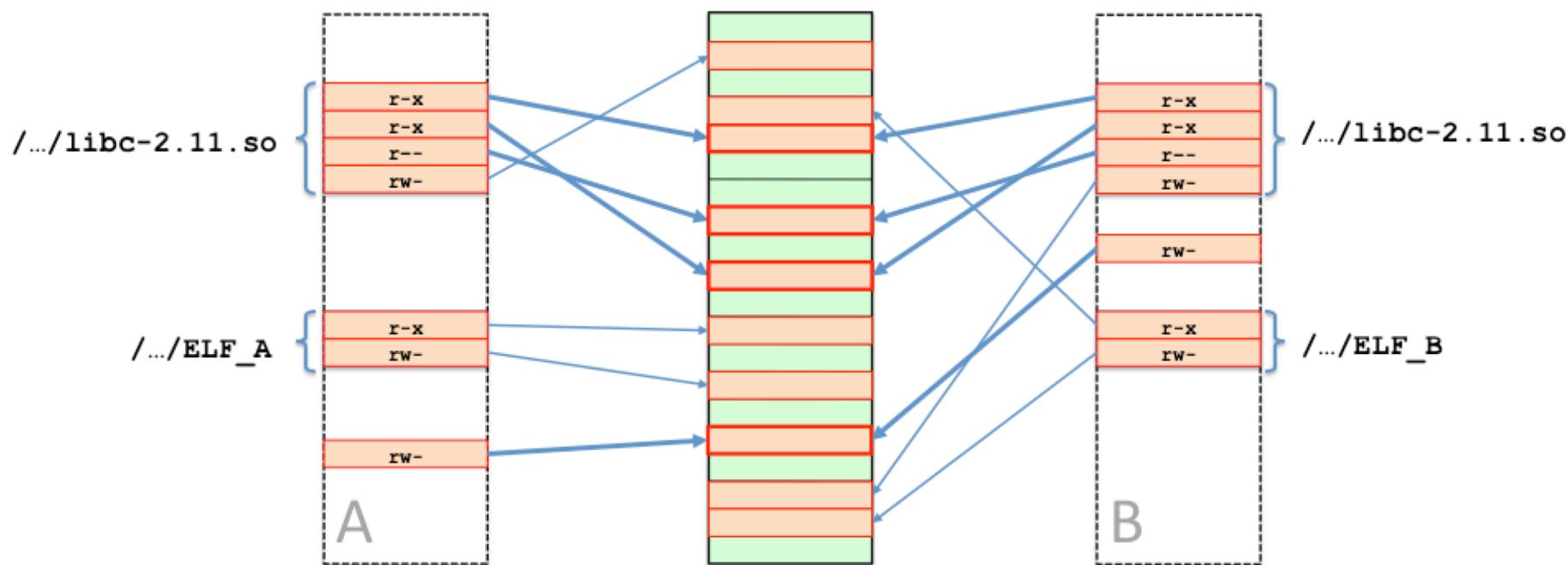
- **Reemplazo de Página**
 - son necesarios cuando la memoria principal está completamente ocupada y se produce un fallo de página:
 - Una página ubicada en memoria principal, denominada **víctima, debe dejar su marco** a la página demandada
 - Si página víctima tiene el bit de modificación a 1 hay que **salvar la víctima a disco (page out)**
 - Se ubica la **página demandada sobre el marco** de la víctima **(page in)**
 - Existen diversos algoritmos para seleccionar la víctima

- Contenido
 - Objetivo de la Memoria Virtual
 - Concepto de Memoria Virtual
 - Soporte a la Memoria Virtual
 - Paginación por Demanda
 - **Memoria Virtual y Gestión de Procesos**
 - Serie de Referencias
 - Algoritmo de Reemplazo FIFO
 - Algoritmo de Reemplazo Óptimo
 - Algoritmo de Reemplazo de LRU

- La MV favorece la concurrencia
 - Al mantener en la memoria principal sólo las zonas calientes de los procesos, aprovechará mejor su capacidad
 - Ejemplo: el proceso A ocupa 11 páginas y B ocupa 8. Y sólo ocupan 10 marcos!



- Compartición de marcos
 - El SO puede ubicar páginas de distintos procesos en el mismo marco
 - Útil para código ejecutable de bibliotecas comunes y para que los procesos puedan compartir variables y archivos proyectados
- Ejemplo: dos procesos A y B (creados a partir de los ejecutables ELF_A y ELF_B) comparten cuatro marcos
 - Tres páginas de la biblioteca libc-2.11.so
 - Una página sin respaldo con variables compartidas



- UNIX: las llamadas fork y exec
 - Una llamada a *exec* invalida todas las páginas de la tabla del proceso
 - Los fallos de página en los accesos posteriores actualizarán los descriptores con el mapa de memoria del nuevo ejecutable
 - Una llamada a *fork* clona la tabla del padre en el proceso recién creado
 - Los procesos padre e hijo disponen de las páginas de sólo lectura (código ejecutable y constantes) en marcos compartidos
 - El SO ubicará las páginas con permiso de escritura (variables, pila, heap) en marcos distintos. La asignación es progresiva (*copy-on-write*): cuando un proceso hace referencia por primera vez a una de ellas, el SO busca un marco nuevo

- Contenido
 - Objetivo de la Memoria Virtual
 - Concepto de Memoria Virtual
 - Soporte a la Memoria Virtual
 - Paginación por Demanda
 - Memoria Virtual y Gestión de Procesos
 - **Serie de Referencias**
 - Algoritmo de Reemplazo FIFO
 - Algoritmo de Reemplazo Óptimo
 - Algoritmo de Reemplazo de LRU

- **Serie de Referencia:**
 - **secuencia de páginas** que han sido **accedidas** durante cierto periodo de tiempo
 - De cada dirección lógica emitida por el procesador se obtiene su número de página
 - Se eliminan las repeticiones: Múltiples referencias consecutivas a una misma página se representa por una sola referencia
- **Ejemplo:**
 - Sistema con páginas de 1000 palabras:
 - Direcciones lógicas del proceso P referenciadas
2500, 5100, 5234, 1800, 1432, 4388, 2124, 8216, 8498
 - Páginas referenciadas
2, 5, 5, 1, 1, 4 , 2 , 8, 8
 - La serie de referencias de dicha secuencia es:
2, 5, 1, 4, 2, 8

- Contenido
 - Objetivo de la Memoria Virtual
 - Concepto de Memoria Virtual
 - Soporte a la Memoria Virtual
 - Paginación por Demanda
 - Memoria Virtual y Gestión de Procesos
 - Serie de Referencias
 - **Algoritmo de Reemplazo FIFO**
 - Algoritmo de Reemplazo Óptimo
 - Algoritmo de Reemplazo de LRU

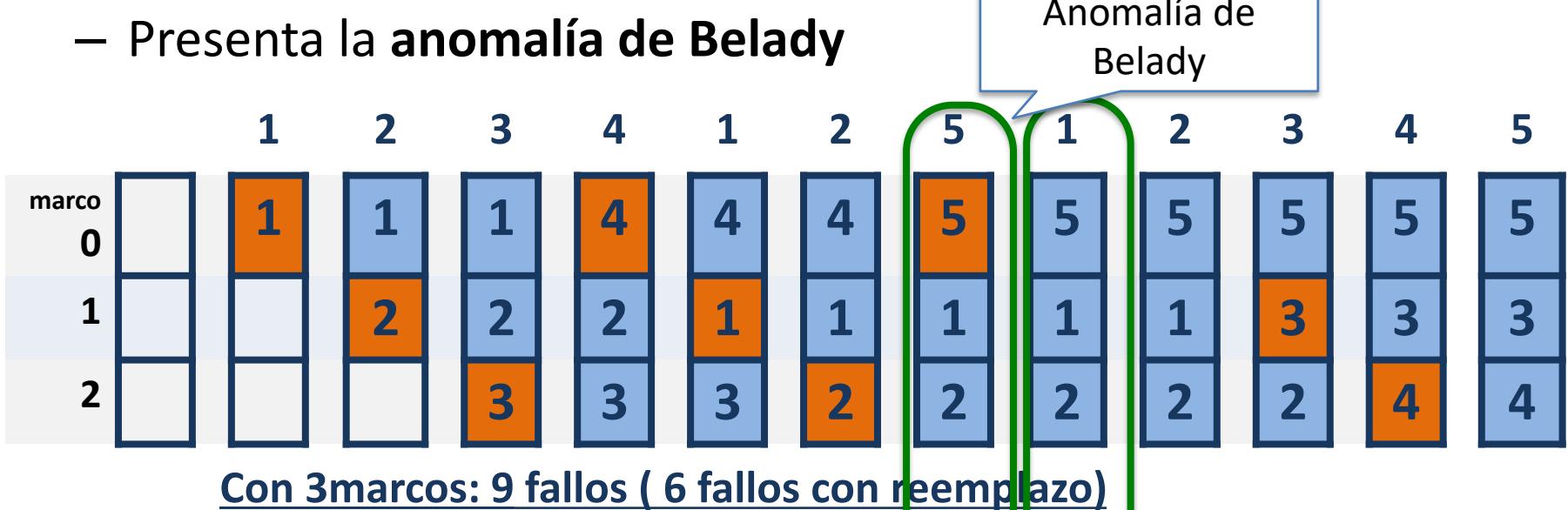
Algoritmo FIFO

- Página **víctima**: la que lleva más tiempo cargada en memoria.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que lleva más tiempo cargada en memoria.
- Presenta la **anomalía de Belady**

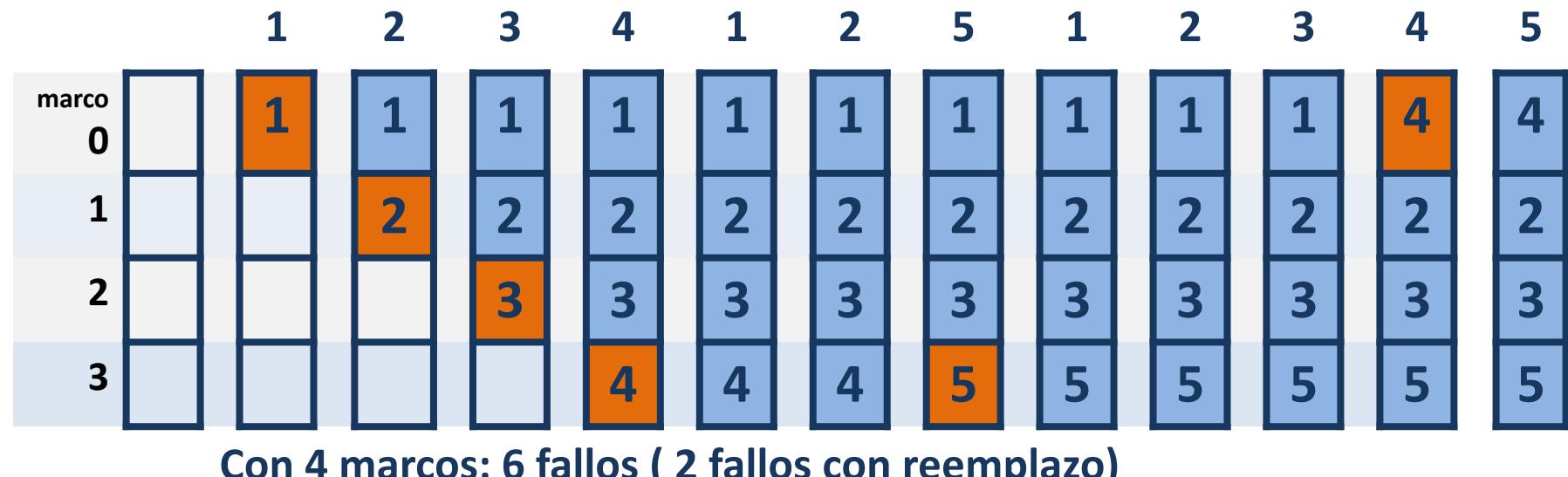


- **Algoritmos de Pila**
 - Garantizan que un conjunto de **páginas** mantenido **con N marcos** es un **subconjunto** del que se mantiene **con N+1 marcos**
- Los algoritmos de **Pila NO** presentan la anomalía de **Belady**

- Contenido
 - Objetivo de la Memoria Virtual
 - Concepto de Memoria Virtual
 - Soporte a la Memoria Virtual
 - Paginación por Demanda
 - Memoria Virtual y Gestión de Procesos
 - Serie de Referencias
 - Algoritmo de Reemplazo FIFO
 - **Algoritmo de Reemplazo Óptimo**
 - Algoritmo de Reemplazo de LRU

Algoritmo Óptimo

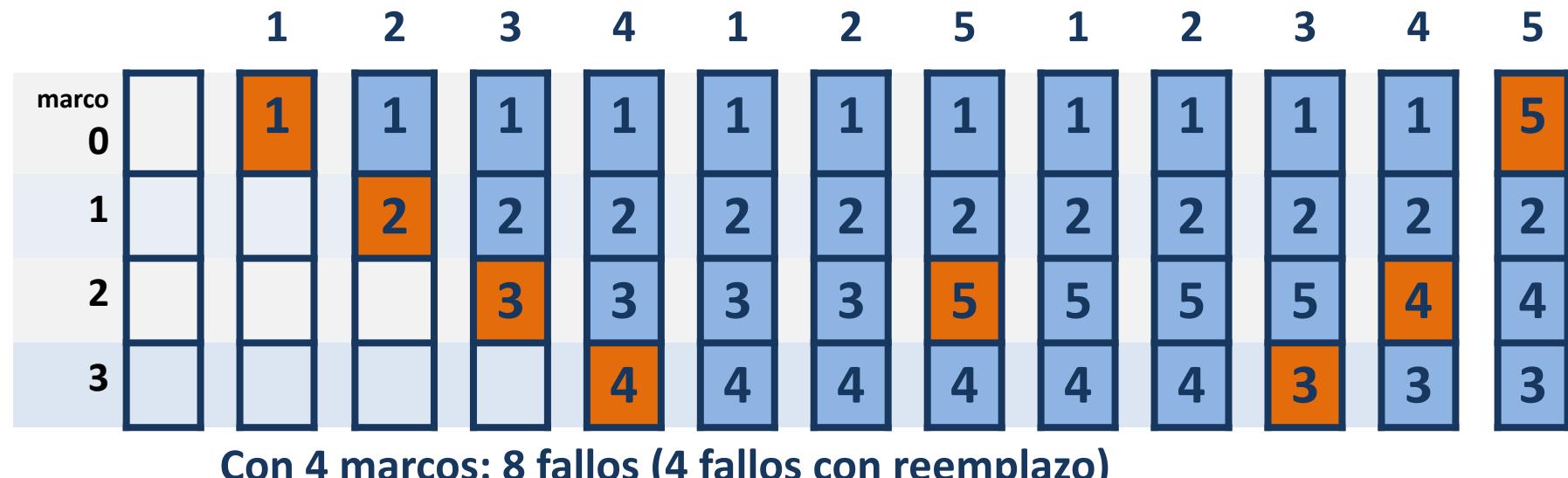
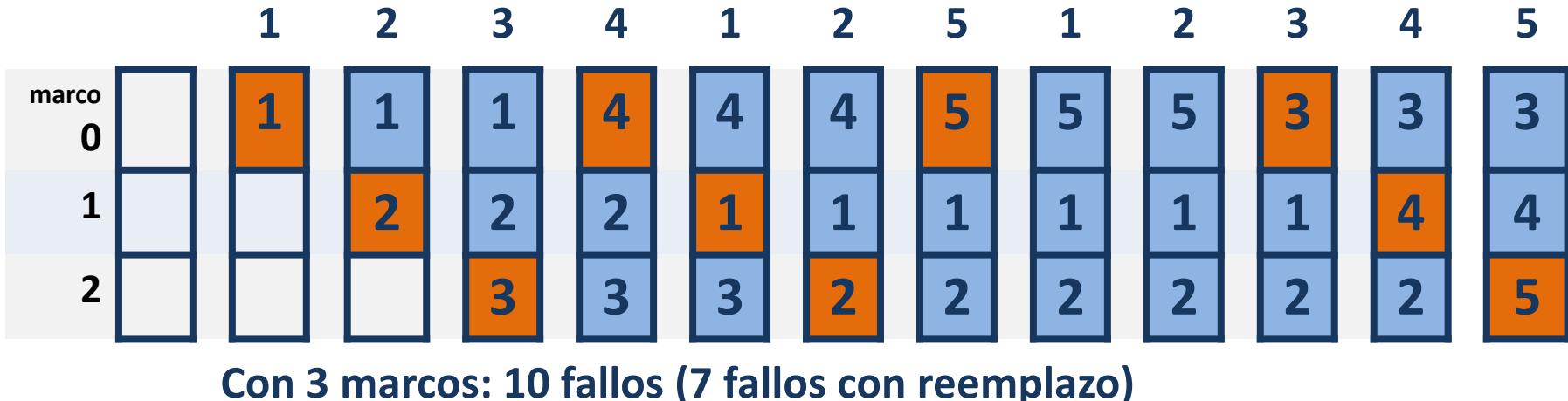
- Página **víctima**: la que **más tiempo tardará** en ser **referenciada**
- Número de fallos mínimo. Implementación imposible



- Contenido
 - Objetivo de la Memoria Virtual
 - Concepto de Memoria Virtual
 - Soporte a la Memoria Virtual
 - Paginación por Demanda
 - Memoria Virtual y Gestión de Procesos
 - Serie de Referencias
 - Algoritmo de Reemplazo FIFO
 - Algoritmo de Reemplazo Óptimo
 - Algoritmo de Reemplazo de LRU**

Algoritmo LRU

- Página víctima: la que hace más tiempo que ha sido referenciada
- Es un algoritmo de Pila

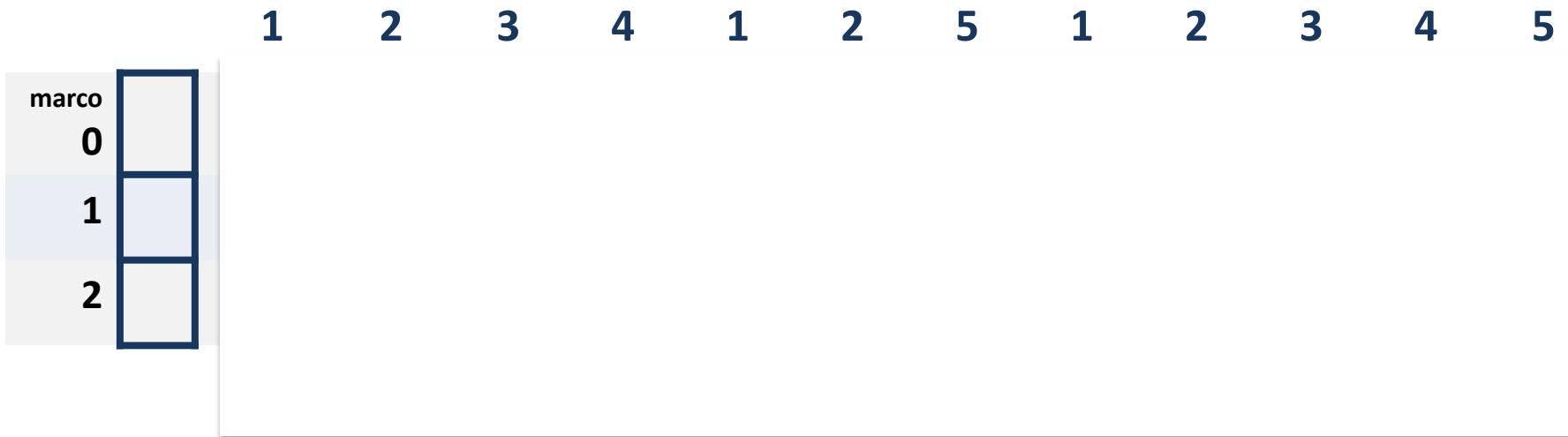


- Implementaciones del LRU
 - Mediante contadores:
 - Cada página tiene un contador asociado
 - Mediante lista ordenada de páginas referenciadas
 - Cuando se referencia una página se pone al final
 - La página víctima es la primera de la lista
- Análisis de eficiencia
 - Ventajas:
 - Buena aproximación al óptimo
 - Inconvenientes:
 - Difícil de Implementar
 - Solución:
 - Algoritmos de aproximación al LRU

- Secuencia del FIFO

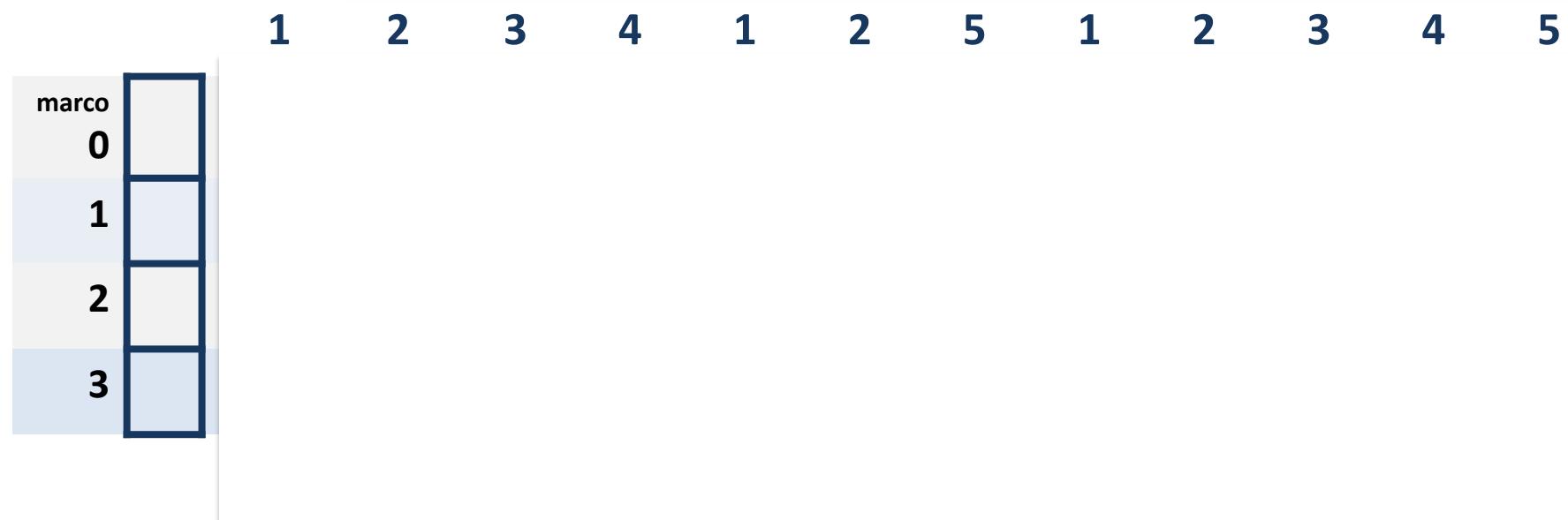
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



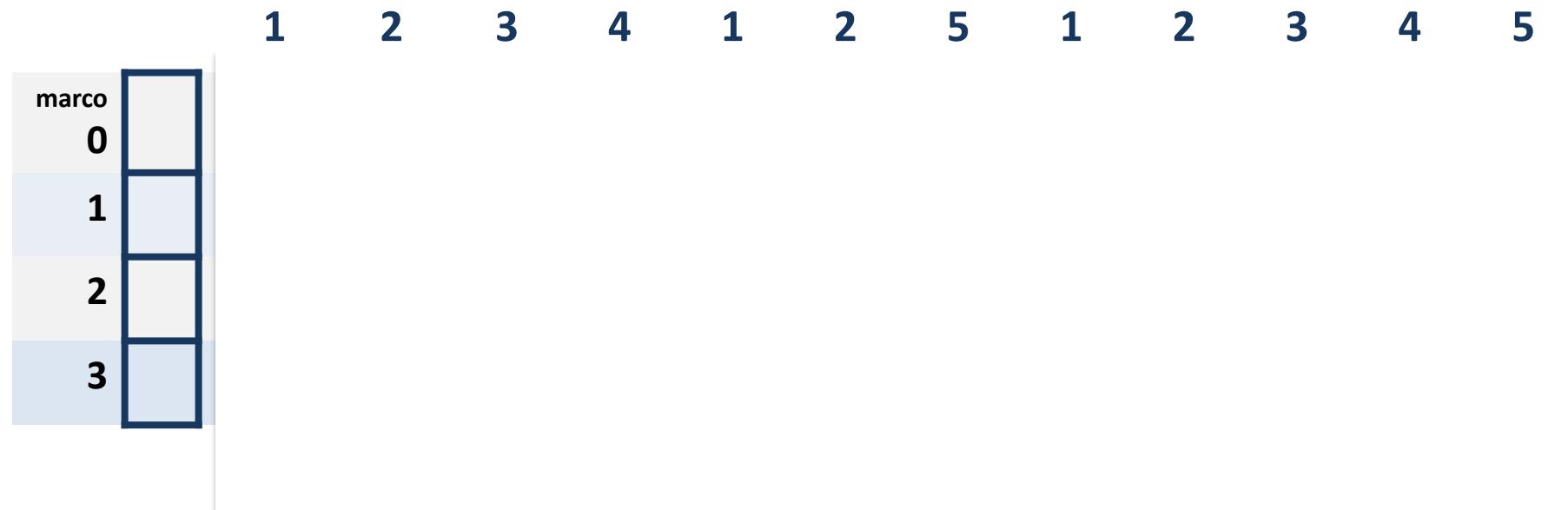
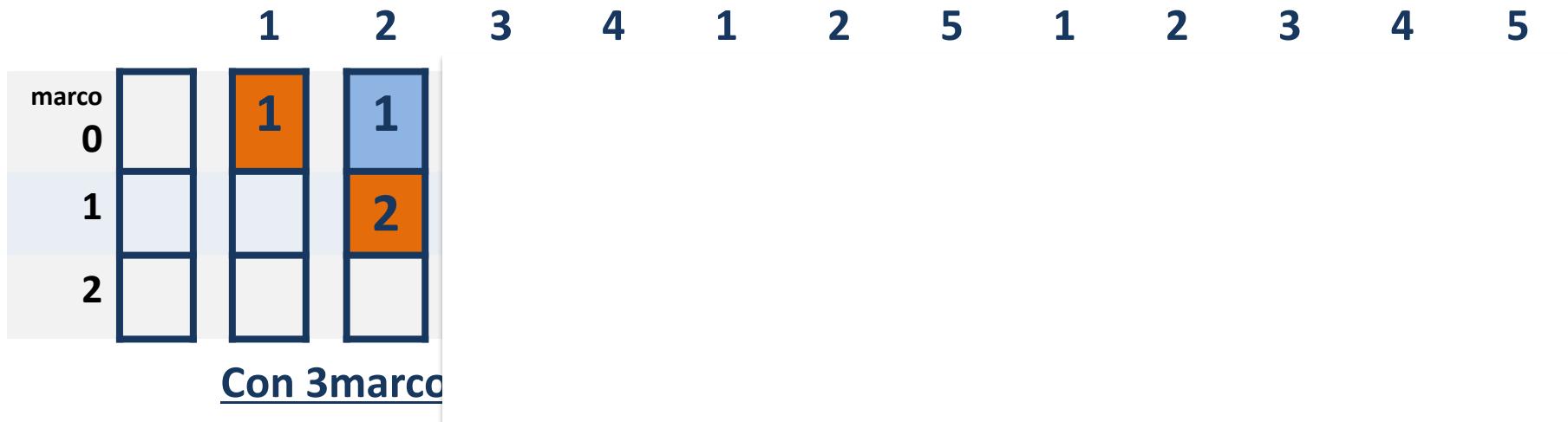
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



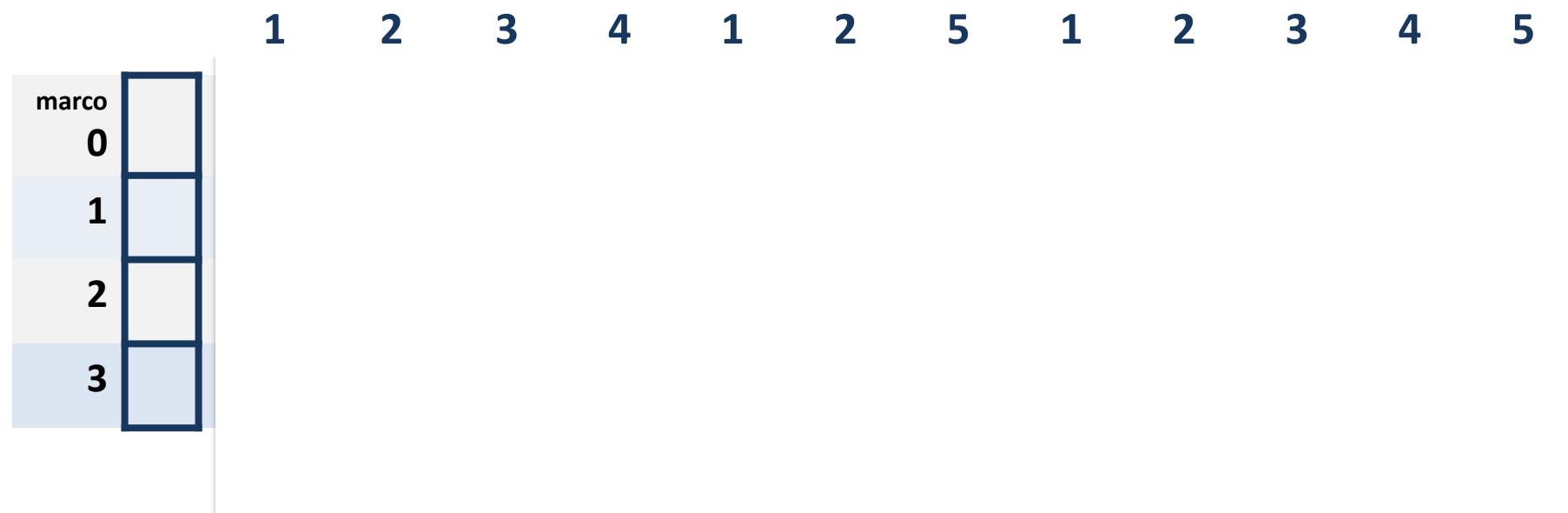
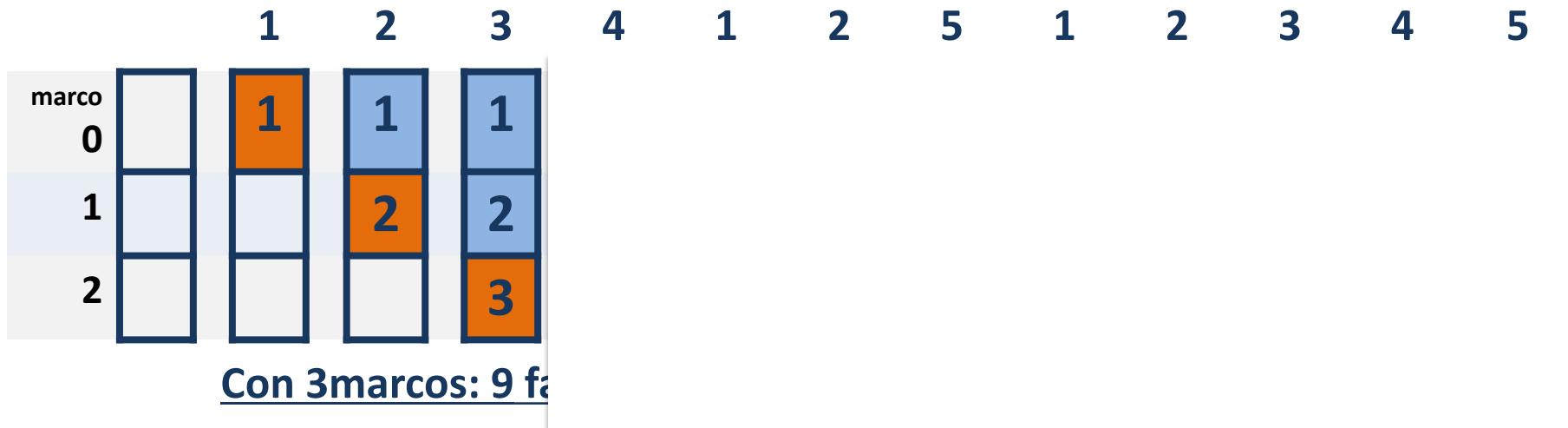
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



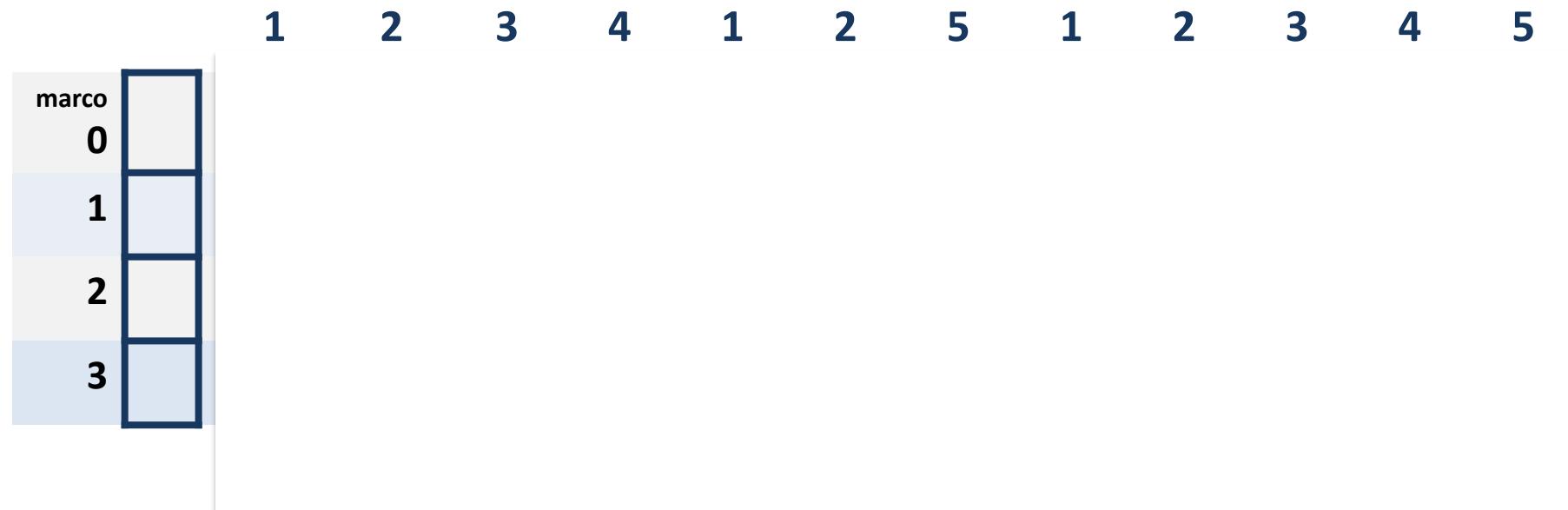
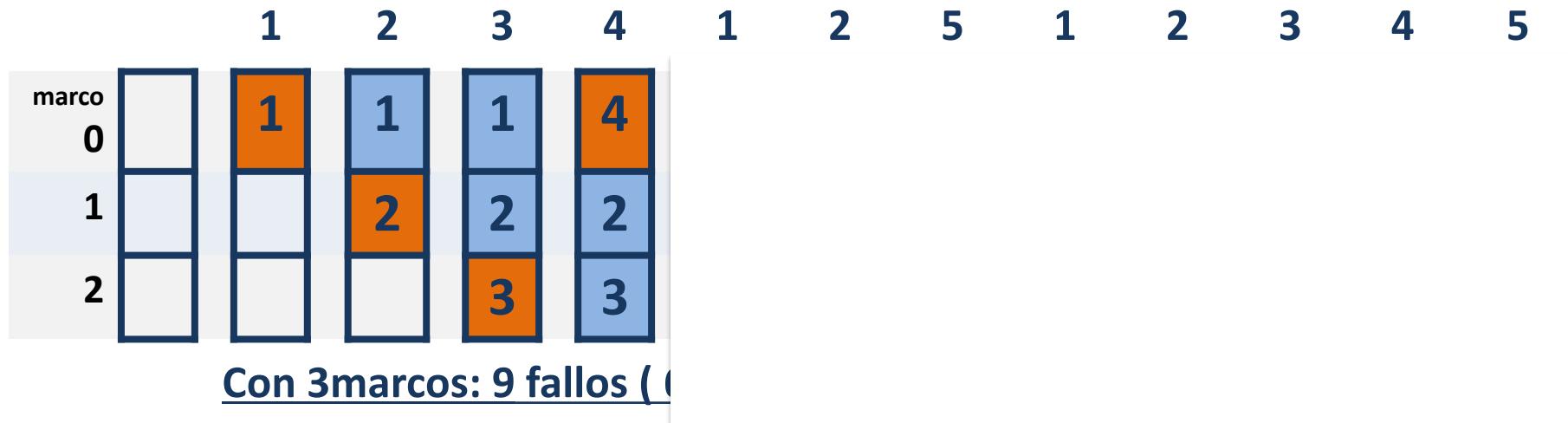
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



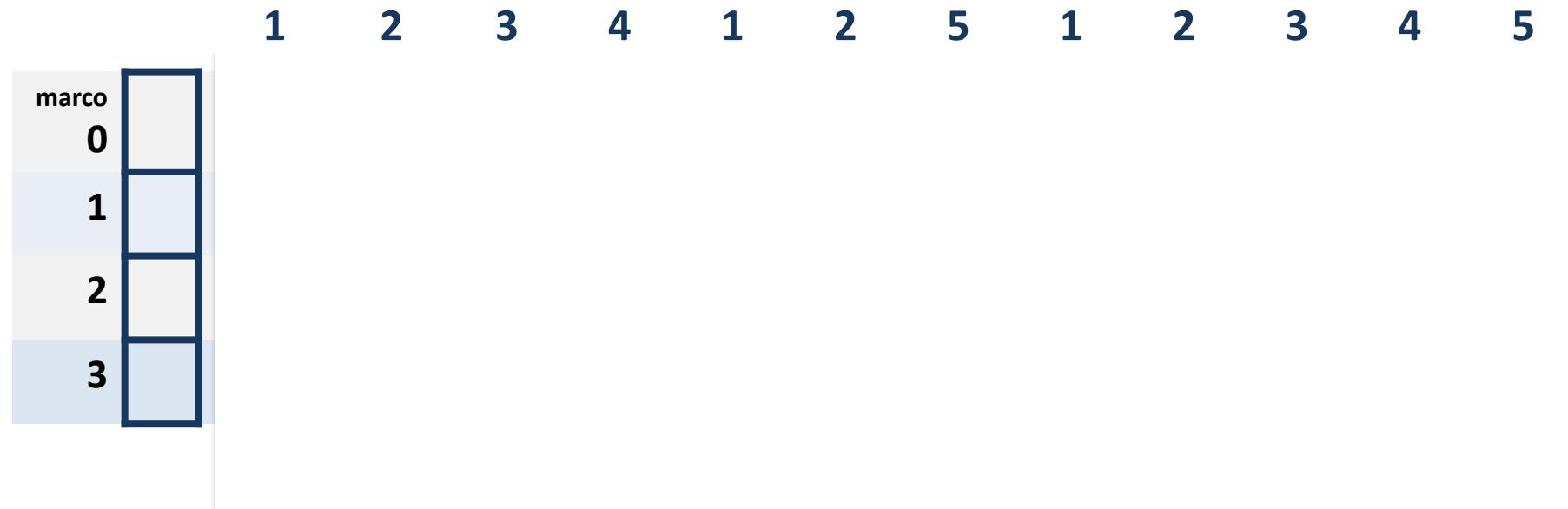
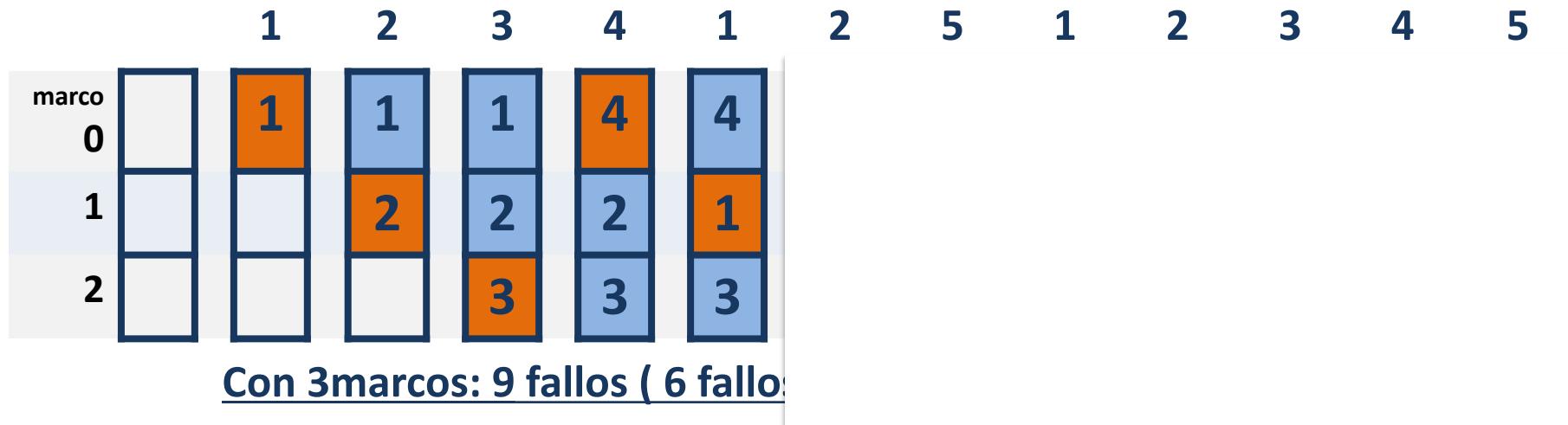
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



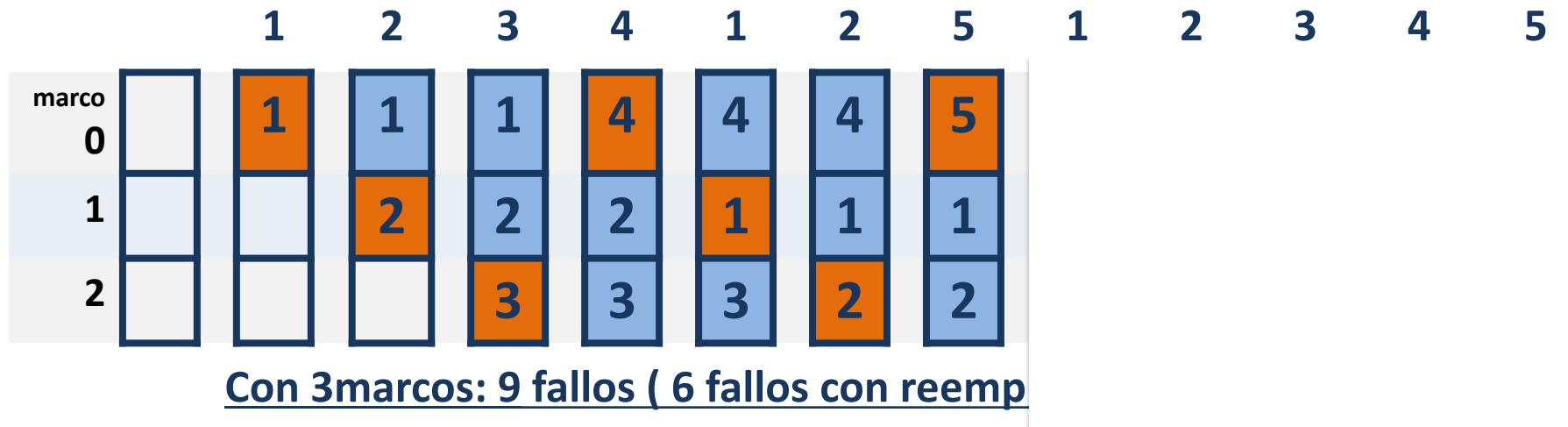
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



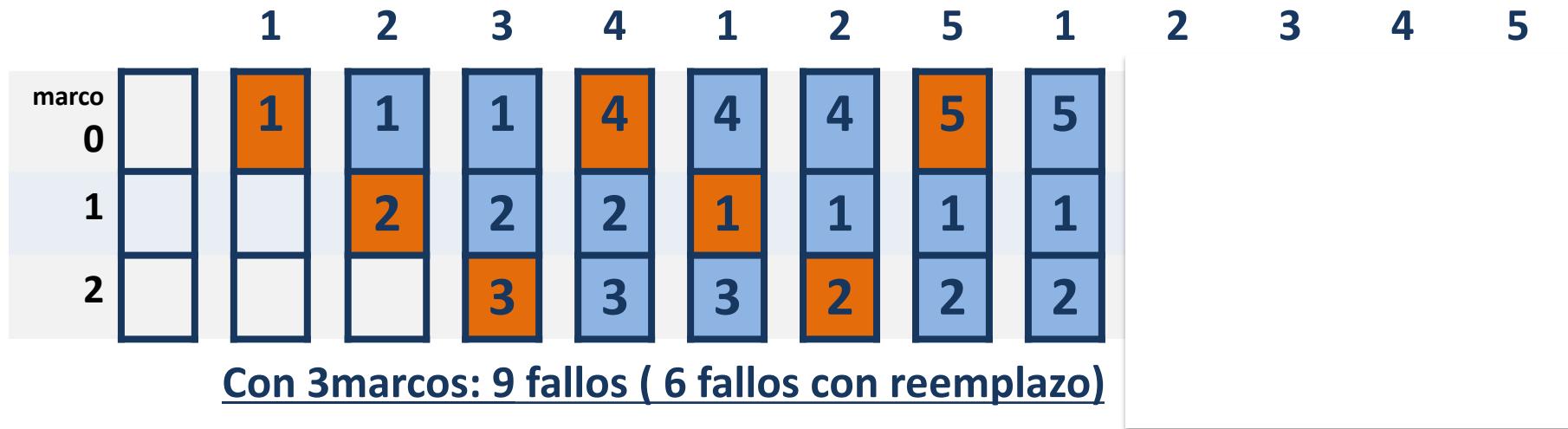
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



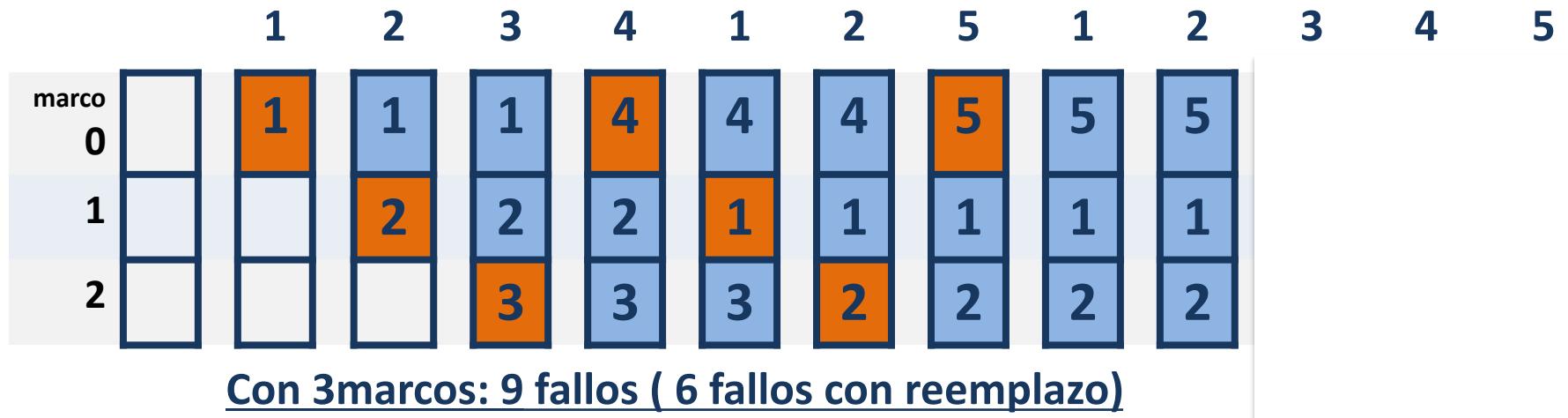
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



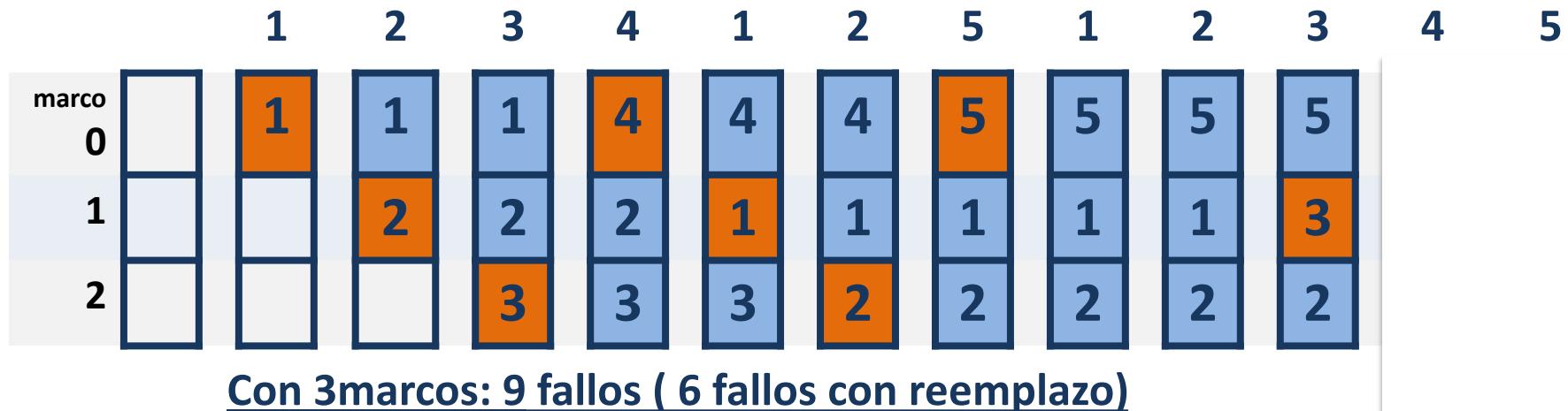
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



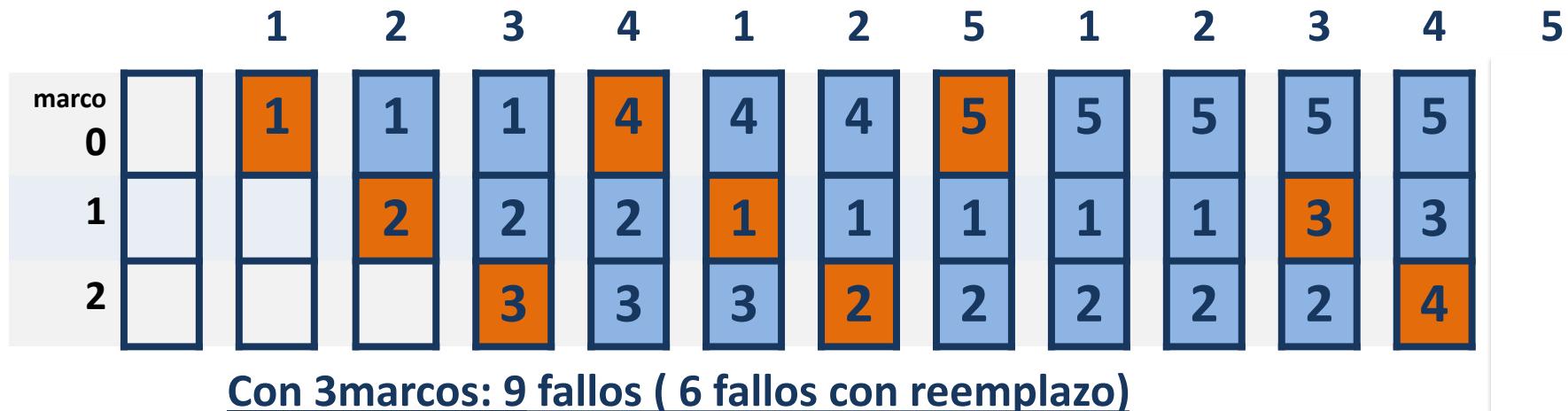
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



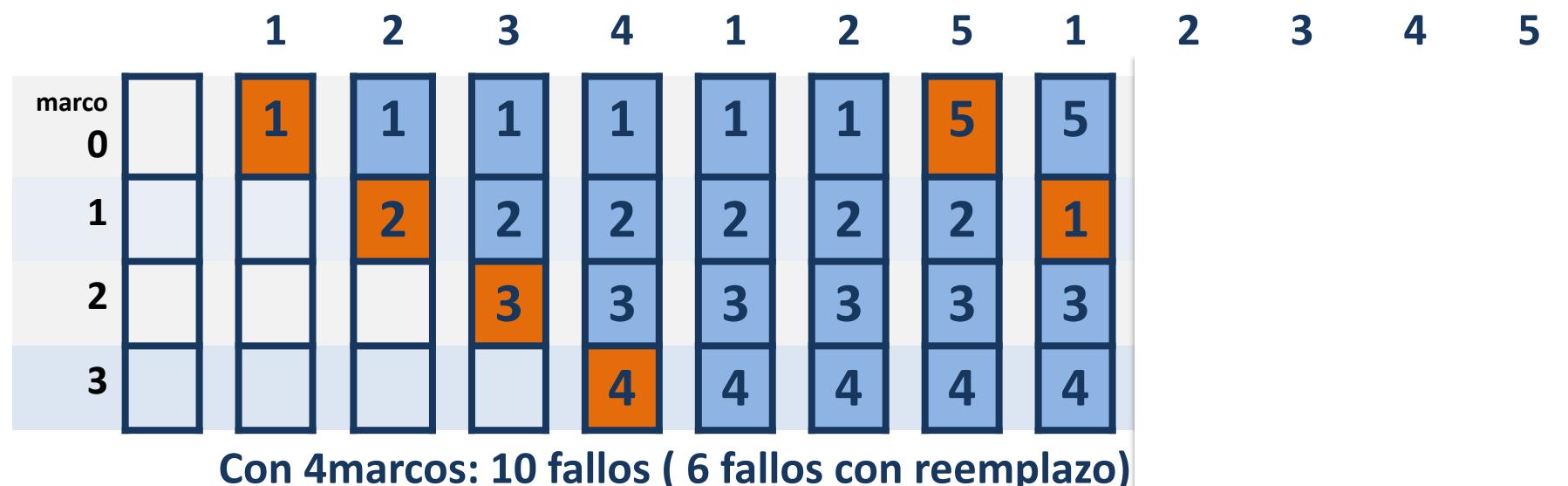
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



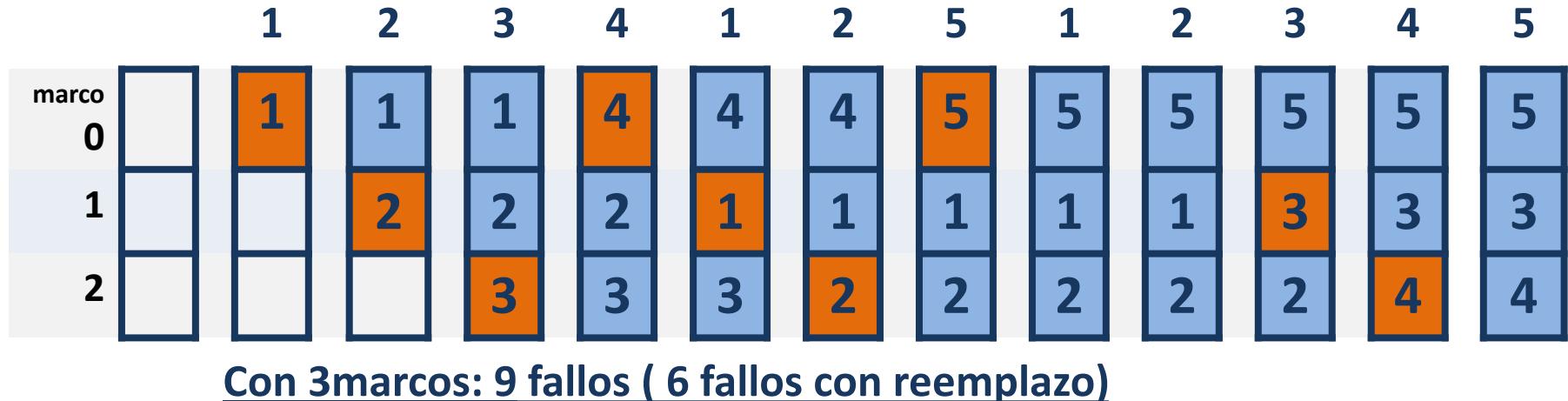
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



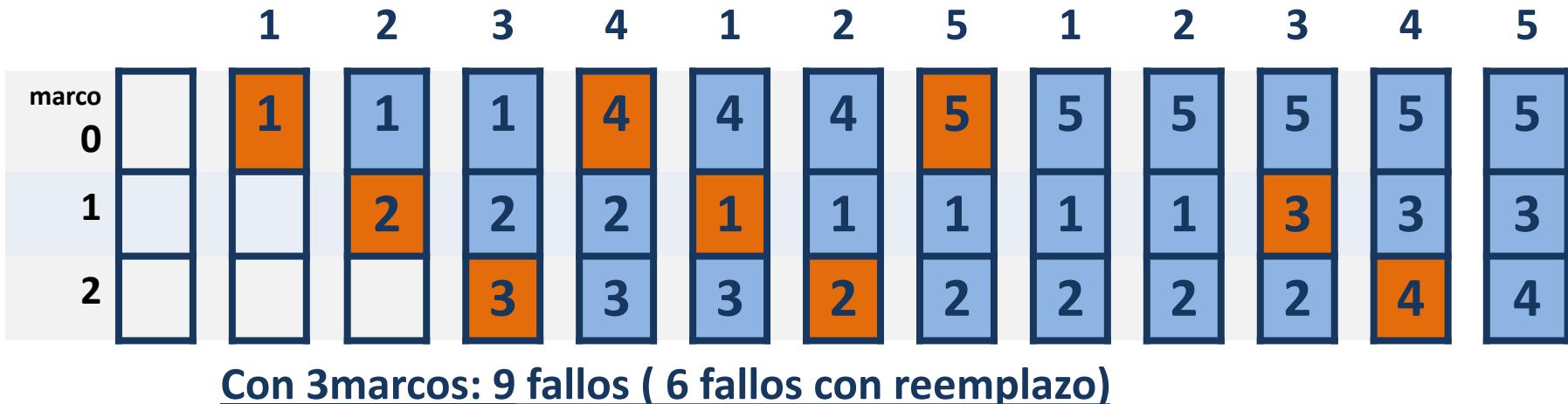
Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**



Algoritmo FIFO

- Página víctima: la que **lleva más tiempo** cargada **en memoria**.
- Presenta la **anomalía de Belady**

