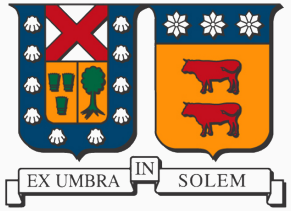


Sistemas Operativos

Memoria Virtual

Viktor Andrés Tapia Vásquez

Segundo Semestre 2021

Departamento de Informática, Campus SSJJ.

Índice de Contenidos

Introducción

Introducción

- Para ser ejecutados los procesos deben estar en memoria principal.
- Pero, rara vez son utilizados completamente.
  - ✓ Gestión de errores.
  - ✓ Rutinas inusuales.
  - ✓ Grandes estructuras de datos.
- El proceso completo podría no estar completo en memoria.
- ¿Qué beneficios obtendríamos?

Introducción

- **Memoria Virtual:** Separa la dirección lógica de la física.
- Solo una parte del proceso necesita estar en memoria.
- Espacio lógico mucho más grande que el físico.
- Permite compartir espacios de memoria entre muchos procesos.
- Creación de procesos más eficiente.
- Aumenta el grado de multiprogramación.
- Se requiere menos E/S para realizar swapping.

Introducción

- **Espacio Virtual:** Vista lógica de un proceso en memoria.
- La memoria física se organiza en frames.
- La MMU debe mapear de lógico a físico.
- Memoria Virtual se puede implementar de dos formas:
  - ✓ Paginación bajo demanda.
  - ✓ Segmentación bajo demanda.

Introducción

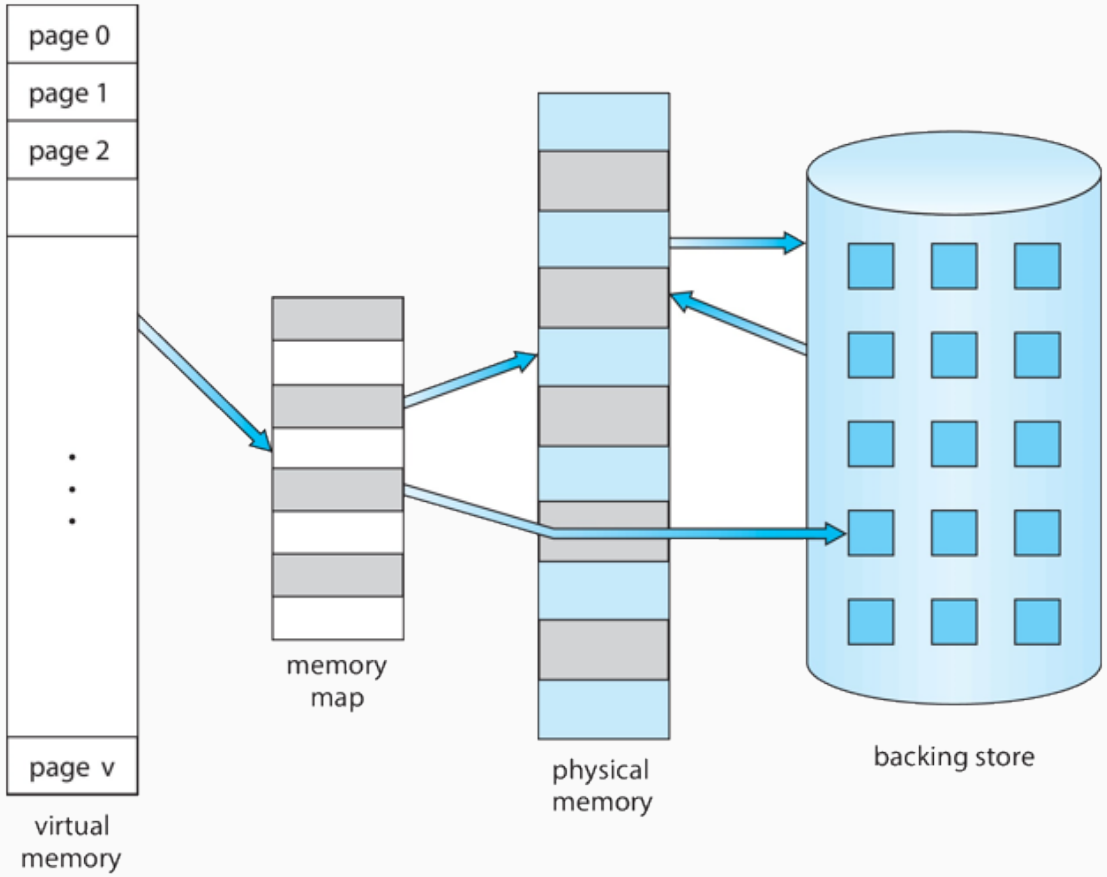


Figure 1: Memoria Virtual

Paginación Bajo Demanda

Paginación Bajo Demanda

- Consiste en llevar páginas a memoria solo cuando se necesiten.
  - ✓ Poco E/S requerido.
  - ✓ Se necesita menos memoria.
  - ✓ Respuesta rápida.
- Al ejecutar un proceso se realiza un intercambio de páginas.
- No se cambia el proceso completo.

Paginación Bajo Demanda

- Si una página se necesita se hace referencia a ella.
  - ✓ Referencia inválida implica abortar.
  - ✓ Si no está en memoria simplemente se carga.
- Se utiliza un intercambiador perezoso.
- Jamás cambia una página a menos que sea necesario.

Paginación Bajo Demanda

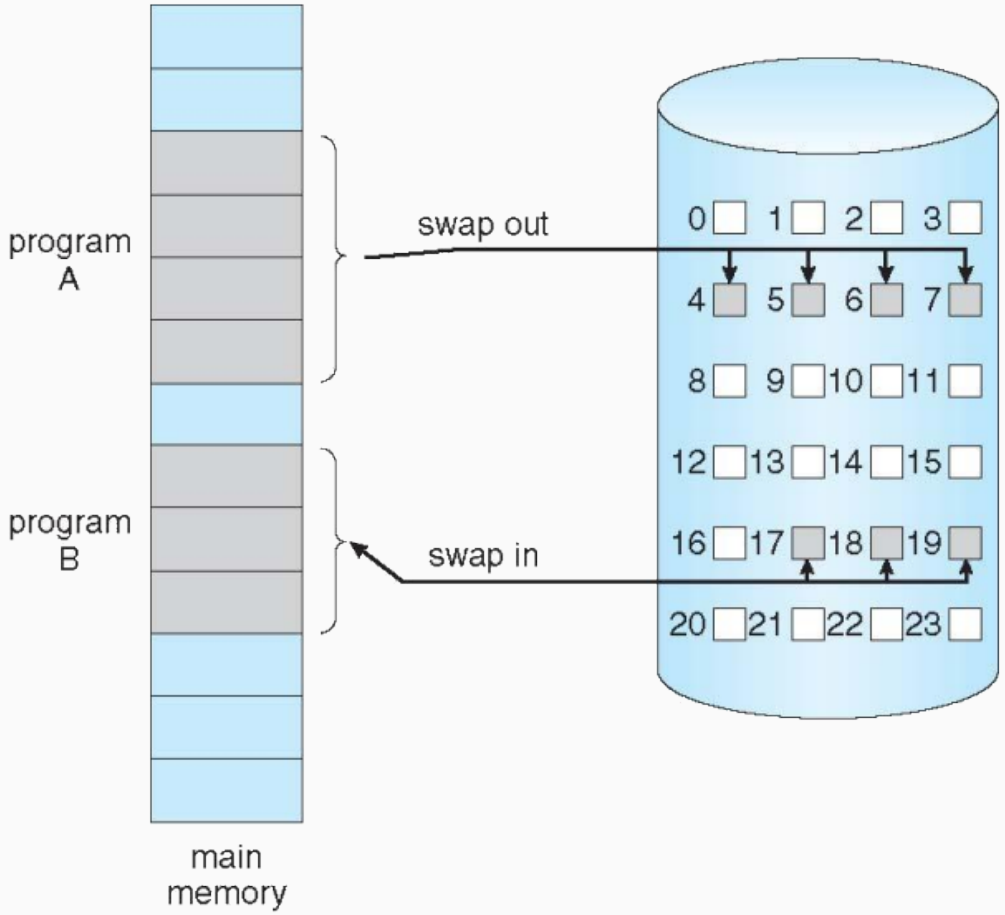


Figure 2: Paginación Bajo Demanda: Ejemplo

Paginación Bajo Demanda

- Se realiza una estimación a priori de las páginas a utilizar.
- Necesitamos soporte de hardware.
- La idea es saber qué páginas ya están en memoria.
- Utilizaremos la técnica del bit válido o inválido.
- El valor lo mantendremos en la tabla de páginas.
- Inicialmente todas están inválidas.
- Si se referencia una inválida se produce una **falla de página**.

Paginación Bajo Demanda

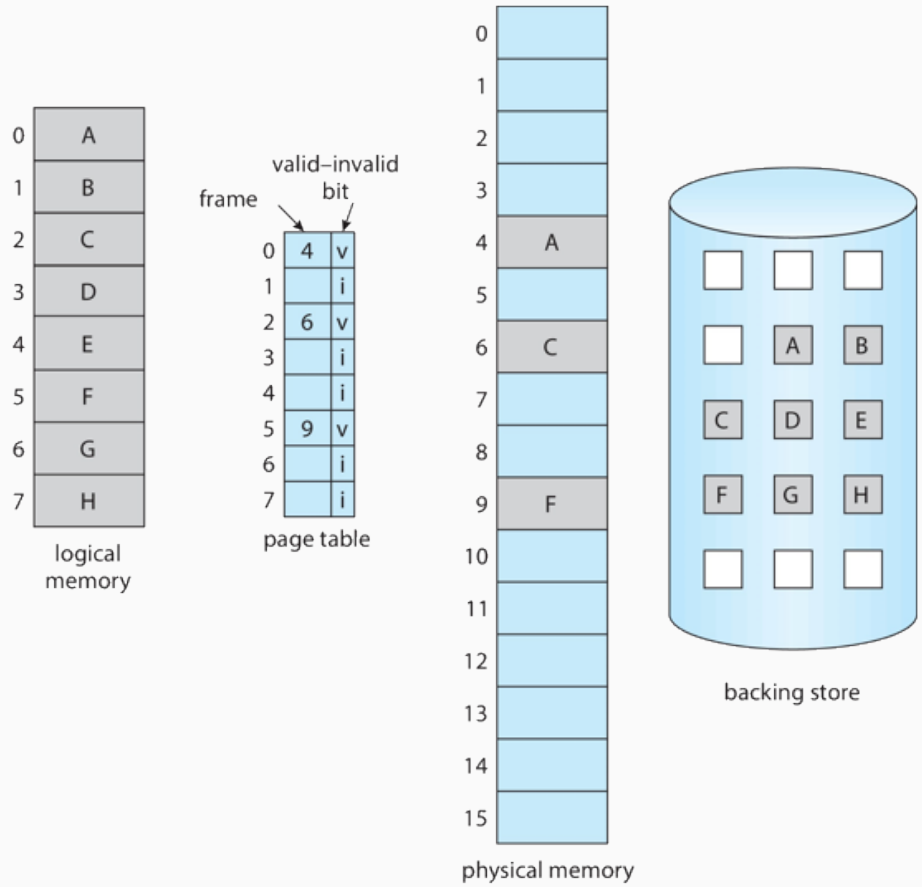


Figure 3: Tabla de Páginas

Paginación Bajo Demanda

- Cuando se referencia una página:
- Validamos el PCB de un proceso para ver si es válida.
- Si es inválida, se termina el proceso.
- Caso contrario se carga la página en memoria.
- Se busca un frame libre.
- Se lee la página desde el disco y se asigna.
- Se actualiza la PCB y la tabla de página.
- Reiniciamos la instrucción interrumpida.

Paginación Bajo Demanda

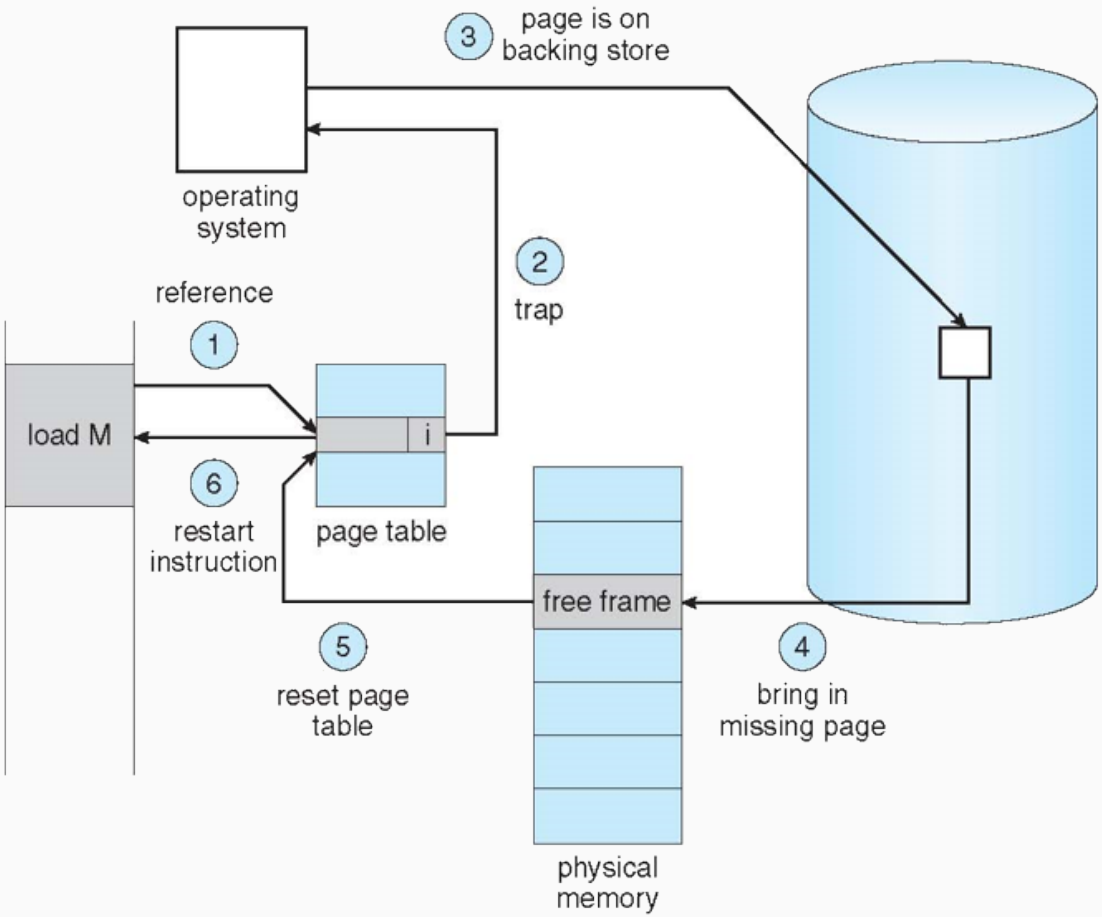


Figure 4: Gestión de una falla de página

Paginación Bajo Demanda

- Caso Extremo: Un proceso sin páginas en memoria.
- Esto se conoce como **paginación pura**.
- Nunca se carga una página a menos que sea requerida.
- Todas las páginas producen una falla (en este caso).
- Una instrucción puede hacer referencia a varias páginas.

Paginación Bajo Demanda

- Esta técnica requiere soporte de hardware:
  - ✓ **Tabla de páginas:** Se agrega el bit V o I.
  - ✓ **Memoria respaldo:** Donde se almacenan páginas.

Requisito Fundamental

Poder reiniciar cualquier instrucción después de un fallo de página.

- El SO debe mantener una lista con los frames disponibles.
- Cuando el SO inicia, toda la memoria es parte de la lista.

Paginación Bajo Demanda - Desempeño

¿Cómo andará el rendimiento? Revisemos el peor caso:

- Trap al SO.
- Guardar datos de registros de apoyo.
- Actualizar la PCB del proceso involucrado.
- Determinar que la interrupción fue por fallo de página.
- Validar si la referencia a la página es válida.
- Si lo es, determinar su ubicación en el disco.

Paginación Bajo Demanda - Desempeño

¿Cómo andará el rendimiento? Revisemos el peor caso:

- Leer desde el disco hasta un frame disponible:
  - ✓ Espera en cola hasta que el E/S responda.
  - ✓ Esperar el tiempo de búsqueda del dispositivo.
  - ✓ Transferir la página.
- Esperar mientras la CPU es utilizada por otros.
- E/S Terminado.

Paginación Bajo Demanda - Desempeño

¿Cómo andará el rendimiento? Revisemos el peor caso:

- Guardar datos de registros de apoyo.
- Actualizar PCB proceso saliente.
- Determinar que la interrupción fue de disco.
- Actualizar tabla de páginas.
- Esperar asignación de CPU.
- Restaurar datos a los registros de apoyo.
- Resumir la interrupción inicial.

Paginación Bajo Demanda - Desempeño

- En resumen, 3 grandes actividades:
  - ✓ Interrupción de servicio.
  - ✓ Lectura de página desde el disco.
  - ✓ Reiniciar el proceso.
- Ratio de rango de fallas: $0 \leq p \leq 1$.
  - ✓ Si $p = 0$ no hay fallas.
  - ✓ Si $p = 1$ toda referencia es una falla.

Paginación Bajo Demanda - Desempeño

- Tiempo Efectivo de Acceso:

$$EAT = (1 - p) \times MA + p \times W$$

- En donde:
  - ✓ MA: Tiempo acceso a memoria.
  - ✓ W: Overhead Falla de Página+SwapIn+SwapOut.
- Cada falla de página es costosa.

Paginación Bajo Demanda - Copy on Write

- **COW:** Procesos, en la creación, comparten páginas en memoria.
- Cuando un procesos modifica una página, se genera una copia.
- Permite creación de procesos más eficiente.
- vfork(): Llamada al sistema variante de fork()
  - ✓ El padre se suspende y el hijo utiliza el espacio del padre.
  - ✓ El hijo debe utilizar exec()

Paginación Bajo Demanda - Copy on Write

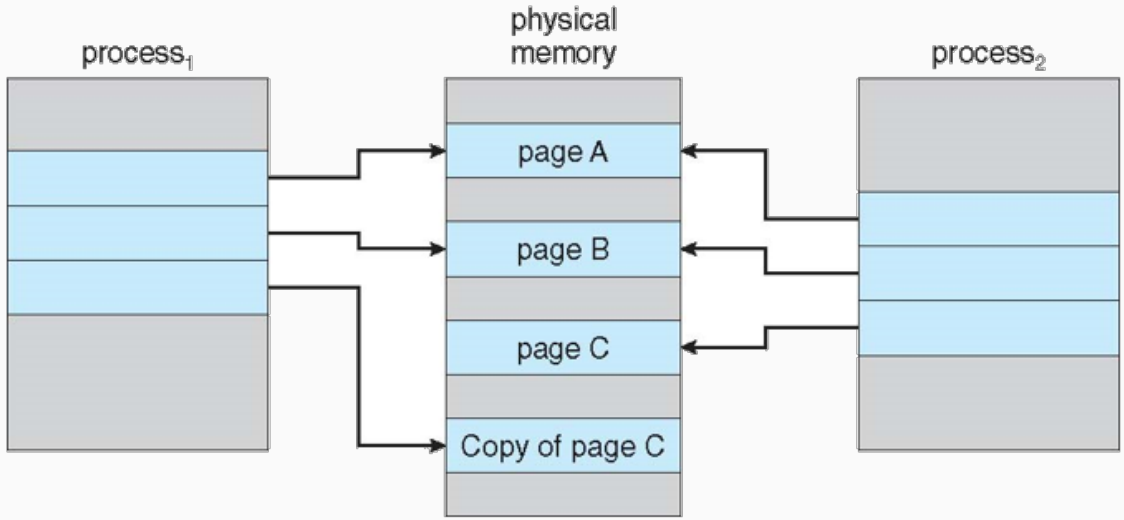


Figure 5: Modificación de página COW

Reemplazo de Páginas

Reemplazo de Páginas

- (+) multiprogramación implica sobreasignación de memoria.
- Esto implica ocupar todos los frames disponibles.
- Si se produce una nueva falla ¿qué pasa?
- Buscamos una página desocupada y la llevamos al disco.

Reemplazo de Páginas

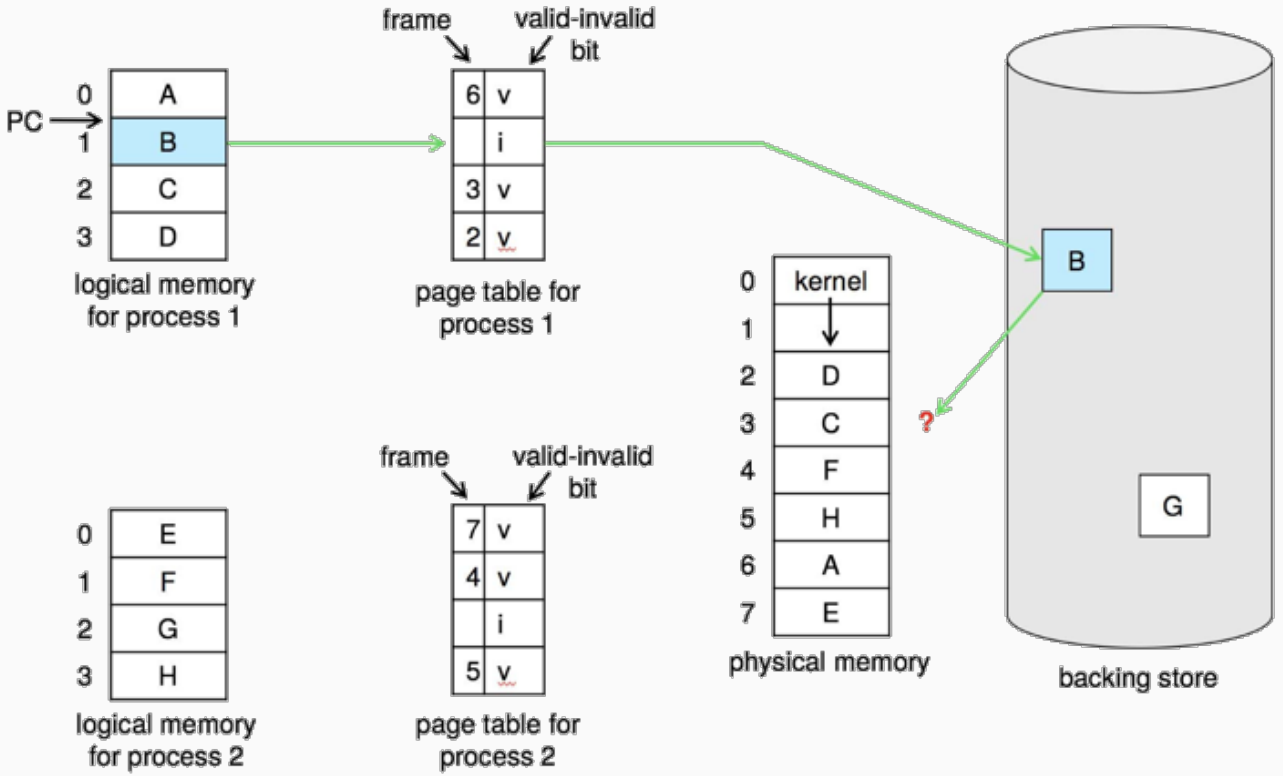


Figure 6: Reemplazo de Páginas

Reemplazo de Páginas

- Requerimos:
  - ✓ Algoritmos.
  - ✓ Desempeño: Minimizar el número de fallas.
- Ojo: Algunas páginas se traerán varias veces a memoria.

Reemplazo de Páginas - Técnica Básica

- Buscamos la página deseada en el disco.
- Buscamos un frame disponible, si existe:
  - ✓ Se usa.
  - ✓ Si no, se utiliza el algoritmo de reemplazo.
- Se carga la página en el frame y se actualiza la información.
- Se reinicia la ejecución del proceso.

Reemplazo de Páginas

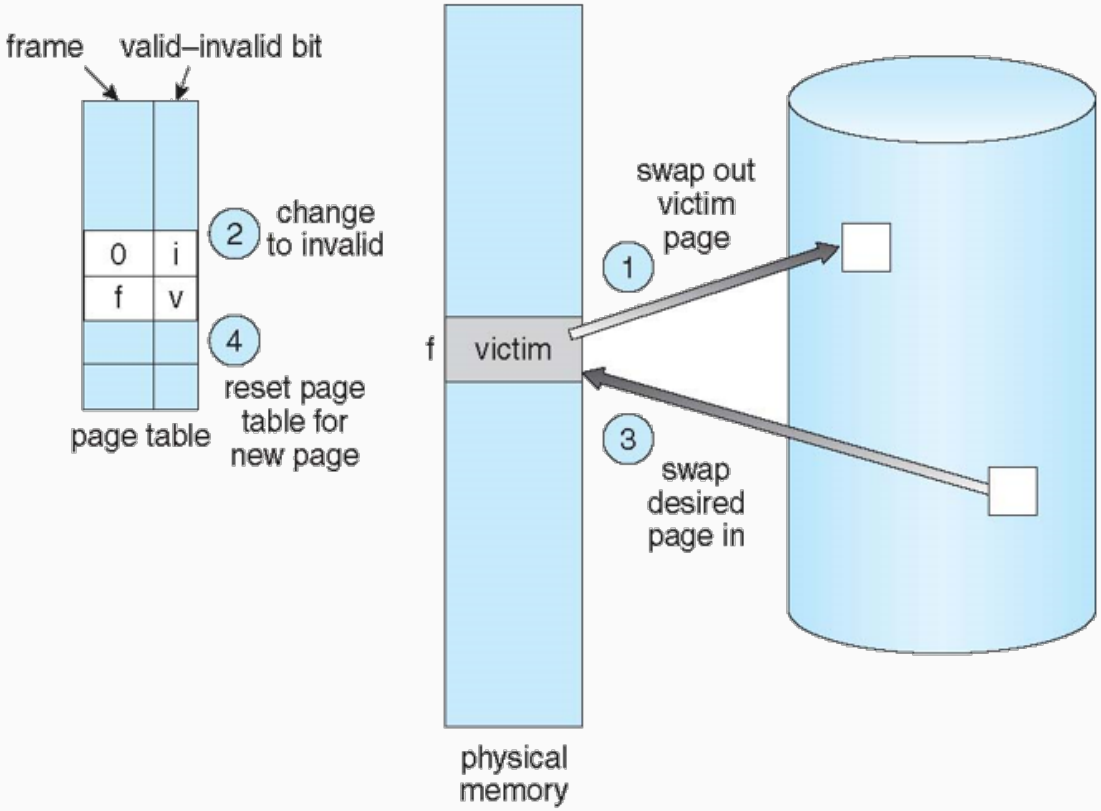


Figure 7: Selección de Víctima

Reemplazo de Páginas - Algoritmos

- Existen diversos algoritmos.
- El objetivo es minimizar el número de fallas de página.
- Se corren sobre un string de referencia.
  - ✓ Cada elemento corresponde al número de la página.
  - ✓ Referencias inmediatas a una página no generan falla.
  - ✓ El resultado depende del total de frames.

Reemplazo de Páginas - Algoritmos

- Ejemplo. Considere la siguiente secuencia de direcciones:

  $$0100, 0432, 0101, 0612, 0102, 0103, 0611, 0102, 0610, 0602, 0105, ...$$

- El string resultante: $1, 4, 1, 6, 1, 6, 1, 6, 1$
- Necesitamos conocer el número de frames disponibles.
- Notación $M(m, r) = p$. En donde:
  - ✓ m :total de frames.
  - ✓ r :índice del string.
  - ✓ p :páginas en memoria para los valores dados.

Reemplazo de Páginas - Algoritmos

- ¿Mientras más frames menos fallas?
- Del ejemplo anterior: string $1, 4, 1, 6, 1, 6, 1, 6, 1$
  - ✓ Con 3 frames se producen 3 fallas.
  - ✓ Con 1 frame se producen 9 fallas.

Reemplazo de Páginas - Algoritmos

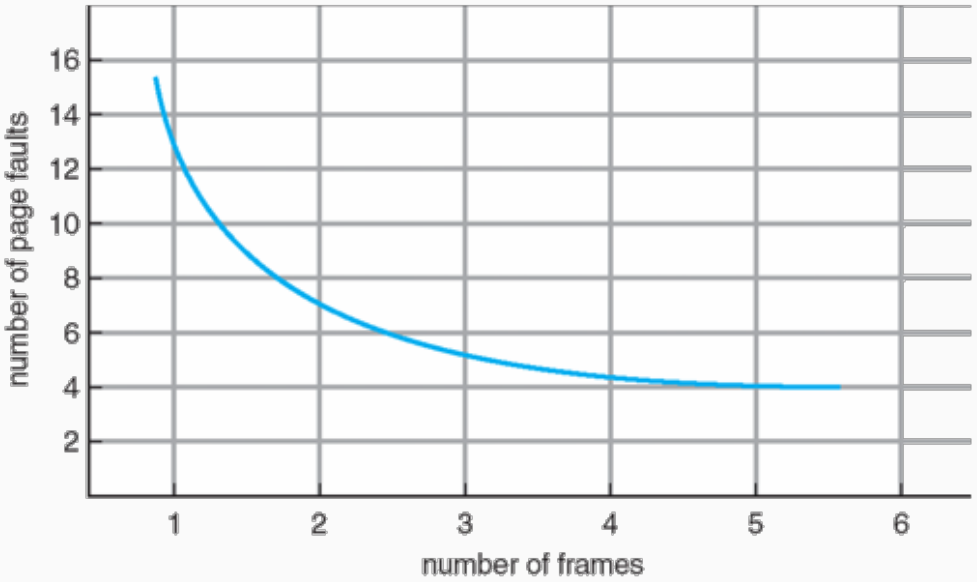


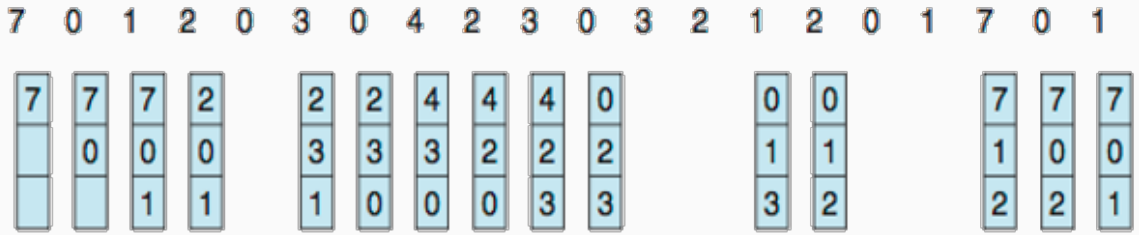
Figure 8: Fallos de página V/S Frames

Reemplazo de Páginas - FIFO

- El más simple de todos.
- La ~~vístima~~ es la primera página en llegar.
- Considerando el siguiente string de referencia:

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

- Y 3 frames disponibles.



- Se producen 15 fallas de página.

Reemplazo de Páginas - FIFO

- Sufre de la anomalía de Belady.
  - ✓ Más frames implican más fallos de páginas.
- Considere el string: $1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5$
  - ✓ Con 3 frames.
  - ✓ Con 4 frames.

Reemplazo de Páginas - FIFO

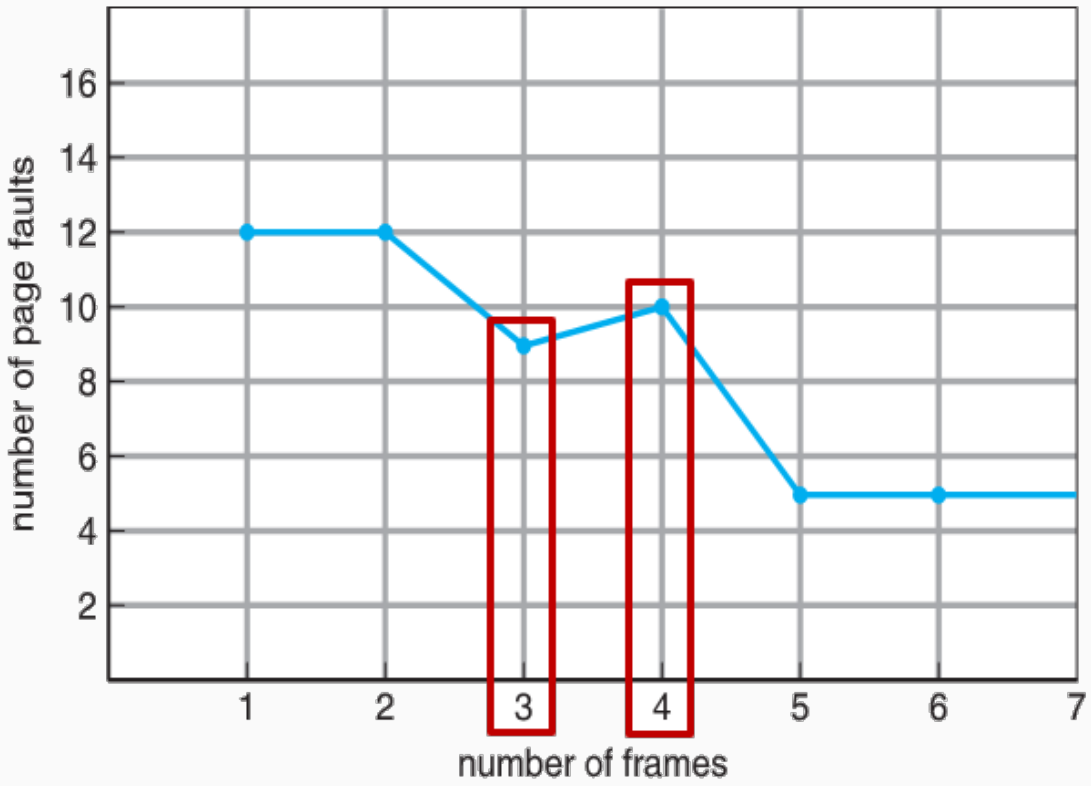


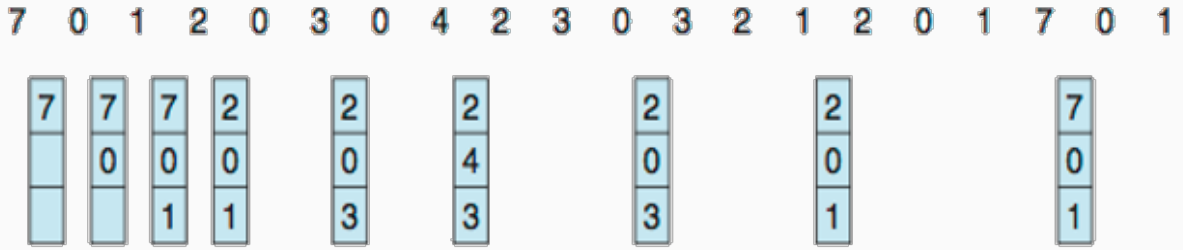
Figure 9: Anomalía de Belady

Reemplazo de Páginas - Óptimo

- Reemplaza la página que no se utilizará en el mayor tiempo.
- Considerando el siguiente string de referencia:

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

- Y 3 frames disponibles.



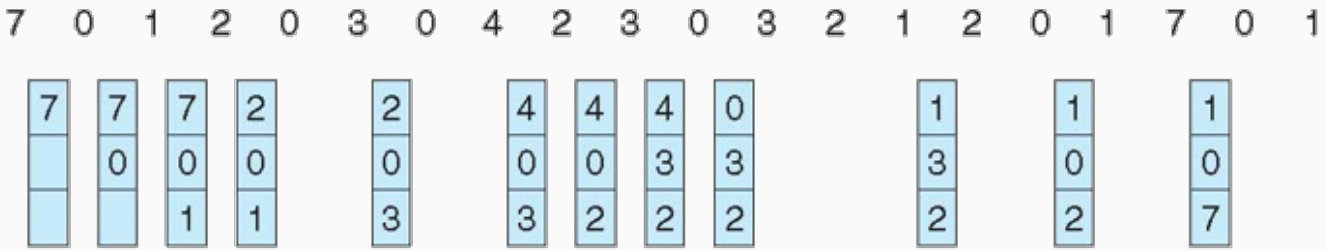
- Se producen 9 fallas de página.
- ¿El problema? No se puede predecir el futuro.

Reemplazo de Páginas - LRU

- Utilizamos la información del pasado.
- Reemplaza la página que no ha sido utilizada en el mayor tiempo.
- Considerando el siguiente string de referencia:

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

- Y 3 frames disponibles.



- Se producen 12 fallas de página.

Reemplazo de Páginas - LRU

¿Cómo se puede implementar?

- Contador:
  - ✓ Cada página cuenta con un contador.
  - ✓ Cada referencia a la página lo incrementa.
  - ✓ Cuando hay que reemplazar se busca el más pequeño.
- Stack:
  - ✓ Se almacenan las páginas referenciadas.
  - ✓ La última referencia va al principio.
  - ✓ No es necesario buscar para reemplazar.
- LRU y OPT no sufren la anomalía de Belady.

Reemplazo de Páginas - LRU

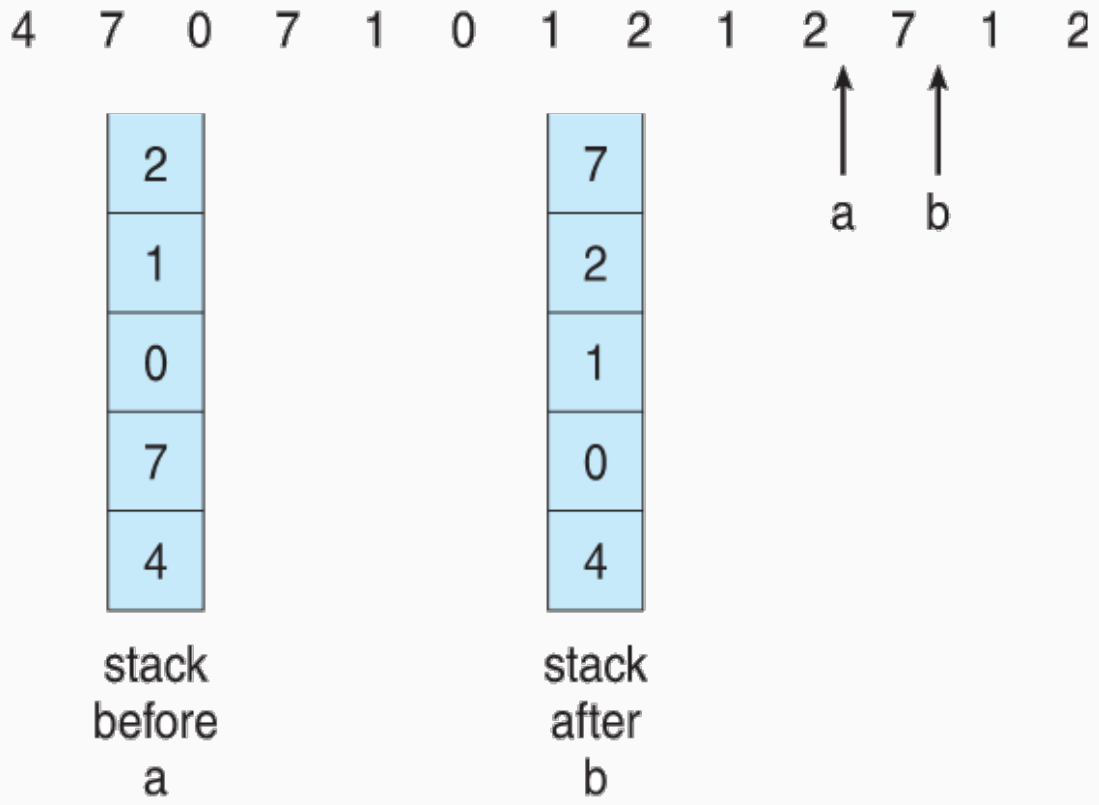


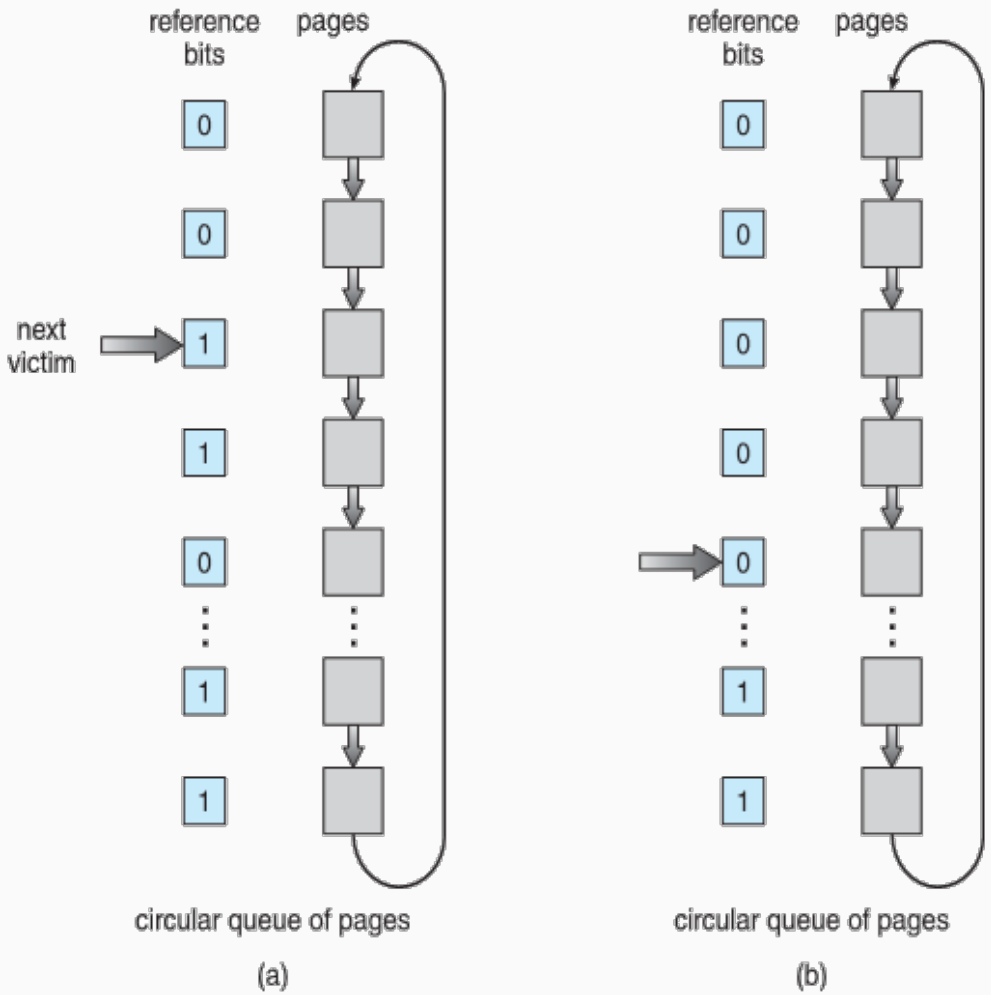
Figure 10: Implementación de LRU con Stack

Reemplazo de Páginas - LRU

Existen algunas variantes:

- Bit de Referencia:
  - ✓ Inicialmente cada página tiene el bit en 0.
  - ✓ Cuando se referencia se cambia el valor a 1.
  - ✓ Reemplazar cualquiera con el bit en 0.
- Segunda Oportunidad:
  - ✓ FIFO y utilizando el bit de referencia.
  - ✓ Si se referencia y el bit es cero se reemplaza.
  - ✓ Caso contrario, dejar el bit en 0 y mantener en memoria.
  - ✓ Reemplazar la siguiente sujeta a las mismas reglas.

Reemplazo de Páginas - LRU



Asignación de Frames

Asignación de Frames

- ¿Cómo se asignan los frames disponibles?
- Cada proceso requiere un mínimo número de frames.
- El SO debiése asignar los frames suficientes.
- Existen dos grandes esquemas (y varias variaciones):
  - ✓ Asignación Fija.
  - ✓ Asignación por Prioridad.

Asignación de Frames

- Fija:
  - ✓ Todos los procesos por igual.
  - ✓ ¿Será eficiente?
  - ✓ Ejemplo: 100 frames y 5 procesos. 20 frames c/u.
- Prioridad:
  - ✓ Mayor prioridad, más frames.
  - ✓ Puede ser por el tamaño del proceso.

Asignación de Frames

La política de reemplazo de páginas puede ser:

- Local:
  - ✓ Se asignan únicamente los frames definidos inicialmente.
  - ✓ Mejor desempeño desde el punto de vista del proceso.
- Global:
  - ✓ Se puede asignar cualquier frame.
  - ✓ Los procesos quitan frames.
  - ✓ Aumenta el throughput.

Thrashing y Working Set

Thrashing

- Proceso sin frames puede generar muchas fallas de página.
- Esto genera:
  - ✓ Bajo uso de CPU.
  - ✓ Aumento del grado de multiprogramación.
  - ✓ Se incorpora un nuevo proceso.
- **Thrashing**: Proceso ocupado realizando Swap In y Swap Out.

Thrashing

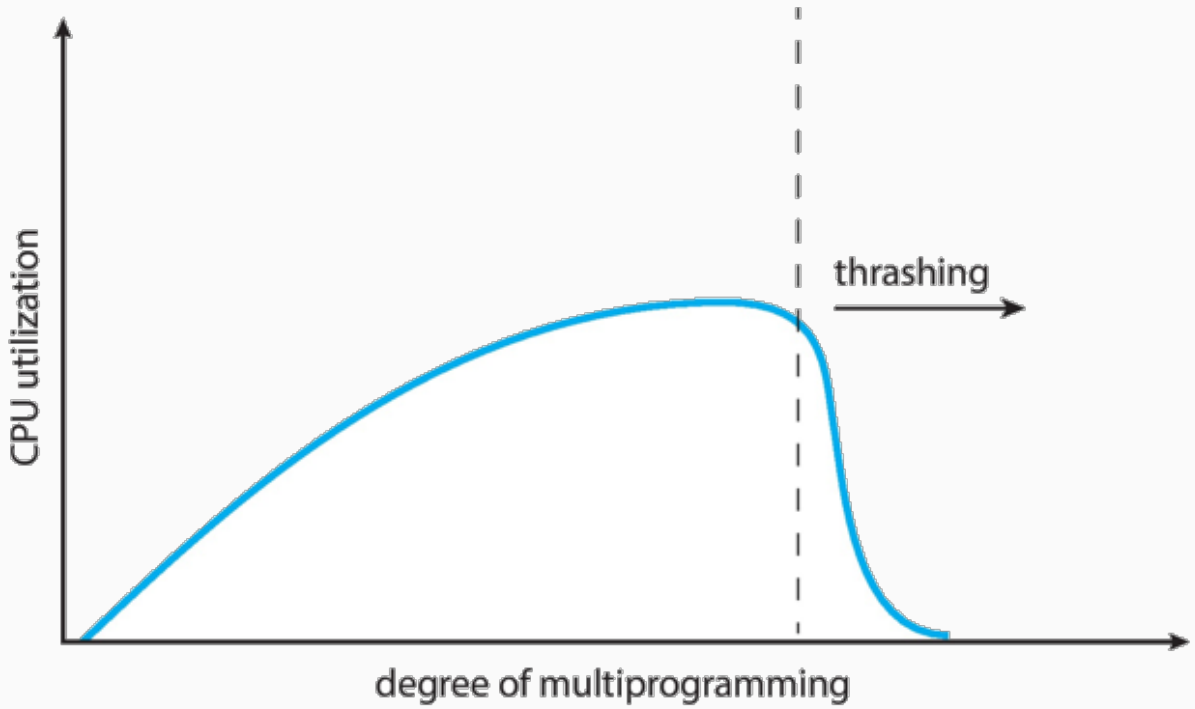


Figure 11: Thrashing

Thrashing

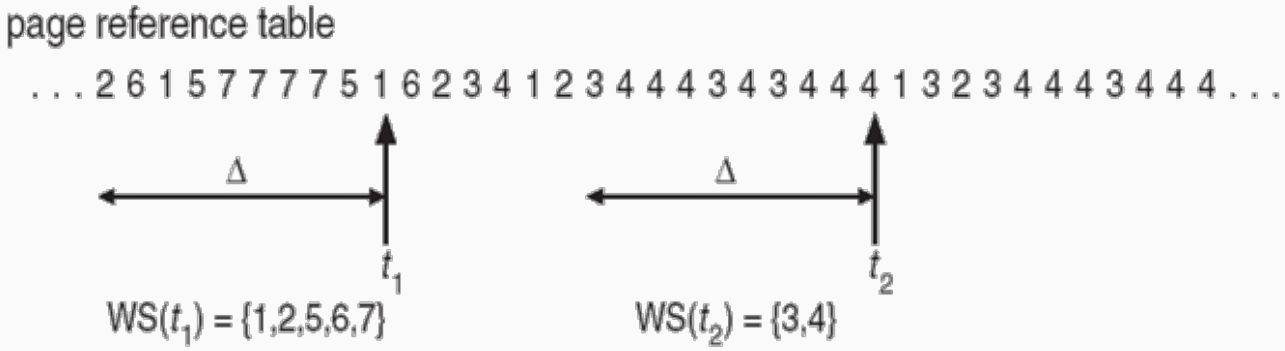
- Podemos limitar (no resolver) el problema:
  - ✓ Utilizando algoritmos de sustitución local o de prioridades.
  - ✓ Forzar a que los procesos en thrashing no quiten frames.
- Problema: Los procesos siguen esperando.

Modelo Working Set

- Δ = Número fijo de páginas referenciadas.
- WSS_i = Número total de páginas referenciadas en el Δ más reciente para el proceso P_i (cambia durante el tiempo).
  - ✓ Si Δ es pequeño no abarcará la localidad completa.
  - ✓ Si Δ es grande, abarcará varias localidades.
  - ✓ Si $\Delta = \infty$ abarcará el programa completo.
- Sea $D = \Sigma WSS_i$ = total frames demandados.

Modelo Working Set

- Si $D > m \Rightarrow$ Thrashing
- Cuando esto ocurre, suspender o sacar el proceso.



Ajuste de Fallas de Página

- Definir una tasa aceptable de fallas de página.
  - ✓ Si la tasa es baja, los procesos pueden perder frames.
  - ✓ Si la tasa es alta, deben ganar frames.

Ajuste de Fallas de Página

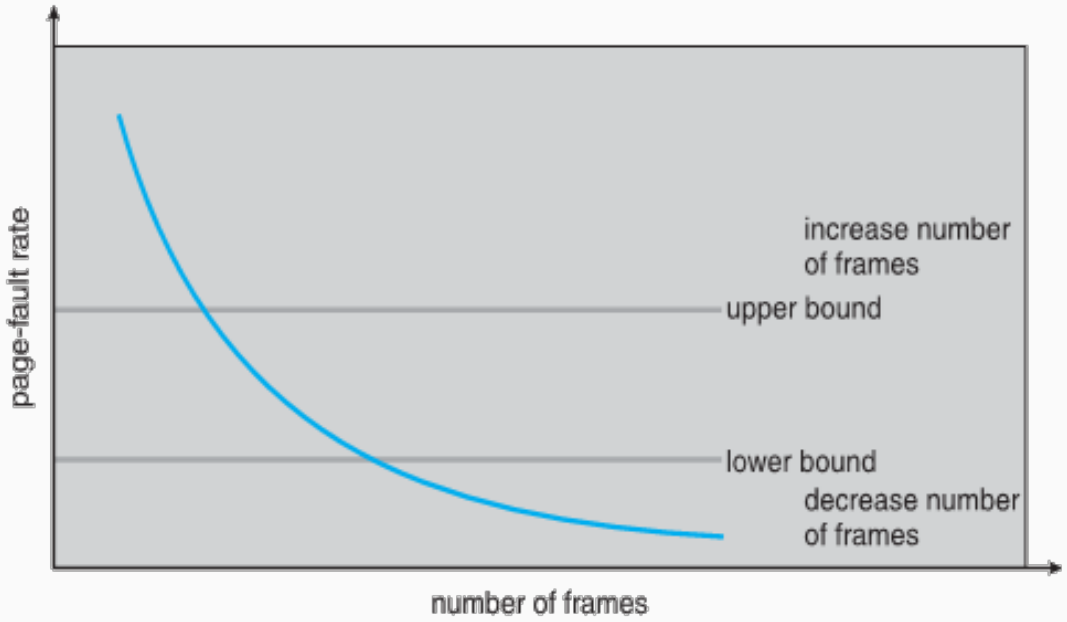
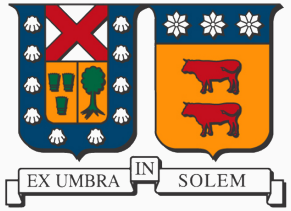


Figure 12: Ajuste de fallas en el tiempo



Sistemas Operativos

Memoria Virtual

Viktor Andrés Tapia Vásquez

Segundo Semestre 2021

Departamento de Informática, Campus SSJJ.