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 12

Memoria Virtual (II)

fSO

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DISCA

- **Objetivo**
  - Estudiar el algoritmo de reemplazo de paginas de **2ª oportunidad** como aproximación al LRU
  - Conocer la problemática de la **hiperpaginación** y las soluciones a la misma
  - Analizar las técnicas de **gestión de marcos** de memoria

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

Contenido

- **Algoritmo de reemplazo de 2ª Oportunidad**
- Asignación de marcos
- Hiperpaginación
- Reserva de Marcos

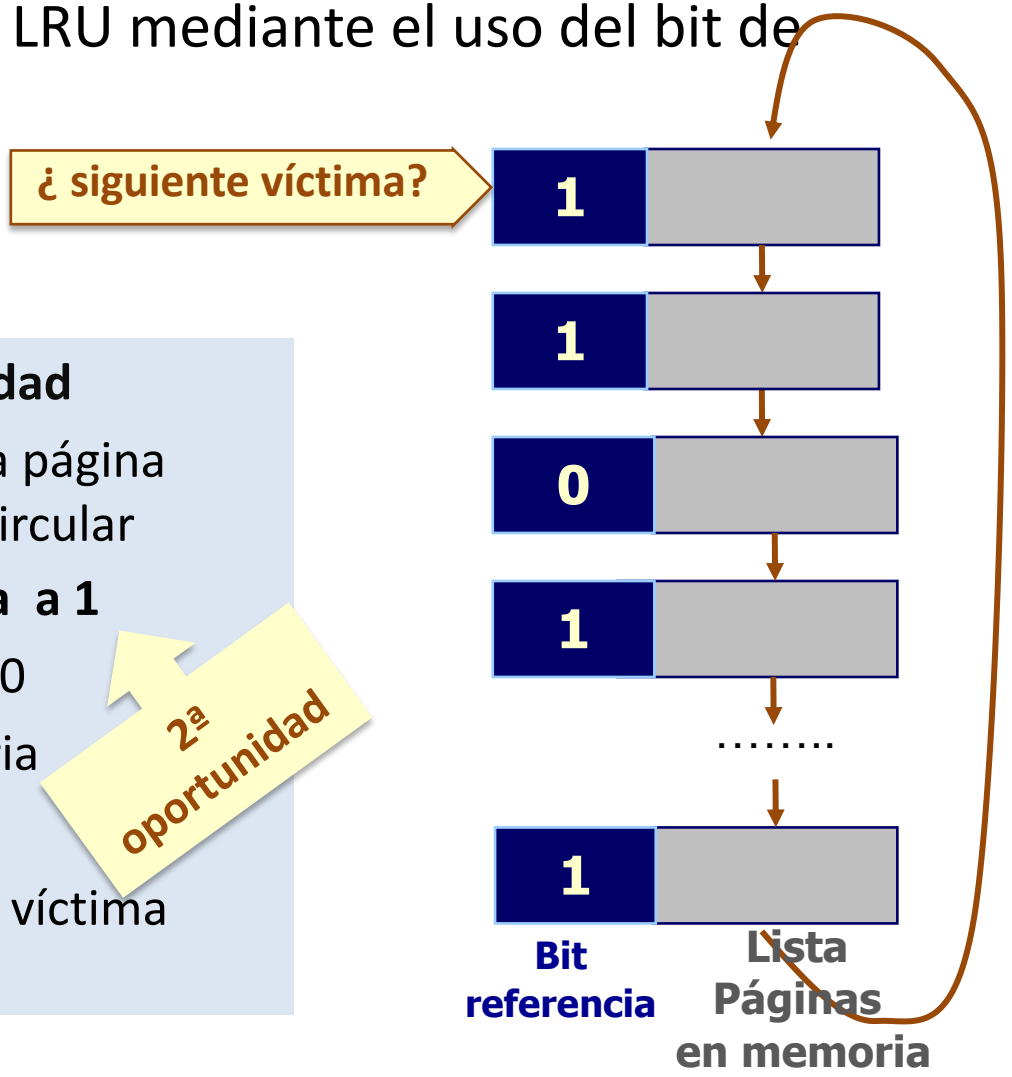
Algoritmo de 2ª oportunidad

- Soportar un algoritmo de reemplazo LRU es caro e ineficiente
- Solución: Aproximación a LRU mediante el uso del bit de referencia

Algoritmo de Segunda Oportunidad

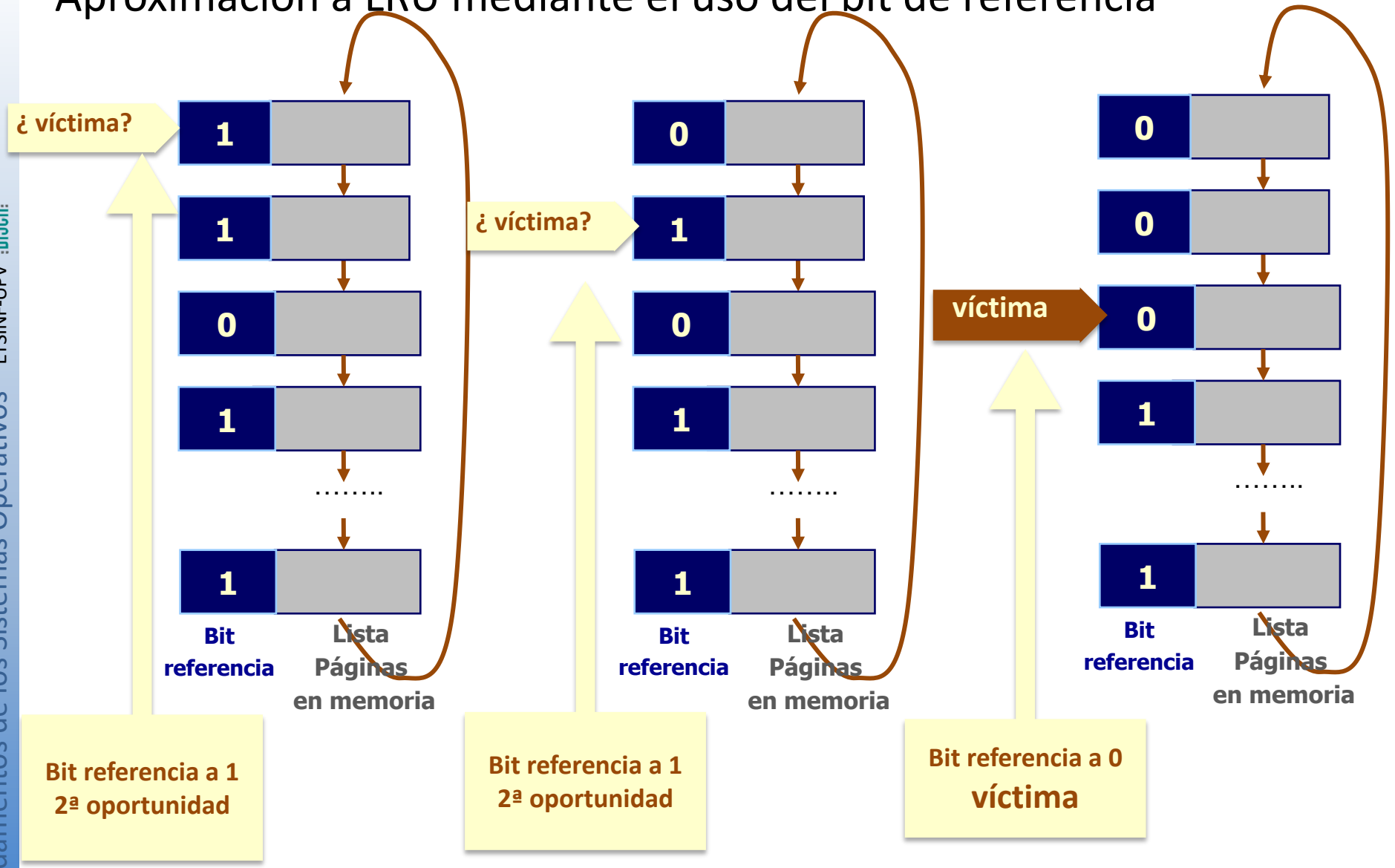
Consulta el bit de referencia de la página ordenadamente y de forma circular

- **Si bit de referencia de página a 1**
  - Poner bit de referencia a 0
  - Dejar la página en memoria
- **Si el bit de referencia es 0**
  - Se elige esta página como víctima



Algoritmo de 2ª oportunidad

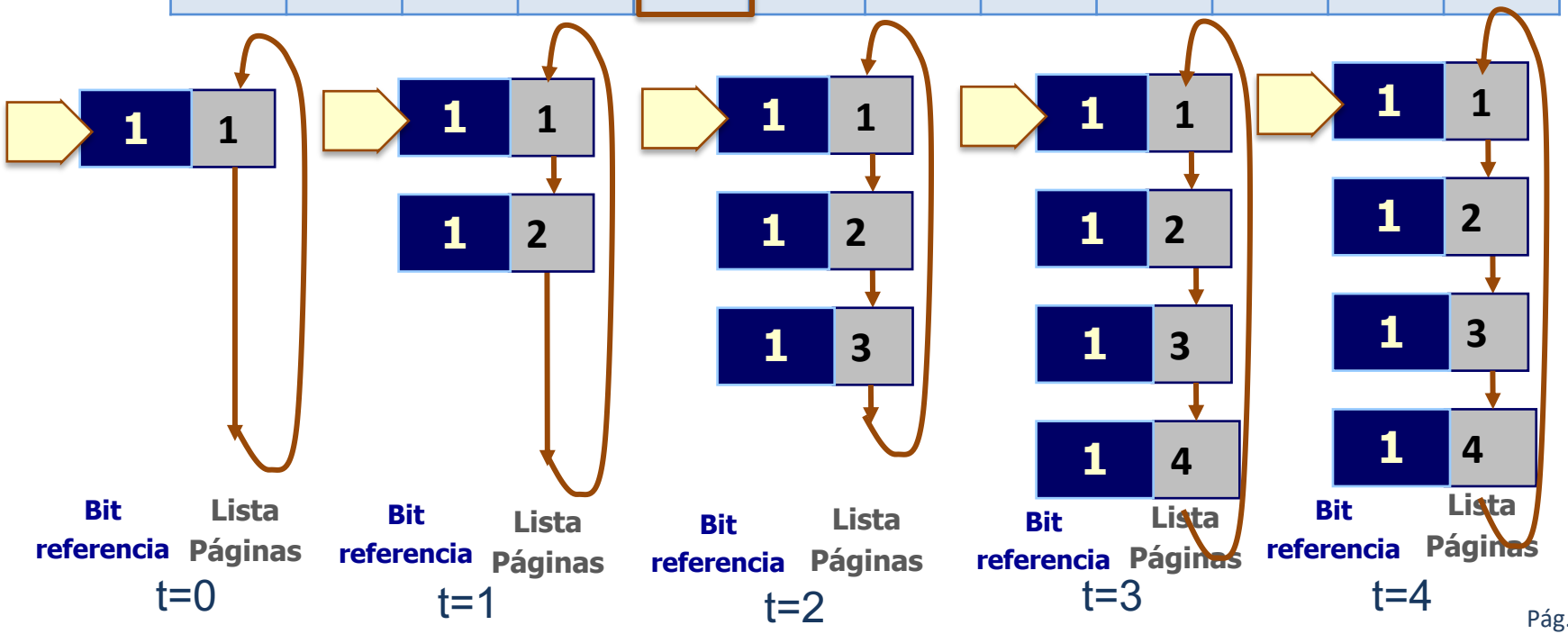
Aproximación a LRU mediante el uso del bit de referencia



Algoritmo de 2ª oportunidad

Ejemplo: Memoria Principal de 4 marcos

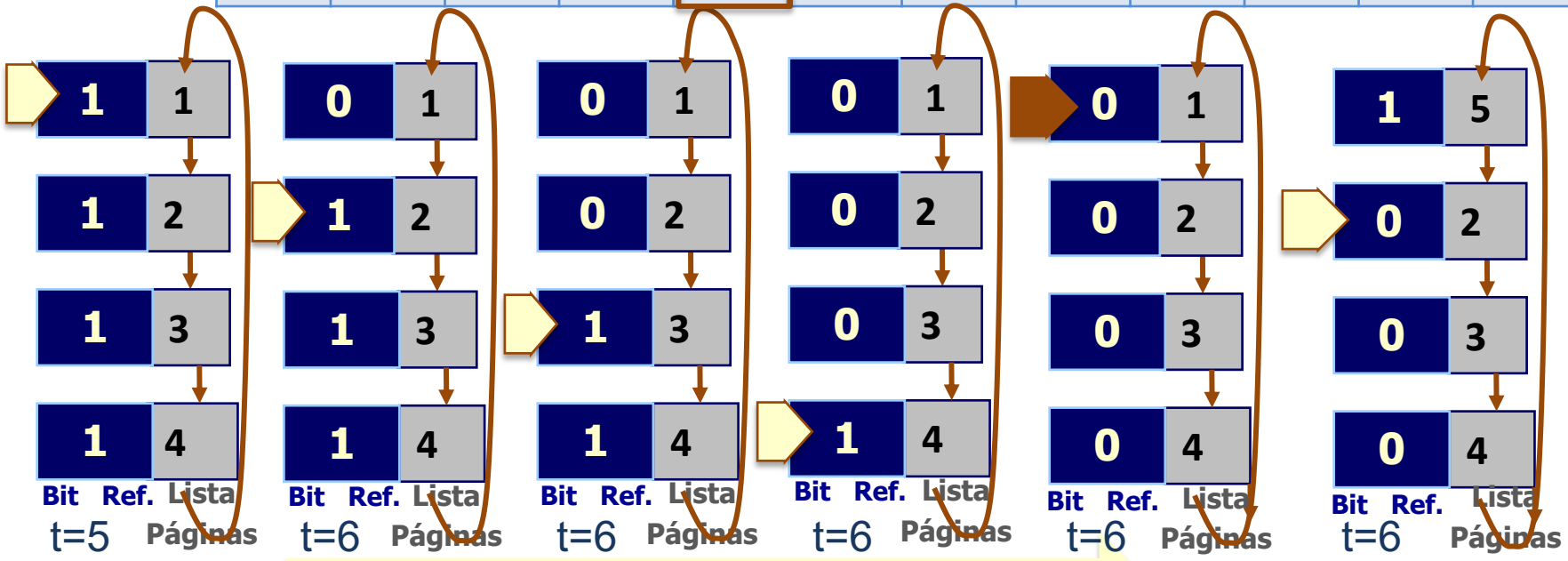
T	0	1	2	3	4	5	6	7	8	9	10
Serie de referencias	1	2	3	4	1	2	5	2	3	1	2
marco 0	1	1	1	1	1						
marco 1		2	2	2	2						
marco 2			3	3	3						
marco 3				4	4						



Algoritmo de 2ª oportunidad

Ejemplo: Memoria Principal de 4 marcos

T	0	1	2	3	4	5	6	7	8	9	10
Serie de referencias	1	2	3	4	1	2	5	2	3	1	2
marco 0	1	1	1	1	1	1	5				
marco 1		2	2	2	2	2	2				
marco 2			3	3	3	3	3				
marco 3				4	4	4	4				

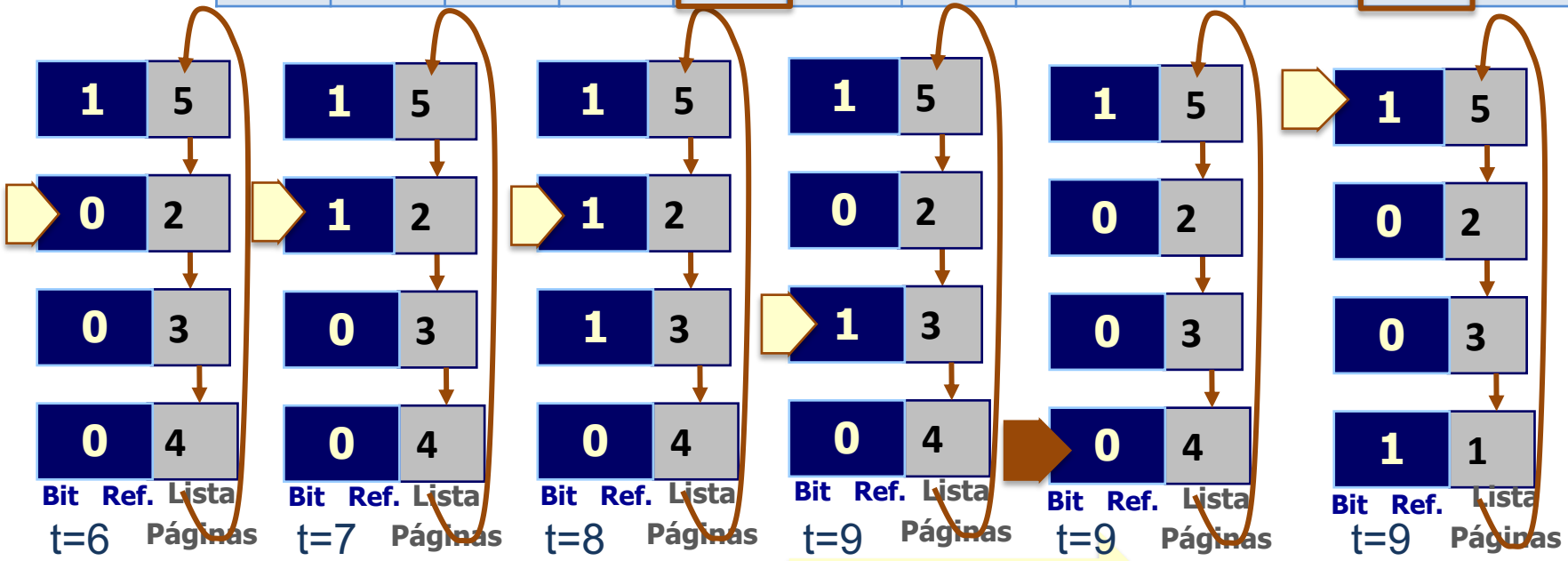


Buscamos víctima con segunda oportunidad

Algoritmo de 2ª oportunidad

Ejemplo: Memoria Principal de 4 marcos

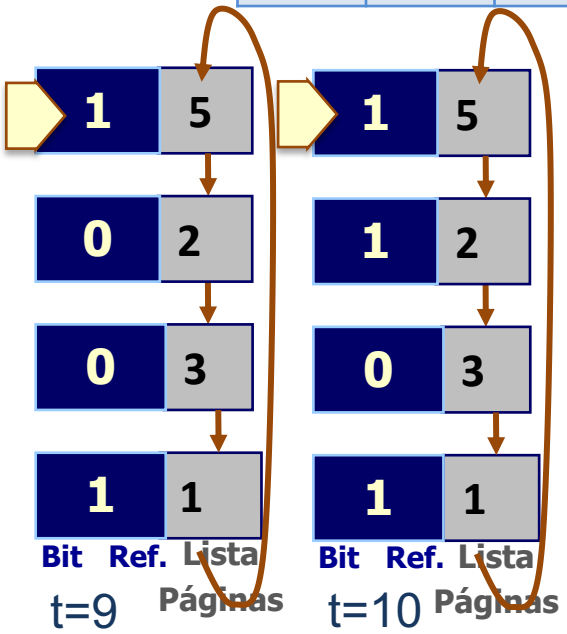
T	0	1	2	3	4	5	6	7	8	9	10
Serie de referencias	1	2	3	4	1	2	5	2	3	1	2
marco 0	1	1	1	1	1	1	5	5	5	5	
marco 1		2	2	2	2	2	2	2	2	2	
marco 2			3	3	3	3	3	3	3	3	
marco 3				4	4	4	4	4	4	1	



Algoritmo de 2ª oportunidad

Ejemplo: Memoria Principal de 4 marcos

T	0	1	2	3	4	5	6	7	8	9	10
Serie de referencias	1	2	3	4	1	2	5	2	3	1	2
marco 0	1	1	1	1	1	1	5	5	5	5	5
marco 1		2	2	2	2	2	2	2	2	2	2
marco 2			3	3	3	3	3	3	3	3	3
marco 3				4	4	4	4	4	4	1	1



Total 6 fallos de página (2 con reemplazo)

Contenido

- Algoritmo de Reemplazo de 2ª Oportunidad
- **Asignación de marcos**
- Hiperpaginación
- Reserva de Marcos

Asignación de marcos

- **Problema de asignación de marcos**
  - Lista de marcos libres:
    - La gestión de marcos requiere una **estructura de datos** donde se mantengan los **marcos libres**
  - Política de reparto de marcos entre procesos usuario y SO
    - Se dota al SO del número de marcos necesario para ejecutarse
    - Se dota a los procesos de un número mínimo inicial de marcos y el resto bajo demanda
    - El número mínimo de marcos que debe asignarse depende de la arquitectura del procesador (nivel de indirección en el juego de instrucciones)

Asignación de marcos

- Políticas de **reparto de marcos**
  - Sea un sistema con **m marcos** y **n procesos** $P_1 \ldots P_i \ldots P_n$ cuyos tamaños son $S_1 \ldots S_i \ldots S_n$ respectivamente
    - **Asignación equitativa**: A todos los procesos se le asignan $\mathbf{A_i}$ marcos por igual

$$A_i = m/n$$

    - **Asignación proporcional**: Al proceso P_i de tamaño S_i se le asignan $\mathbf{A_i}$ marcos calculados como

$$A_i = \frac{S_i * m}{\sum S_i}$$

    - **Asignación prioritaria**: se asignan más marcos a los procesos más prioritarios

Asignación de marcos

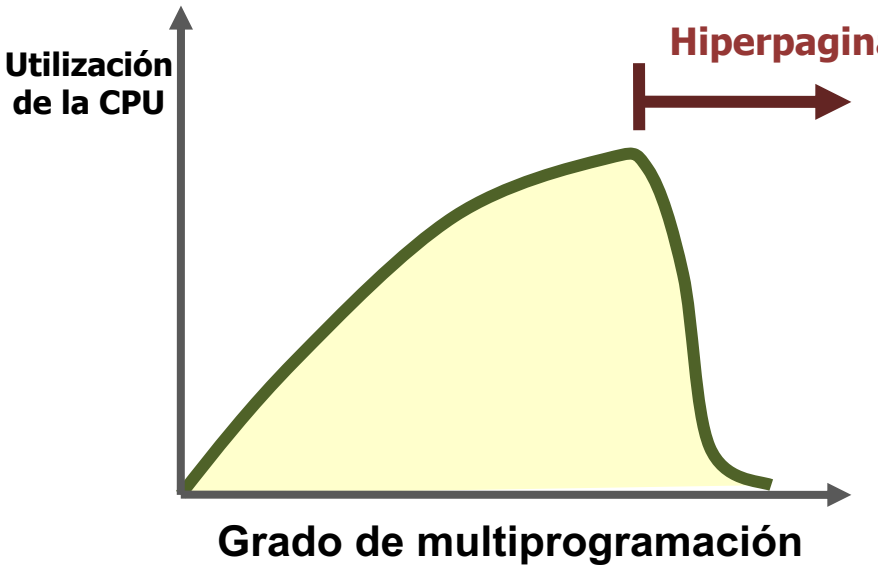
- **Ámbito de las políticas de reemplazo**
  - **Reemplazo local**: el sistema elige páginas víctimas únicamente entre aquellas que ocupan marcos pertenecientes al mismo proceso que genera el fallo de página
    - No se puede elegir como víctima páginas de otro proceso
    - El número total de marcos asignados a un proceso no varía
    - Su ejecución no se ve influenciada por la del resto de procesos
      - Ventajas: Tiempo de respuesta razonable
      - Inconveniente: Peor gestión global de memoria
  - **Reemplazo global**: El sistema elige páginas víctimas de entre todas las que tienen asignados marcos en memoria;
    - Puede elegir como víctima páginas de otro proceso
    - El número total de marcos asignados a un proceso puede variar
      - Ventajas: Gestión global de memoria optimizada
      - Inconveniente: Tiempo de respuesta elevado

Contenido

- Algoritmo de Reemplazo de 2ª Oportunidad
- Asignación de marcos
- **Hiperpaginación**
- Reserva de Marcos

Hiperpaginación (thrashing)

- Problema de **hiperpaginación**:
  - La memoria comienza a escasear y los procesos generan **muchos fallos de página** → pasan más tiempo haciendo E/S que cómputo
  - El SO ante el **bajo uso de la CPU** acepta más procesos para ejecución
  - Se agrava el problema al tener que repartir la misma memoria entre más procesos→ aumenta la tasa de fallos de página



¿Como resolver la hiperpaginación?

–Prevenir y anticiparse al problema

  –Utilizar un modelo de área activa

  –Control de la tasa de fallos

–Una vez detectada la hiperpaginación

  –Pasar procesos a disco utilizando planificador a medio plazo

Hiperpaginación (thrashing)

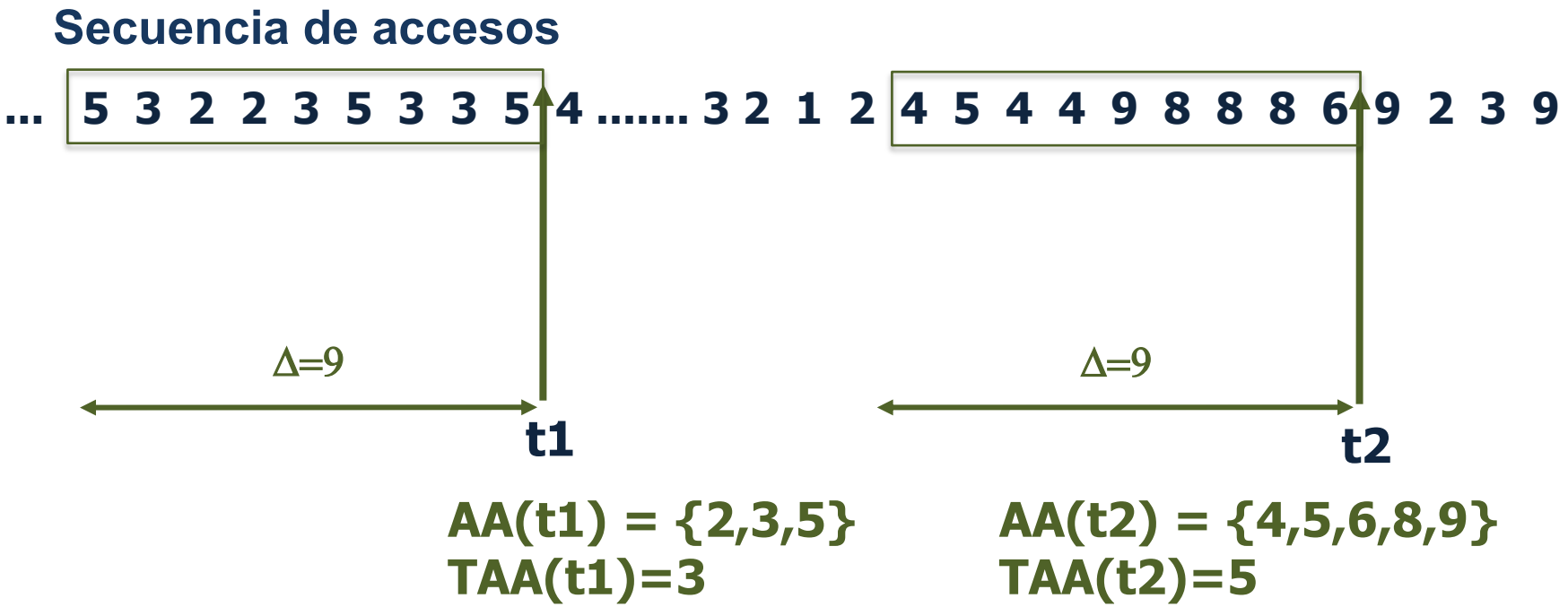
- Principio de **localidad de referencia**
  - Comportamiento de los programas según el cual, basándonos en el pasado reciente de accesos, se predice con una precisión razonable, qué instrucciones y datos se referenciarán en un futuro próximo, ya que estos no varían de forma abrupta sino gradualmente
  - Localidad:
    - Conjunto de páginas que un proceso utiliza conjuntamente.
    - Determinarla es costoso de aplicar (requiere muchos recursos)
  - Se produce hiperpaginación cuando

    tamaños de localidad > tamaño memoria total

Hiperpaginación (thrashing)

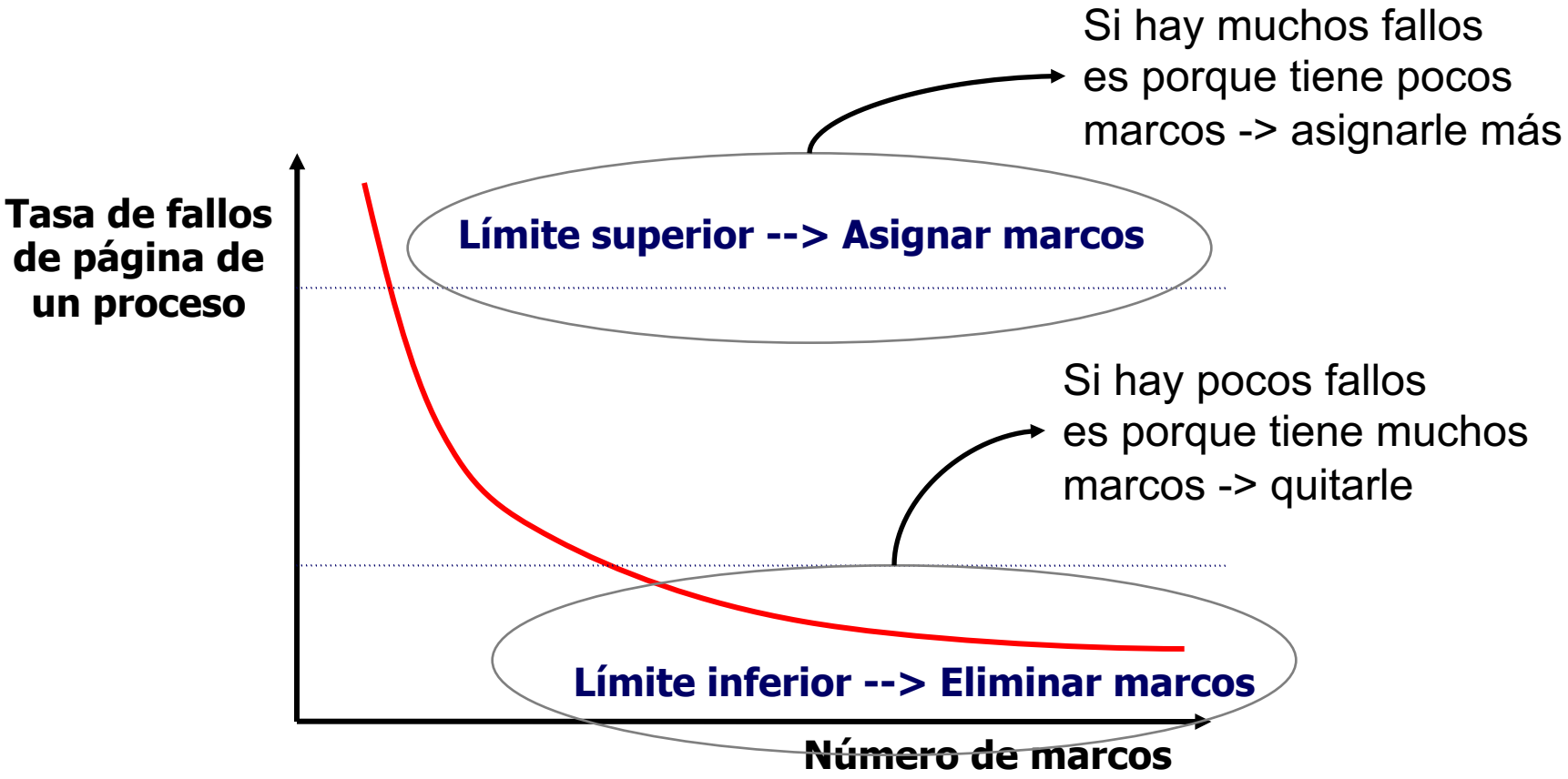
- Técnica preventiva: **Modelo del área activa**:
  - Asume el principio de **localidad de referencia**
  - Determinar el **número de páginas que un proceso debe tener en memoria simultáneamente** para obtener un buen rendimiento y evitar la hiperpaginación
  - **Área activa (AA) o "working set" (WS):**
    - Conjunto de páginas accedidas en las últimas referencias
    - Ventana de área activa
      - Número fijo Δ de referencias utilizadas para calcular el AA
      - AA formado por las páginas accedidas en las Δ últimas referencias
    - Tamaño de área activa: Número de páginas diferentes que conforman el AA_i del proceso P_i → TAA_i
    - En un sistema con **m marcos y n procesos** $P_1 \ldots P_i \ldots P_n$ se produce **hiperpaginación** cuando $\mathbf{\Sigma\ TAA_i > m}$

- **Modelo del área activa**
  - Ejemplo
    - Ventana de área activa Δ = 9
    - Cálculo de AA y TAA para un proceso P en los instantes t1 y t2



Hiperpaginación (thrashing)

- **Control de la tasa de fallos de página**
  - Técnica preventiva que analiza directamente la tasa de fallos para determinar si se está entrando en hiperpaginación



Contenido

- Algoritmo de Reemplazo de 2ª Oportunidad
- Asignación de marcos
- Hiperpaginación
- **Reserva de Marcos**

Reserva de marcos

- **Concepto**
  - **RESERVA DE MARCOS =** Conjunto de marcos que muchos SO modernos reservan de forma continuada, como almacén de marcos libres. Es un porcentaje de la memoria principal.
- **Objetivos**
  - **Reducir el tiempo invertido para servir un fallo de página**
    - Se intenta tener marcos libres disponibles
    - Se utiliza el algoritmo de reemplazo
      - Sólo cuando el nivel de marcos libres baje demasiado
      - Para buscar varias víctimas amortizando su uso
    - Realizar el "page out"
      - Se escriben varias páginas cada vez en el disco
  - Utilizar la reserva para **resolver el problema de hiperpaginación**

Reserva de marcos

- **Gestión de la reserva de marcos**
  - El S.O. garantiza que siempre habrá cierto número de **marcos libres**
  - Se fijan ciertos umbrales:
    - Número mínimo de marcos disponibles (Mmin)
    - Número recomendable de marcos disponibles (Mrec)

      Mrec >> Mmin
  - No es necesario utilizar algoritmos de reemplazo muy eficientes
    - Los primeros sistemas VMS utilizaban el algoritmo FIFO, pues su MMU no tenía bit de referencia.
    - Es normal utilizar un algoritmo de segunda oportunidad (Windows, UNIX SVR4, UNIX 4.4BSD, Linux, HP OpenVMS, ...).

- **Gestión de reserva de marcos: Proceso monitor**
  - Existe un proceso interno del sistema que periódicamente **monitoriza el número de marcos libres** (frame_free)
    - Si **frame_free > Mrec,** entonces no hay que hacer nada
    - Si **frame_free < Mmin** entonces:
      - Se escriben en disco algunos procesos completos (swap out) hasta llegar a **Mrec** marcos libres
      - Se escogen como víctimas aquellos procesos que lleven más tiempo sin hacer nada (suspendidos). Por ej., servidores que no hayan recibido ninguna petición.
    - Si **Mmin <= frame_free <= Mrec** entonces:
      - Se buscan procesos con un elevado número de marcos asignados y con una baja tasa de fallos de página. Se aplica el algoritmo de reemplazo para “robarles” algunos marcos.
      - Se seleccionan varias víctimas en cada ejecución del proceso interno. El número concreto depende de cada sistema operativo.

- **Gestión de la reserva en la hiperpaginación**
  - Si se produce hiperpaginación, el número de marcos libres baja rápidamente
  - El proceso monitor del nivel de la reserva lo detecta cuando
    - Entre dos activaciones sucesivas de dicho proceso el nivel de **frame_free** baja más de lo conveniente.
  - Soluciones:
    - Expulsar (swap out) procesos completos hasta llegar a un nivel recomendable (Mrec).
      - Gestión utilizada en OpenVMS y Windows.
    - Otros monitores liberan un número constante de marcos en cada activación (si el nivel está por debajo de Mrec).
      - Entonces el monitor pasa a activarse con mayor frecuencia.
      - Gestión empleada en UNIX SVR4.

- **Contenido de los marcos de la reserva**
  - Si el S.O. elige una víctima su marco no se utiliza de inmediato para otra página el marco PASA A LA RESERVA
    - Si las **páginas víctimas vuelven a ser referenciadas** por el proceso rápidamente
      - Hay una probabilidad alta de que todavía estén disponibles en la reserva y se puedan reasignar sin necesidad de volverlas a leer del disco
      - El S.O. recuerda cuál es el contenido de cada marco de su reserva
    - Si los marcos incluidos en la reserva corresponden a **páginas modificadas**
      - No son considerados para atender fallos de página de forma inmediata. Ya que debe escribirse su contenido en un fichero o partición de intercambio
        - » Cuando se supera cierto mínimo, todas esas páginas son escritas en el fichero o partición de intercambio, de una sola vez
        - » Se evita hacer el "page out" por cada página, minimizando el número global de escrituras necesarias.
      - Una vez escritas las páginas sus marcos son libres y pueden utilizarse para servir fallos de página