



Practica 4

1. Memoria Virtual

a. Describa qué beneficios introduce este esquema de administración de la memoria.

Beneficios de la memoria virtual:

1. **Uso eficiente de la memoria física:** Se intenta alojar en memoria la mayor cantidad de paginas necesarias posibles
2. **Aprovechamiento del modelo de localidad:** Se aprovecha del comportamiento natural de los procesos, donde las referencias a datos y programas tienden a agruparse
3. **Capacidad de recuperacion y continuidad:** Permite al programa continuar su ejecucion a pesar de que una pagina no este cargada, seleccionando una victima y reemplazandola con la nueva
4. **Gestion dinamica de recursos:** La asignacion de marcos puede ajustarse a cada proceso

b. ¿En que se debe apoyar el Kernel para su implementación?

Para lograr esto, el kernel se apoya en 4 herramientas

- **TLB:** Necesita guardar en cache las ubicaciones que ya conoce para no perder tiempo buscando en tablas gigantes
- **Marcas:** Necesita que el HW le avise que paginas se estan usando, por ejemplo: el bit de referencia (R)
- **Page Faults:** El sistema depende de la capacidad de generar y manejar fallos de pagina, cada vez que se necesita cargar una pagina en un marco y no esta en memoria fisica

- **Frames y Swaps:** El SO debe reservar marcos específicos para descarga asincrónica. Si tiene que guardar una página modificada en disco, la deja en uno de esos marcos y sigue atendiendo otros procesos sin quedarse colgada

c. Al implementar esta técnica utilizando paginación por demanda, las tablas de páginas de un proceso deben contar con información adicional además del marco donde se encuentra la página. ¿Cuál es esta información? ¿Por qué es necesaria?

Además de saber donde está la página, la tabla necesita 3 etiquetas:

- **Bit de Validez:** *¿Está en RAM?*, Fundamental para saber si el proceso puede seguir o frenar y traer la página desde disco.
- **Bit de Referencia:** *¿Se usó recién?*, Sirve para que el algoritmo de reemplazo sepa a quien sacar
- **Bit de Modificación:** *¿Se modificó?*, Avisa al sistema para saber si la pag se modificó desde que se cargó, si fue así debe guardarla en disco antes de borrarla

2. Fallos de Página (Page Faults)

a. ¿Cuándo se producen?

Un fallo de página se produce cada vez que es necesario asignar una página en un marco de memoria física. Ocurre cuando un proceso intenta acceder a una página que no se encuentra cargada actualmente en memoria real.

b. ¿Quién es responsable de detectar un fallo de página?

c. Describa las acciones que emprende el Kernel cuando se produce un fallo de página.

1. **Hacer lugar:** Si la memoria esta llena, elige una "pagina victima" para sacar usando un algoritmo
2. **Guardar los cambios:** Si la "pagina victima" fue modificada, se guardan los cambios en disco antes de borrarla; sino, se elimina directamente
3. **Traer la nueva:** Pide al disco la pagina que falta, este pedido tiene prioridad absoluta y se atiende antes que otros
4. **Reinicia:** Actualiza la tabla y le dice al proceso que intente ejecutar la instruccion de nuevo

3.

Suponga que la tabla de páginas para un proceso que se está ejecutando es la que se muestra a continuación:

Pagina	Bit V	Bit R	Bit M	Marco
0	1	1	0	4
1	1	1	1	7
2	0	0	0	-
3	1	0	0	2
4	0	0	0	-
5	1	0	1	0

Asumiendo que:

- El tamaño de la pagina es de 512 bytes.
- Cada direccion de memoria referencia 1 byte.
- Los marcos se encuentran contiguos y en orden en memoria (0, 1, 2...) a partir de la direccion fisica 0

? ¿Qué dirección física, si existe, correspondería a cada una de las siguientes direcciones virtuales? (No gestionar ningún fallo de página en caso de producirse)

a. 1052

$$p = (1052/512) = 2, d = (1052)MOD(512) = 28$$

Fallo de pagina, el bit V es 0

b. 2221

$$p = (2221/512) = 4, d = (2221)MOD(512) = 173$$

Fallo de pagina, el bit V es 0

c. 5499

$$p = (5499/512) = 10, d = (5499)MOD(512) = 379$$

Corresponde a una direccion ilegal

d. 3101

$$p = (3101/512) = 6, d = (3101)MOD(512) = 29$$

Corresponde a una direccion ilegal

4. Analice cómo impacta el tamaño de una página (pequeña o grande) en paginación por demanda.

Paginas grandes

Lo bueno

- Mas eficientes para el HW
- Menos interrupciones (menos PF)

Lo malo

- Se desperdicia memoria

Paginas chicas

Lo bueno

- Mucha precision (mejor ajuste)
- Se aprovecha cada byte de memoria

Lo malo

- El sistema pasa mucho tiempo detenido atendiendo PF

5. Asignacion Fija vs Asignacion Dinamica

a. Describa cómo trabajan estas 2 políticas.

Asignacion Fija:

El kernel decide de entrada cuantos marcos le tocan a cada proceso y ese numero no cambia durante la ejecucion. Hay dos formas de hacer este reparto

- **Reparto equitativo:** Se divide la memoria total por la cantidad de procesos y a todos le toca la misma cantidad
- **Reparto proporcional:** Es mas justo, se asignan marcos segun el tamaño del proceso

Asignacion Dinamica:

El kernel se adapta a las necesidades del momento, los procesos reciben los marcos que necesitan

- Se basa en **Working Set:** El sistema mira que paginas uso el proceso recientemente y le asigna los marcos necesarios para tener esas paginas cargadas
- Si la suma de los marcos que piden todos los procesos supera la memoria, se produce **thrashing**. Esta asignacion intenta evitar eso ajustando el tamaño del working set dinamicamente


b. Dada la siguiente tabla de paginas y teniendo 40 frames en memoria principal, cuantos marcos le corresponden a cada uno

Proceso	Total de Paginas Requeridas
1	15
2	20

3	20
4	8

Reparto equitativo:

- $M = 40$ (Marcos Disponibles)
- $N = 4$ (Numero de Procesos)

 $M_p = \frac{M}{N} = 10$

Podemos observar que solo el proceso 4 queda con Marcos \geq Paginas, el resto no le alcanza

Reparto Proporcional:



Formula:

$\frac{VP \cdot m}{VT}$ Siendo **VP** necesidad del proceso, **m** la cantidad de marcos totales y **VT** la suma de todas las necesidades

$$VT = 15 + 20 + 20 + 8 = 63$$

$$m = 40$$

Proceso 1:

$$\frac{15 \cdot 40}{63} = 9$$

Proceso 2:

$$\frac{20 \cdot 40}{63} = 13$$

Proceso 3:

$$\frac{20 \cdot 40}{63} = 13$$

Proceso 4:

$$\frac{8 \cdot 40}{63} = 5$$

Proceso	Reparto Equitativo	Reparto Proporcional
1	10	9
2	10	13

Proceso	Reparto Equitativo	Reparto Proporcional
3	10	13
4	10	5

c. ¿Cual de los dos repartos usados en b) resulto mas eficiente?

A mi parecer el reparto proporcional llevara a una menor cantidad de fallos de pagina en comparacion al reparto equitativo.

6. Reemplazo de paginas

a. Clasifique estos algoritmos de malo a bueno de acuerdo a la tasa de fallos de página que se obtienen al utilizarlos.

FIFO

Es el mas simple y el menos eficiente, trata a los marcos como una cola circular y reemplaza la pagina mas vieja que entro en memoria. No tiene en cuenta localidad ni frecuencia de uso

Segunda Chance

Optimizacion del FIFO. Reduce la tasa de fallos comparado con FIFO ya que evita expulsar paginas que son referenciadas activamente

LRU

Genera una tasa de fallos baja y aceptable pero tiene una contra y es el overhead, osea, el costo de guardar el instante de ultima referencia.

OPT

El algoritmo perfecto, con la tasa de fallos minima posible. Es imposible de implementar pero a nivel teorico es el algoritmo mas eficiente que se puede hacer

b. Analice su funcionamiento. ¿Cómo se podrían implementar?

FIFO

Se podria implementar con una cola circular

Segunda Chance

Se podría implementar con una cola circular y un bit que indique si el mismo ha sido referenciado

LRU

Se podría implementar guardando para cada página cuando fue accedida por última vez y calculando entre todas la más lejana

OPT

Imposible de implementar, requiere conocer el futuro.

c. Sabemos que la página a ser reemplazada puede estar modificada. ¿Cómo detecta el Kernel esta situación? ¿Qué acciones adicionales deben realizarse cuando se encuentra ante esta situación?

Cada página lleva un bit de modificación, si el mismo se encuentra en 1 (la página fue modificada), el kernel antes de sacarla de memoria debe guardar su estado en disco; en caso de que el mismo esté en 0 se elimina directamente

7. Alcance del reemplazo

a. Describa como trabaja reemplazo global y local

Reemplazo local:

El kernel se limita a seleccionar una página propia del proceso que generó el fallo. El proceso resuelve su falta de espacio desalojando una de sus páginas sin afectar al resto de procesos

Reemplazo global:

El kernel puede elegir como víctima una página perteneciente a cualquier proceso del sistema. O sea, el kernel puede robarle un marco a otro proceso

b. ¿Es posible utilizar la política de "Asignación Fija" de marcos junto con la política de "Reemplazo Global"? Justifique.

Genera una contradiccion ya que la asignacion fija reparte equitativamente marcos a todos los procesos, que el kernel pueda robarle a otro proceso un marco dejaria de ser equitativo ya que los marcos de un proceso disminuirian y los de otro aumentarían

8. Considere la siguiente secuencia de referencias a paginas

👉 1, 2, 15, 4, 6, 2, 1, 5, 6, 10, 4, 6, 7, 9, 1, 6, 12, 11, 12, 2, 3, 1, 8, 1, 13, 14, 15, 3, 8

FIFO

Marco / Pagina	1	2	15	4	6	2	1	5	6	10	4	6	7	9	1	6	12	11	12	2	3	1	8	1	13	14	15	3
F1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	6	6	6	6	6	6	1	1	1	1	1	1	3
F2		2	2	2	2	2	2	2	10	10	10	10	10	10	10	12	12	12	12	12	12	8	8	8	8	8	8	
F3			15	15	15	15	15	15	15	15	15	15	7	7	7	7	11	11	11	11	11	11	13	13	13	13	13	
F4				4	4	4	4	4	4	4	4	4	9	9	9	9	9	2	2	2	2	2	14	14	14	14		
F5					6	6	6	6	6	6	6	6	6	1	1	1	1	1	3	3	3	3	15	15	15			
PF?		X	X	X	X		X	X		X		X	X	X	X		X	X		X	X	X	X		X	X	X	X
FIFO																										# X =	21	

Segunda Chance

Marco / Pagina	1	2	15	4	6	2	1	5	6	10	4	6	7	9	1	6	12	11	12	2	3	1	8	1	13	14	15	3
F1	1	1	1	1	1	1	1*	1	1	1	4	4	4	4	4	4	12	12	12*	12*	12*	12*	12	12	12	12	15	15
F2		2	2	2	2	2*	2*	2	2	2	2	2	7	7	7	7	7	11	11	11	11	11	8	8	8	8	3	
F3			15	15	15	15	4	5	5	5	5	5	5	9	9	9	9	9	9	2	2	2	2	2	13	13	13	
F4				4	4	4	4	4	4	10	10	10	10	10	1	1	1	1	1	1	3	3	3	3	3	14	14	14
F5					6	6	6	6	6*	6*	6	6*	6*	6*	6*	6*	6	6	6	6	6	1	1	1*	1*	1*	1	1
PF?		X	X	X	X		X		X	X		X	X	X	X	X	X	X		X	X	X	X		X	X	X	X
Seq. Chance																										# X =	20	

LRU

Marco / Pagina	1	2	15	4	6	2	1	5	6	10	4	6	7	9	1	6	12	11	12	2	3	1	8	1	13	14	15	3
F1	1	1	1	1	1	1	1	1	1	1	1	6	6	6	6	6	6	6	6	6	6	1	1	1	1	1	1	1
F2		2	2	2	2	2	2	2	2	2	4	4	4	4	4	4	12	12	12	12	12	12	12	12	13	13	13	12
F3			15	15	15	15	15	5	5	5	5	5	7	7	7	7	7	11	11	11	11	11	8	8	8	8	8	3
F4				4	4	4	4	4	4	10	10	10	10	10	1	1	1	1	1	1	3	3	3	3	3	3	15	15
F5					6	6	6	6	6	6	5	5	5	6	9	9	9	9	9	9	2	2	2	2	2	2	14	14
PF?	X	X	X	X	X		X		X	X	X	X	X	X		X	X		X	X	X	X		X	X	X	X	
LRU																										# X =	22	

OPT

Marco / Pagina	1	2	15	4	6	2	1	5	6	10	4	6	7	9	1	6	12	11	12	2	3	1	8	1	13	14	15	3
F1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	13	13	13	13
F2		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	8	8	8	8	8	8
F3			15	15	15	15	15	5	5	10	10	10	10	9	9	9	9	9	9	9	9	9	9	9	9	14	14	14
F4				4	4	4	4	4	4	4	4	4	7	7	7	7	7	7	7	7	3	3	3	3	3	3	3	3
F5					6	6	6	6	6	6	6	6	6	6	6	6	6	11	11	11	11	11	11	11	11	11	15	15
PF?	X	X	X	X	X			X		X			X	X			X	X	X		X		X		X	X	X	
OPT																										# X =	17	