Caso #2
ISIS 1311 TIC

Autores:

Jeronimo Vasquez Ponce 202223824
Kevin Efren Cruz Holguin

Semestre 2025–20

Descripción del algoritmo (Opción 1)

La Opción 1 tiene como objetivo generar los **archivos de referencias de página** que representan los accesos a memoria virtual de cada proceso que suma matrices.

Cada proceso definido en el archivo de configuración (`config.txt`) recibe como entrada el tamaño de las matrices a sumar. Aunque en la configuración se hable de una sola matriz, en la práctica cada proceso trabaja siempre con **tres matrices**:

- **M1**: primera matriz de entrada.
- **M2**: segunda matriz de entrada.
- **M3**: matriz de salida, donde se almacena el resultado de la suma.

Para cada celda (i, j) de la matriz, el algoritmo produce tres accesos de memoria:

1. Lectura de $M1[i][j]$.
2. Lectura de $M2[i][j]$.
3. Escritura en $M3[i][j]$.

Las matrices se almacenan en memoria virtual en **orden secuencial por filas (row-major order)**, una detrás de otra (primero M1, luego M2 y finalmente M3). Cada entero ocupa 4 bytes. Con base en el tamaño de página (TP), el tamaño de la matriz (NF y NC) y la posición (i, j), se calcula:

- El número de página virtual en el que cae la dirección.
- El desplazamiento dentro de esa página.
- El tipo de acceso: lectura (`r`) o escritura (`w`).

El resultado de esta fase son los archivos `procX.txt`, donde cada línea corresponde a una referencia generada, siguiendo el formato:

$$M\# : [i - j],\ \text{PáginaVirtual},\ \text{Desplazamiento},\ \text{Acción}$$

Por ejemplo:

$$M1 : [0 - 0], 0, 0, r$$

indica un acceso de lectura a la matriz 1, posición $(0, 0)$, almacenada en la página virtual 0, desplazamiento 0.

—

Estructuras de datos usadas (Opción 2)

La Opción 2 se encarga de **simular la ejecución del sistema de memoria virtual** utilizando como insumo los archivos de referencias generados en la Opción 1.

Para ello, se emplean las siguientes estructuras de datos principales:

- **Tabla de páginas**: un mapa que asocia cada página virtual con una entrada que contiene:
  - Si la página está cargada en memoria (`bit de validez`).
  - El marco físico en el que reside la página.
  - El contador de último uso, empleado por la política de reemplazo.
- **Cola de procesos**: mantiene los procesos en ejecución, turnándose en un esquema de *round-robin*.
- **Contador global**: se incrementa en cada acceso y permite implementar la política **LRU (Least Recently Used)**, identificando cuál página fue usada hace más tiempo.

Dinámica de actualización:

- Cuando ocurre un **hit** (la página ya está en memoria), se actualiza el campo de último uso en la tabla de páginas y se incrementa el contador de referencias procesadas.
- Cuando ocurre un **fallo de página**:
  1. Si el proceso aún tiene marcos libres asignados, la página se carga directamente en uno de ellos y se incrementa el contador de accesos a SWAP en 1.
  2. Si no hay marcos libres, se aplica **LRU**: se selecciona la página menos recientemente usada, se expulsa de la memoria y se carga la nueva página en ese marco. En este caso, el acceso a SWAP se incrementa en 2 (una escritura y una lectura).

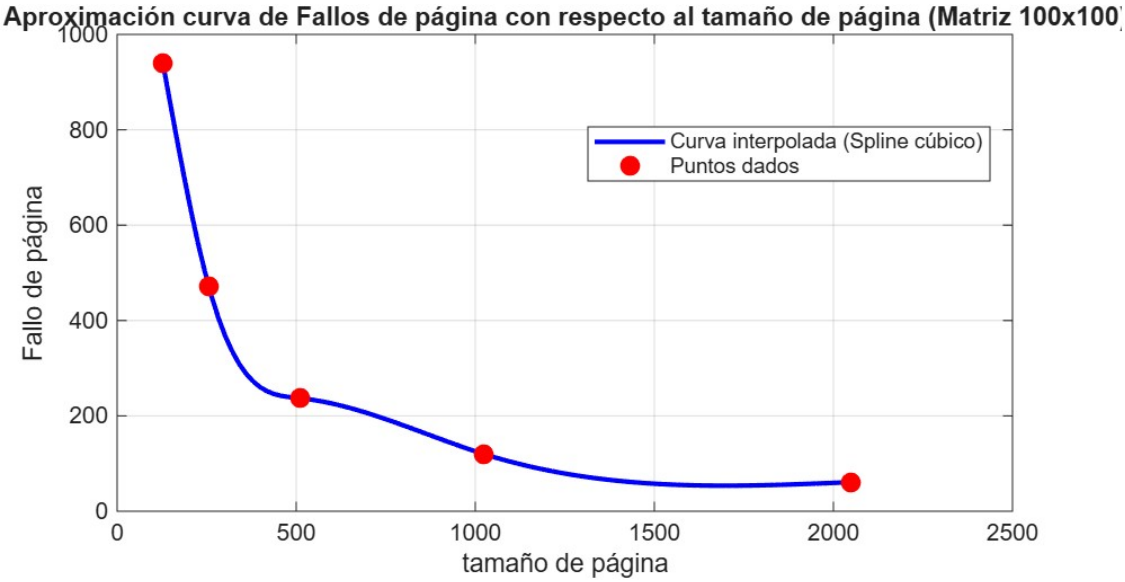
La simulación produce al final estadísticas por proceso: número total de referencias, número de fallos, hits, accesos a SWAP, tasa de fallos y tasa de éxito. Estos datos permiten analizar cómo influyen el tamaño de página, el número de marcos y el tamaño de la matriz en el desempeño de la memoria virtual.

Resultados experimentales

Se realizaron experimentos variando el tamaño de página y de matrices.

Caso base (matriz 100x100, variando TP)

La siguiente figura muestra el comportamiento del número de fallos de página con respecto al tamaño de página para una matriz de 100×100:



Análisis gráfico: Matriz 100×100

También se evaluó de forma independiente el comportamiento de la matriz de 100×100 (Proceso 0), variando el tamaño de página entre 128 y 2048 bytes.

TP (bytes)	#Referencias	Fallos	Tasa de fallos
128	30000	939	0,0313
256	30000	471	0,0157
512	30000	237	0,0079
1024	30000	120	0,0040
2048	30000	61	0,0020

Interpretación: En el caso de la matriz 100×100, se confirma que el número de fallos de página disminuye rápidamente al aumentar el tamaño de página.

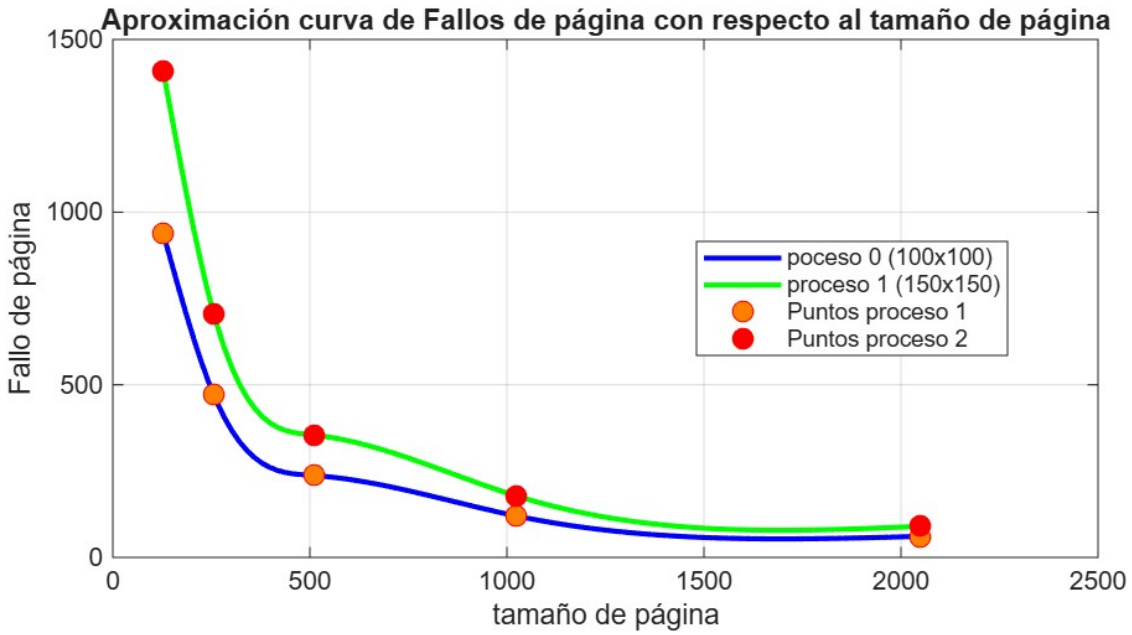
- Con TP=128, se producen casi mil fallos, lo que equivale a una tasa de 3,13%.
- Al doblar el TP a 256, los fallos se reducen a la mitad (471).
- Con TP=512 y superiores, los fallos siguen disminuyendo hasta llegar a solo 61 con TP=2048, lo que representa una tasa de 0,2%.

Esto demuestra que el acceso secuencial por filas en la suma de matrices genera una **localidad espacial muy alta**, ya que al incrementar el tamaño de página, los datos necesarios permanecen más tiempo en memoria y se aprovecha mejor cada carga de página. El resultado es una curva de fallos decreciente casi exponencial con respecto al tamaño de página, confirmando la eficiencia de este patrón de acceso.

Análisis gráfico Matriz 100 x 100, 150 x 100

Además de los escenarios básicos, se corrió el programa con dos procesos:

- Proceso 0: matrices de 100×100.
- Proceso 1: matrices de 150×100.



Se variaron los tamaños de página (**TP**) entre 128 y 2048 bytes, y se midió el número de fallos de página y la tasa de fallos.

TP (bytes)	Proceso	#Referencias	Fallos	Tasa de fallos
128	0	30000	939	0,0313
	1	45000	1409	0,0313
256	0	30000	471	0,0157
	1	45000	706	0,0157
512	0	30000	237	0,0079
	1	45000	354	0,0079
1024	0	30000	120	0,0040
	1	45000	178	0,0040
2048	0	30000	61	0,0020
	1	45000	90	0,0020

Interpretación de la gráfica: La tendencia es clara: a medida que aumenta el tamaño de página (**TP**), el número de fallos de página disminuye drásticamente. Con TP=128, la tasa de fallos es cercana al 3%, mientras que con TP=2048 cae a apenas 0,2%.

Esto ocurre porque con páginas más grandes, cada carga trae a memoria una mayor cantidad de elementos contiguos de la matriz. Dado que la suma de matrices accede a posiciones de forma secuencial por filas, el programa aprovecha la **localidad espacial**.

En otras palabras:

- Con TP pequeño, los accesos se dispersan más entre páginas, generando muchos fallos.
- Con TP grande, cada página contiene una franja amplia de la matriz, reduciendo la necesidad de traer nuevas páginas desde disco.
- El comportamiento es similar en ambos procesos (100x100 y 150x100), mostrando que la tendencia se mantiene independiente del tamaño de matriz.

En la gráfica de resultados (incluida en este informe) se observa cómo la curva de fallos decrece de forma casi exponencial con el incremento del tamaño de página, validando las propiedades de localidad espacial del problema de sumar matrices.

Pruebas adicionales

Además, se realizaron 4 escenarios adicionales:

- **Prueba 1:** Matriz 50×50, TP=128, marcos=4. Referencias: 7500. Fallos: 237. Tasa de fallos: 3,16%. Tasa de éxito: 96,84%. SWAP=470.
- **Prueba 2:** Matriz 200×200, TP=256, marcos=8. Referencias: 120000. Fallos: 1875. Tasa de fallos: 1,56%. Tasa de éxito: 98,44%. SWAP=3742.
- **Prueba 3:** Matriz 150×100, TP=512, marcos=16. Referencias: 45000. Fallos: 354. Tasa de fallos: 0,79%. Tasa de éxito: 99,21%. SWAP=692.
- **Prueba 4:** Matriz 300×300, TP=1024, marcos=32. Referencias: 270000. Fallos: 1057. Tasa de fallos: 0,39%. Tasa de éxito: 99,61%. SWAP=2082.

Pruebas de los anexos

Con las configuraciones del anexo (TP=128, procesos de 4×4 y 8×8):

- Proceso 0: Referencias: 48. Fallos: 2. SWAP=2. Tasa de fallos: 4,17%.
- Proceso 1: Referencias: 192. Fallos: 6. SWAP=8. Tasa de fallos: 3,13%.

Interpretación de resultados

Los resultados muestran que:

- Al aumentar el tamaño de página, los fallos de página disminuyen significativamente.
- Al aumentar el número de marcos disponibles, el desempeño mejora (menos fallos y menos accesos a SWAP).
- Para matrices más grandes, aunque el número absoluto de fallos crece, la **tasa de fallos relativa disminuye**, lo que significa mejor aprovechamiento de la localidad espacial.

Esto corresponde a lo esperado: mayor tamaño de página y mayor número de marcos permiten que más datos se mantengan en memoria, reduciendo la frecuencia de reemplazos.

Tiempo de respuesta y accesos a SWAP

El tiempo de respuesta de un proceso se ve directamente afectado por la cantidad de accesos a **SWAP**. Cada vez que hay un fallo de página se debe traer la página desde memoria secundaria, lo que implica un retardo significativo. Más accesos a SWAP $\Rightarrow$ mayor latencia de respuesta del proceso.

Localidad del problema de sumar matrices

El problema de sumar matrices tiene una **localidad espacial alta**, ya que los accesos se realizan de manera secuencial por filas. Cada página cargada se aprovecha al máximo antes de avanzar a la siguiente. Esto se refleja en la disminución de la tasa de fallos a medida que aumentan el tamaño de página y el número de marcos. En consecuencia, sumar matrices es un caso de acceso con buena localidad.