Universidad de Los Andes, Bogotá, Colombia.
Semestre 2021-20
Infraestructura Computacional (ISIS 2203)
Caso 2

Estudiante: Daniel S. Londoño B. - 201821363
Login: ds.londono@uniandes.edu.co

---

1. Explicación de las Estructuras de Datos Utilizadas

Las estructuras de datos que se usaron para la realización del caso fueron dos particularmente. En primera instancia, se usó una Cola (`Queue`) para modelar la tabla de paginas en memoria real, la cual por simplicidad en esta arquitectura de solución almacena objetos de tipo Pagina. Para poder recorrer de forma óptima está estructura de datos se hizo uso del Iterator (`java.util.Iterator`) debido a su eficiencia. En segunda instancia, al momento de clasificar las páginas por clases (de acuerdo a las especificaciones del algoritmo NRU), se hizo uso de diversos ArrayList únicamente con la finalidad de agrupar temporalmente a un conjunto de páginas por clases teniendo en cuenta sus bits *R* y *M*.

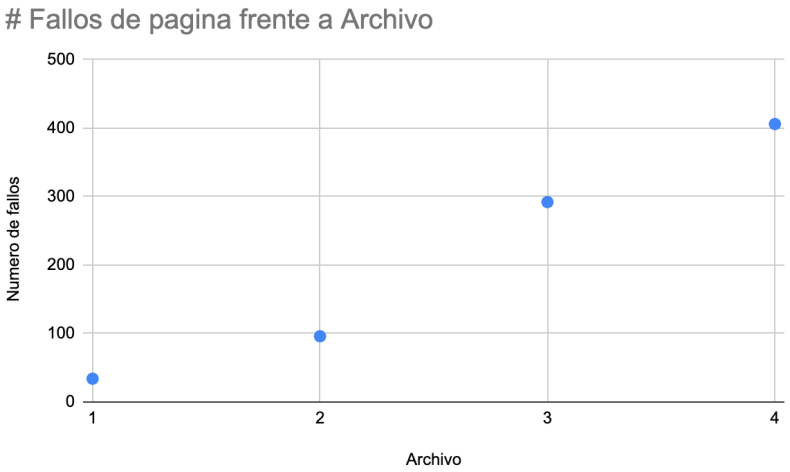
2. Esquema de Sincronización

El esquema de sincronización empleado en este caso se encuentra exclusivamente en el recurso compartido por los dos threads presentes. En este caso, es la tabla de páginas en memoria real. En la clase `RAM.java` propuesta en la solución, se evidencia el uso de `synchronized` en el método que carga marcos de página, y en el método que 'limpia' (i.e que pone en 0) el bit *R* de todas las páginas de acuerdo al algoritmo NRU. La razón por la cual estos métodos tienen que estar sincronizados es porque de no estarlo, ambos threads podrían alterar de forma significativa el estado de cada marco de página al tiempo (especificamente sus bits de referencia), y así ocasionar problemas con el proceso que hipotéticamente se estaría ejecutando.

3. Datos Recopilados

La gráfica acontinuación muestra el número de fallos de pagina para cada archivo de prueba propuesto para el caso.

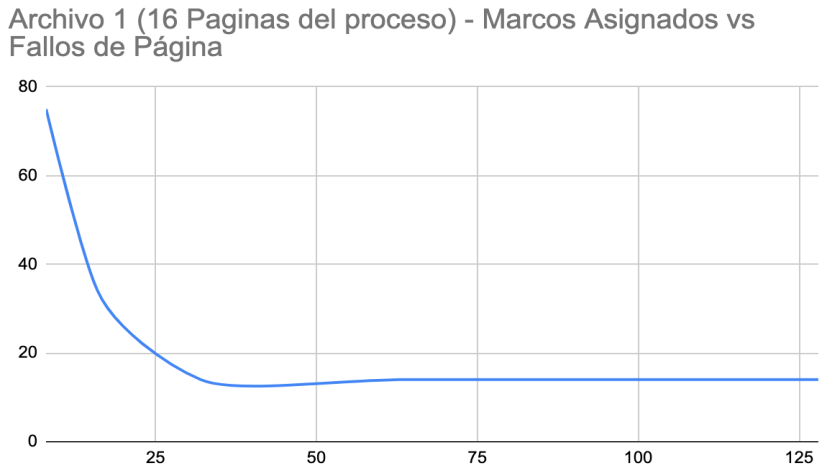
- Archivo 1: ./data/referencias8_16_75.txt
- Archivo 2: ./data/referencias8_32_75.txt
- Archivo 3: ./data/referencias8_64_75.txt
- Archivo 4: ./data/referencias8_128_75.txt



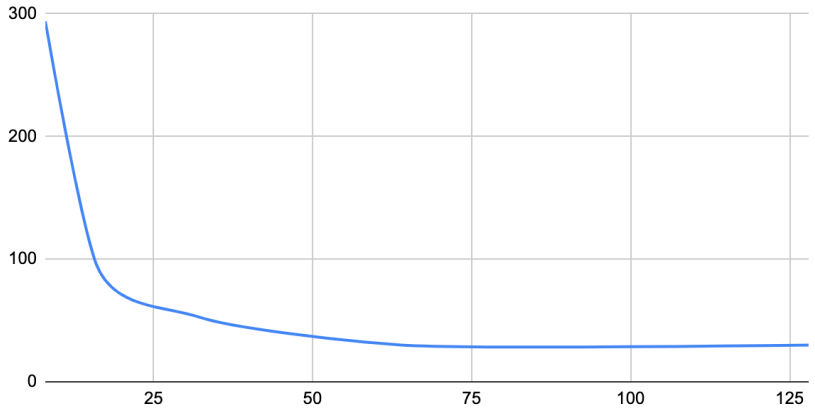
El número de fallos de página son 34, 96, 292 y 406, para el archivo 1, 2, 3 y 4, respectivamente. El número de fallas contabilizadas, para un mismo archivo de configuración puede variar entre ejecuciones del programa debido a que este corre con threads. Los threads en este programa manipulan los bits de referencia de los marcos de página, y, sabiendo que el sistema operativo gestiona estos threads como él lo considere conveniente, pueden haber ocasiones donde un thread empiece antes o después que otro (por fracciones muy pequeñas de tiempo), modificando bits más lento o más rápido que el otro, cambiando así el número de fallos de página dada la funcionalidad del algoritmo NRU.

4. Comportamiento del Sistema

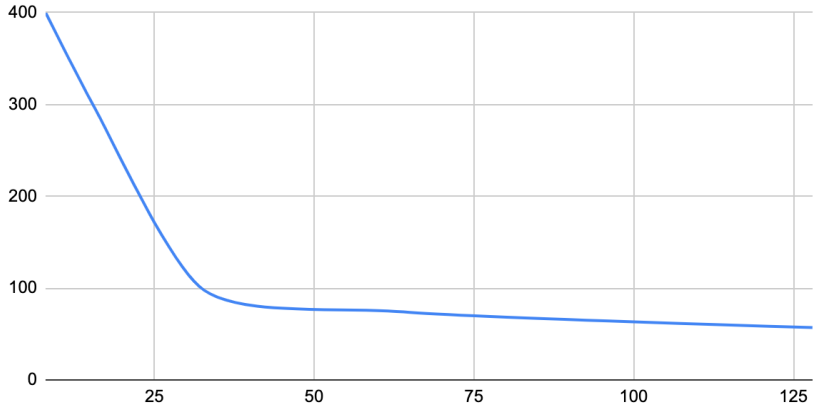
Para poder analizar de forma eficiente el comportamiento del sistema, para cada proceso se ejecutó el programa con marcos de página variables (de 4 a 128 siguiendo potenciación 2^n) y se comparó con el número de fallos de página. Esta información fue tabulada de forma que fuera posible identificar qué tantos fallos de página ocurren dependiendo del número de marcos de página asignados en memoria real, para así verificar el correcto funcionamiento del programa. Los resultados se ven en los gráficos acontinuación.



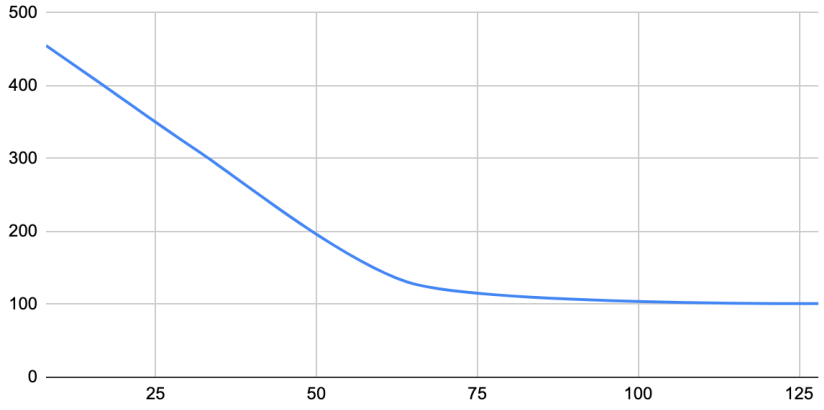
Archivo 2 (32 Paginas del proceso) - Marcos Asignados vs Fallos de Página



Archivo 3 (64 Paginas del proceso) - Marcos Asignados vs Fallos de Página



Archivo 4 (128 Paginas del proceso) - Marcos Asignados vs Fallos de Página



5. Interpretación de los Resultados

De los anteriores gráficos podemos evidenciar que los resultados tienen sentido, ya que a menor número de marcos de página asignados en memoria real, el proceso va a generar más fallos de página simplemente porque habrá menos espacio para cargar sus páginas en la secuencia dada. El número de fallos de página es inversamente proporcional al número de marcos de página asignados en memoria real, y esto se evidencia en las gráficas, ya que todas tienen un comportamiento exponencial (e^{-x}). Es igualmente posible evidenciar en las gráficas el cumplimiento del programa del concepto de espacio de trabajo. Este afirma que "*para todo proceso existe un número x, conocido espacio de trabajo, tal que si a ese proceso se le asignan menos marcos de página, el número de fallas de página crece considerablemente*" (Notas de Clase).