**Informe Caso 2**

1. Descripción del algoritmo usado para generar las referencias de página (modo uno).

El algoritmo comienza tomando la imagen que proporciona el usuario para leer la longitud del mensaje asociado. Con base en la longitud del mensaje, se calcula el número de referencias necesarias. La fórmula es: longitud del mensaje por 17 más 16, donde 17 proviene del hecho de que, por cada carácter, se inicializa una posición en el vector, se realizan 8 lecturas y 8 escrituras (16 en total). Se suma 16 ya que se usan 16 bytes al principio para almacenar la longitud del mensaje (que solo se referencian en lectura).

Luego, se solicita al usuario el tamaño de la página. Con esta información, se calcula el número total de páginas necesarias para almacenar tanto la imagen como el mensaje. Para la imagen, se multiplica el alto por el ancho por 3 (porque cada píxel tiene tres componentes: red, green, blue, que ocupan un byte), y se divide por el tamaño de la página. Además, se añade el espacio necesario para el vector del mensaje, que es la longitud del mensaje dividida por el tamaño de la página (redondeado hacia arriba).

Después, el algoritmo recorre la matriz tridimensional que representa la imagen y, por cada byte, calcula la página, desplazamiento y componente donde se almacenará el mensaje. Esto se realiza utilizando: bytes recorridos divididos por el tamaño de página para determinar la página; bytes módulo el tamaño de página para determinar el desplazamiento; y el módulo 3 para alternar entre los componentes red, green y blue.

A medida que se recorren los bytes, se escriben los bits del mensaje en la imagen. Los primeros 16 bytes se leen de manera continua, pero a partir del byte 17, se ejecuta lo siguiente: antes de escribir un nuevo caracter en el vector, se inicializa la posición donde se colocará el caracter en 0; Se lee el byte correspondiente de la imagen y se hace una referencia que indica que se leyó ese byte de la imágen y otra referencia indicando que se escribió un bit en la posición del vector del mensaje respectivo. Para saber en qué posición del vector del mensaje se debe colocar un caracter, se lleva un contador de bits. De esta manera, cada vez que se completen 8 bits, el vector del mensaje se desplaza un lugar. De forma similar, también se lleva un contador de bytes para saber en qué página y en qué desplazamiento tendría la referencia de lectura de la imagen y la referencia de escritura del vector del mensaje. Se dividen los bytes que se llevan recorridos sobre el tamaño de la página para saber la página, y módulo entre los bytes que se llevan recorridos y el tamaño de la página para conocer el desplazamiento. Cada vez que se lee un byte, tanto el contador de bits como el contador de bytes recorridos se incrementa, y así el proceso continúa hasta completar todo el mensaje y generar las referencias calculadas.

**Nota**: Al final del documento hay una sección que indica como probar el código

1. Descripción de las estructuras de datos usadas para simular el comportamiento del sistema de paginación y cómo usa dichas estructuras (cuándo se actualizan, con base en qué y en qué consiste la actualización).

Se utiliza una clase llamada Referencia que guarda información de cada operación (tipo de operación, número de página y desplazamiento). Estos son almacenados en una lista de referencias que representa las solicitudes de acceso a las páginas y es recorrida por el simulador de paginación para procesar cada acceso.

Los marcos se representan como una lista que inicialmente está vacía y que se sincroniza entre dos hilos (uno de simulación y otro de modificación de bits). Esta lista se llena hasta alcanzar el máximo de marcos (ingresado por parámetro por el usuario). Cada vez que se procesa una referencia, se verifica si la página correspondiente ya está en la memoria (en los marcos). Si la página ya está, se cuenta como un "hit". Además, cada vez que una página es leída, su bit de lectura se actualiza a true para reflejar que ha sido accedida recientemente.

Cuando hay una falla de página, se procede a reemplazar una página del marco de páginas. Para esto, se utiliza una clase Página que contiene bits de referencia; un bit para modificación y otro bit para lectura. El sistema de reemplazo se basa en el método NRU (Not Recently Used), que clasifica las páginas según sus bits de referencia y modificación. Al buscar una página para reemplazar, se sigue un orden de prioridad de las clases: Primero, se busca una página que no haya sido ni leída ni modificada (bit de lectura false, bit de modificación false). Si no hay páginas en esa condición, se selecciona una que haya sido modificada pero no leída (bit de lectura false, bit de modificación true). La tercera categoría son páginas que hayan sido leídas pero no modificadas (bit de lectura true, bit de modificación false). Finalmente, si no hay páginas de las clases anteriores, se selecciona una que haya sido leída y modificada (bit de lectura true, bit de modificación true). Esta clasificación asegura que las páginas menos usadas recientemente sean las que primero se consideren para el reemplazo, optimizando así la gestión de memoria.

Una vez identificada la página a reemplazar, esta se elimina de la lista de marcos y se agrega la nueva. Para mantener actualizados los bits de referencia, se emplea un proceso que los reinicia periódicamente. Esto simula la disminución de la prioridad de las páginas a medida que pasa el tiempo y facilita el funcionamiento del algoritmo NRU al recalcular las clases de las páginas con base en los estados de sus bits.

Estas estructuras se actualizan cada vez que se procesa una referencia, se realiza un reemplazo de página, o cuando el hilo de envejecimiento restablece los bits de referencia. Las actualizaciones dependen de si una página ya está presente en los marcos, el estado de los bits y la necesidad de reemplazar una página cuando los marcos están llenos, siguiendo el proceso de clasificación mencionado anteriormente.

1. Esquema de sincronización usado. Justifique brevemente dónde es necesario usar sincronización y por qué.

En el sistema de paginación descrito, la sincronización es fundamental para mantener la consistencia de los datos debido a la interacción de dos hilos principales: uno que modifica las páginas y otro que verifica su estado para posibles cambios. Dado que estas páginas se almacenan en una estructura de datos compartida, la sincronización garantiza que las modificaciones de un hilo sean visibles y consistentes para el otro.

La clase SimuladorPaginacion y la clase AlgoritmoEnvejecimiento comparten y modifican la lista de marcos. En SimuladorPaginacion, se utilizan bloques synchronized para asegurar que las operaciones de acceso y modificación de la lista de marcos (como la verificación de si una página está en memoria y su posterior reemplazo) se realicen de manera segura, permitiendo que solo un hilo pueda hacer cambios en la lista a la vez. Esto es clave para evitar que, por ejemplo, un hilo intente acceder a una página mientras otro hilo la está reemplazando, lo que podría causar inconsistencias.

En AlgoritmoEnvejecimiento, se sincroniza el proceso de actualización de las páginas, lo que implica modificar ciertos atributos de las páginas para simular su envejecimiento. Dado que ambos hilos pueden acceder a los marcos simultáneamente, esta sincronización es esencial para evitar que dos hilos alteren la edad de una misma página al mismo tiempo, lo que podría afectar las decisiones de reemplazo.

Sin esta sincronización, podría ocurrir que el hilo que verifica el estado de las páginas para decidir un reemplazo no tenga en cuenta los últimos cambios realizados por el hilo que modifica los estados de las páginas, debido a la falta de exclusión mutua. Esto podría llevar a que no se elija la página óptima para ser reemplazada, afectando la eficiencia del sistema de paginación. Por eso, los métodos sincronizados aseguran que los cambios realizados por un hilo sean considerados por el otro, manteniendo la integridad del sistema.

1. Una tabla con los datos recopilados (y porcentaje de hits y misses por cada escenario simulado).

|  |  |  |  |
| --- | --- | --- | --- |
| Páginas de 512, parrots 370x261mod(100).bmp, mensaje:100 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 1716 | 1712 | 4 |
| 8 | 1716 | 1712 | 4 |
| Páginas de 1024, parrots 370x261mod(100).bmp, mensaje:100 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 1716 | 1713 | 3 |
| 8 | 1716 | 1713 | 3 |
| Páginas de 2048, parrots 370x261mod(100).bmp, mensaje:100 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 1716 | 1714 | 2 |
| 8 | 1716 | 1714 | 2 |
| Páginas de 512, parrots 370x261mod(1000).bmp, mensaje:1000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 17016 | 16997 | 19 |
| 8 | 17016 | 16997 | 19 |
| Páginas de 1024, parrots 370x261mod(1000).bmp, mensaje:1000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 17016 | 17006 | 10 |
| 8 | 17016 | 17006 | 10 |
| Páginas de 2048, parrots 370x261mod(1000).bmp, mensaje:1000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 17016 | 17011 | 5 |
| 8 | 17016 | 17011 | 5 |
| Páginas de 512, parrots 370x261mod(2000).bmp, mensaje:2000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 34016 | 33945 | 71 |
| 8 | 34016 | 33979 | 37 |
| Páginas de 1024, parrots 370x261mod(2000).bmp, mensaje:2000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 34016 | 33997 | 19 |
| 8 | 34016 | 33997 | 19 |
| Páginas de 2048, parrots 370x261mod(2000).bmp, mensaje:2000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 34016 | 34006 | 10 |
| 8 | 34016 | 34006 | 10 |
| Páginas de 512, parrots 370x261mod(4000).bmp, mensaje:4000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 68016 | 67910 | 106 |
| 8 | 68016 | 67944 | 72 |
| Páginas de 1024, parrots 370x261mod(4000).bmp, mensaje:4000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 68016 | 67962 | 54 |
| 8 | 68016 | 67979 | 37 |
| Páginas de 2048, parrots 370x261mod(4000).bmp, mensaje:4000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 68016 | 67997 | 19 |
| 8 | 68016 | 67997 | 19 |
| Páginas de 512, parrots 370x261mod(8000).bmp, mensaje:8000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 136016 | 135703 | 313 |
| 8 | 136016 | 135805 | 211 |
| Páginas de 1024, parrots 370x261mod(8000).bmp, mensaje:8000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 136016 | 135910 | 106 |
| 8 | 136016 | 135944 | 72 |
| Páginas de 2048, parrots 370x261mod(8000).bmp, mensaje:8000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 136016 | 135979 | 37 |
| 8 | 136016 | 135979 | 37 |
| Páginas de 512, parrots 384x256mod(100).bmp, mensaje:100 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 1716 | 1713 | 3 |
| 8 | 1716 | 1713 | 3 |
| Páginas de 1024, parrots 384x256mod(100).bmp, mensaje:100 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 1716 | 1714 | 2 |
| 8 | 1716 | 1714 | 2 |
| Páginas de 2048, parrots 384x256mod(100).bmp, mensaje:100 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 1716 | 1714 | 2 |
| 8 | 1716 | 1714 | 2 |
| Páginas de 512, parrots 384x256mod(1000).bmp, mensaje:1000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 17016 | 16998 | 18 |
| 8 | 17016 | 16998 | 18 |
| Páginas de 1024, parrots 384x256mod(1000).bmp, mensaje:1000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 17016 | 17007 | 9 |
| 8 | 17016 | 17007 | 9 |
| Páginas de 2048, parrots 384x256mod(1000).bmp, mensaje:1000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 17016 | 17011 | 5 |
| 8 | 17016 | 17011 | 5 |
| Páginas de 512, parrots 384x256mod(2000).bmp, mensaje:2000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 34016 | 33980 | 36 |
| 8 | 34016 | 33980 | 36 |
| Páginas de 1024, parrots 384x256mod(2000).bmp, mensaje:2000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 34016 | 33998 | 18 |
| 8 | 34016 | 33998 | 18 |
| Páginas de 2048, parrots 384x256mod(2000).bmp, mensaje:2000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 34016 | 34007 | 9 |
| 8 | 34016 | 34007 | 9 |
| Páginas de 512, parrots 384x256mod(4000).bmp, mensaje:4000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 68016 | 67911 | 105 |
| 8 | 68016 | 67945 | 71 |
| Páginas de 1024, parrots 384x256mod(4000).bmp, mensaje:4000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 68016 | 67980 | 36 |
| 8 | 68016 | 67980 | 36 |
| Páginas de 2048, parrots 384x256mod(4000).bmp, mensaje:4000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 68016 | 67998 | 18 |
| 8 | 68016 | 67998 | 18 |
| Páginas de 512, parrots 384x256mod(8000).bmp, mensaje:8000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 136016 | 135687 | 329 |
| 8 | 136016 | 135874 | 142 |
| Páginas de 1024, parrots 384x256mod(8000).bmp, mensaje:8000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 136016 | 135911 | 105 |
| 8 | 136016 | 135945 | 71 |
| Páginas de 2048, parrots 384x256mod(8000).bmp, mensaje:8000 caracteres | | | |
| Marcos Asignados | Total Referencias | Hits | Fallas |
| 4 | 136016 | 135980 | 36 |
| 8 | 136016 | 135980 | 36 |

1. Una serie de gráficas que ilustren el comportamiento del sistema. Para eso muestre gráficas donde fije tamaño de página y grafique Tamaño de imagen vs. Marcos asignados vs. Porcentaje de hits. La gráfica al final del enunciado ilustra el tipo de gráfica que buscamos.
2. Corra los escenarios y genere gráficas que muestren los datos recopilados para los diferentes escenarios.
3. Además de los escenarios definidos, considere otras configuraciones que le permitan entender cómo afecta la memoria virtual el desempeño del programa.

Algunos escenarios adicionales que podrían afectar la memoria son:

* Cambiar los tamaños de las páginas permitiría analizar cómo se relacionan estos con las fallas de página y el uso de la RAM.
* Usar marcos de página diferentes a 4 u 8 permitiría observar cómo impacta la cantidad de memoria asignada en el sistema de paginación y su relación con el porcentaje de hits y misses.
* Agregar procesos o threads adicionales que utilicen el mismo espacio de memoria que el algoritmo ayudaría a analizar cómo esto afecta el desempeño del sistema.
* Modificar el tamaño de la imagen y del mensaje encriptado cambiaría la cantidad de referencias generadas, lo que permitiría observar el comportamiento de la memoria bajo diferentes condiciones.

1. Incluya las gráficas de tiempo (hits, misses, total).
2. Escriba su interpretación de los resultados: ¿corresponden a los resultados que esperaba, con respecto al número de marcos asignados? Explique su respuesta.

Es importante destacar que al aumentar el tamaño de página, las fallas de página disminuyen. Esto sucede porque un tamaño de página mayor permite almacenar más datos en cada página, lo que reduce la frecuencia con la que se deben reemplazar las páginas en memoria. Este comportamiento es consistente con lo que se esperaba pues con un tamaño mayor de páginas, ya que aumenta la probabilidad de que las páginas ya existan en memoria lo que reduce la necesidad de cargar nuevas páginas desde el SWAP aumentando así el número de hits.

En los mensajes más cortos, los resultados son prácticamente iguales tanto con 4 como con 8 marcos de página, lo cual no corresponde con lo esperado teóricamente, ya que se esperaría una diferencia significativa. Sin embargo, cuando se procesan mensajes más largos (4000 y 8000 caracteres) y se utilizan tamaños de página más pequeños (512 o 1024 bytes), sí se observa una diferencia entre los resultados obtenidos con 4 y 8 marcos de página, siendo los 8 marcos los que presentan menos fallas en las pruebas. Esto sí coincide con lo esperado teóricamente, ya que un mayor número de marcos proporciona más espacio en la memoria real para almacenar las páginas, lo que reduce la cantidad de fallas de página.

1. ¿Si la localidad del problema manejado fuera diferente cómo variarían los resultados? Explique su respuesta. (considere una localidad mayor y una localidad menor).

Si se tiene una mayor localidad, es decir, si se accede con mayor frecuencia a las mismas páginas o a páginas cercanas, se puede asumir que habría menos fallas de página, ya que las páginas permanecerían más tiempo en memoria. En cambio, con una baja localidad, donde los accesos son más dispersos, las páginas serían reemplazadas con mayor frecuencia, lo que produciría más fallas de página.

**Anexo: consideraciones para correr el código**

Cuando el programa inicia, se muestra un menú de tres opciones: la primera opción es para generar las referencias, la segunda para calcular hits y fallas y la tercera para detener el programa.

Primera opción: generar referencias

Para generar las referencias, el programa pide la ruta del archivo del cuál se quiere generar las referencias. La ruta del archivo deber ser puesta completa. Por ejemplo: “C:\Users\majoa\Documents\Universidad\QuintoSemestre\InfraComp\Archivos\caso2-parrots\_mod.bmp". Al presionar ‘Enter’ las referencias se generarán y quedarán guardadas en la siguiente ruta: ‘src/Referencias’ en un archivo llamado ‘referencias.txt’. Este archivo es el que usa el programa por default para la opción 2.

En la carpeta ‘escenarios’ se encuentran los archivos con los que se realizaron las pruebas. En el nombre de estos se encuentran las dimensiones de la imagen, y dentro de paréntesis, la longitud del mensaje escondido que contienen.

Segunda opción: cálculo de hits y fallas

Para la segunda opción el programa pide el número de marcos de página. Posteriormente, toma el archivo ‘referencias.txt’ y calcula los hits y fallas que hay. En otras palabras, a parte del número de marcos de página, el programa no solicita nada más al usuario. Finalizado el proceso, se muestran por consola la cantidad de hits y fallas encontradas.