INFORME CASO 2 INFRACOMP

Mariana Ortega Ramírez 202211233

Carlos Fernando Díaz Vargas 202210262

Andrés Felipe Cordero 202011880

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

El proceso de generación de referencias de página es crucial para la simulación del sistema de paginación. Este proceso está implementado en la clase Proceso, particularmente en el método recuperarMensaje(). A continuación, se describirá en detalle cómo este algoritmo genera las referencias y organiza el acceso a la memoria:

1. Lectura de la imagen y preparación del proceso: El constructor de la clase Proceso recibe la imagen y el tamaño de página como parámetros y los almacena en sus atributos correspondientes. La imagen es leída a través de la clase Imagen, que extrae el contenido de un archivo BMP de 24 bits.
2. Generación de referencias: El método recuperarMensaje() comienza leyendo la longitud del mensaje escondido en la imagen, la cual se encuentra almacenada en los primeros 16 bits del archivo de imagen. Cada acceso a los bytes de la imagen y al mensaje escondido se registra como una "referencia". Estas referencias están estructuradas como operaciones de lectura y escritura en las páginas de memoria virtual. Cada referencia indica si es una lectura (R) o una escritura (W) y apunta a la dirección de memoria virtual correspondiente.
3. Estructura del archivo de referencias: El archivo generado tiene una serie de encabezados que proporcionan información sobre la imagen y la simulación, incluyendo:
    - Tamaño de página (P): El tamaño de cada página de memoria en bytes.
    - Número de filas y columnas (NF y NC): Dimensiones de la imagen.
    - Número de referencias (NR): Total de referencias generadas durante el acceso a la imagen y el mensaje.
    - Número de páginas virtuales (NP): Cantidad de páginas virtuales necesarias para almacenar la imagen y el mensaje.

Una vez que se generan estos encabezados, el algoritmo procede a generar y escribir las referencias una por una, recorriendo la imagen y el mensaje bit por bit. Este proceso se realiza en los métodos leerLongitudReferencias() y recuperar() de la clase Imagen.

4. Acceso a la imagen y al mensaje: El acceso a la imagen se realiza de manera secuencial. El algoritmo recorre los bytes de la imagen en orden row-major, accediendo primero a todos los píxeles de una fila antes de pasar a la siguiente. Los

bits menos significativos de los componentes RGB de cada píxel son los que contienen el mensaje. Una vez recuperados, estos bits son agrupados en bytes para formar los caracteres del mensaje. Cada uno de estos accesos genera una referencia de lectura o escritura, dependiendo de si se está leyendo de la imagen o escribiendo en el vector del mensaje.

2. 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).

El comportamiento del sistema de paginación está cuidadosamente simulado mediante diversas estructuras de datos, todas implementadas en la clase SimuladorMemoria. Estas estructuras permiten controlar el estado de las páginas de memoria y gestionar las fallas y hits. A continuación, se detallan estas estructuras y cómo funcionan dentro del sistema:

1. Tabla de páginas (tablaPaginas y tablaPaginasInv):
    - tablaPaginas: Es un array que mapea las páginas virtuales a los marcos físicos de memoria. Cada entrada en este array indica en qué marco físico se encuentra la página virtual correspondiente. Si una página no está en memoria (falla de página), su valor es -1.
    - tablaPaginasInv: Esta tabla es el inverso de tablaPaginas, y permite determinar qué página virtual se encuentra en cada marco físico de memoria. Es esencial para encontrar y reemplazar páginas cuando se produce una falla de página y no hay marcos disponibles.
2. Bits de referencia y modificación (bitsR y bitsM):
    - bitsR: Este array contiene el bit de referencia para cada página. Indica si la página ha sido referenciada (accedida) recientemente. Este bit es clave para los algoritmos de reemplazo de página, como el algoritmo NRU (Not Recently Used), que se implementa en la simulación.
    - bitsM: El bit de modificación indica si una página ha sido modificada desde que fue cargada en memoria. Si una página es modificada (escritura), este bit se establece en 1. El algoritmo NRU utiliza este bit para dar prioridad al reemplazo de páginas que no han sido modificadas, reduciendo la necesidad de escribir las páginas de vuelta a disco.
3. Contadores de hits y fallas de página:
    - Hits: Cada vez que se accede a una página que ya está en memoria, se produce un hit. El método manejarHit() incrementa el contador de hits y actualiza el bit de referencia de la página accedida.
    - Fallas de página: Si la página requerida no está en memoria, se produce una falla de página, lo que desencadena el método manejarFalloPagina(). Este

método determina si hay un marco libre disponible para cargar la página o si es necesario reemplazar una página en memoria utilizando el algoritmo NRU.

4. Algoritmo de reemplazo NRU (Not Recently Used):
    - El algoritmo NRU clasifica las páginas en cuatro clases en función de los bits R (referencia) y M (modificación). Las páginas que no han sido referenciadas ni modificadas tienen la máxima prioridad para ser reemplazadas, mientras que las páginas que han sido referenciadas y modificadas tienen la menor prioridad. El método encontrarMarcoNRU() es responsable de seleccionar la página adecuada para el reemplazo basándose en esta clasificación.
    - Las clases se dividen de la siguiente forma:
        - Clase 0: Páginas no referenciadas y no modificadas.
        - Clase 1: Páginas no referenciadas pero modificadas.
        - Clase 2: Páginas referenciadas y no modificadas.
        - Clase 3: Páginas referenciadas y modificadas.
5. Sincronización: Dado que el sistema está diseñado para simular concurrencia, se utilizan mecanismos de sincronización como bloques synchronized y locks para evitar condiciones de carrera. Estos mecanismos se aplican en métodos clave como accederPagina(), envejecerPaginas(), y finalizarSimulacion() para garantizar que las actualizaciones a la tabla de páginas y los bits de referencia se realicen de manera segura cuando múltiples threads acceden a estas estructuras simultáneamente.

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

El uso de concurrencia es esencial para simular un sistema de paginación realista. El esquema de sincronización en el sistema implementado está diseñado para coordinar los accesos concurrentes a las estructuras de datos compartidas (como las tablas de páginas y los bits de referencia). A continuación, se describe el esquema de sincronización y las técnicas utilizadas:

1. Uso de CyclicBarrier: La clase CyclicBarrier se utiliza para sincronizar la finalización de los dos threads principales de la simulación:
    - procesoThread: Este thread simula el acceso a las páginas durante la ejecución del proceso que está recuperando el mensaje de la imagen.
    - relojThread: Este thread simula el reloj de envejecimiento, que periódicamente reinicia los bits de referencia de las páginas, imitando el comportamiento de un reloj en un sistema de paginación.

La barrera asegura que ambos threads esperen el uno al otro antes de finalizar, lo que garantiza que la simulación de acceso a páginas y envejecimiento se completen correctamente antes de calcular los resultados finales.

2. Bloques synchronized y uso de Lock:
    - Métodos como accederPagina(), manejarFalloPagina(), y envejecerPaginas() están protegidos por bloques synchronized para asegurar que solo un thread acceda y modifique la tabla de páginas y los bits de referencia a la vez.
    - El uso de Lock en métodos como finalizarSimulacion() proporciona una alternativa flexible para sincronizar el acceso a la variable continuarSimulacion, asegurando que el valor de esta variable solo sea modificado de manera segura por un thread.
3. Envejecimiento de páginas: El método envejecerPaginas() se ejecuta periódicamente cada 20 ms y resetea los bits de referencia de todas las páginas. Esto simula el comportamiento de un sistema que evalúa cuáles páginas han sido referenciadas en cada ciclo del reloj y prepara el sistema para futuros reemplazos de página.

4.Una tabla con los datos recopilados (y porcentaje de hits y misses por cada escenario simulado), Una serie de gráficas que ilustren el comportamiento del sistema, análisis de gráficas y datos:

Para mostrar que el algoritmo es correcto, al correrlo con la imagen de ejemplo de caso2-parrots_mod.bmp, y el mensaje de caso2-mensaje_dollshousep1.txt, las referencias dan:

P=256

NF=256

NC=384

NR=81956

NP=1171

Las cuales cumplen la diferencia de +/- 5% entre los valores mostrados en el enunciado.

Ahora bien, se procederá a revisar las pruebas y el análisis de los resultados.

Para realizar las pruebas, se utilizaron dos imágenes: imagen1.bmp, es una imagen de 500 x 300 pixeles, y caso2-parrots.bmp, de 384 x 256 pixeles. Y para estas pruebas se hicieron variaciones de:

- Tamaño de Pagina: 512, 1024, 2048
- Marcos de página: 4,8
- Caracteres de mensaje escondido: 109, 938, 1648, 3467, 8191

Se obtuvieron los siguientes experimentos:

paginas	512					
Imagen	Marcos Pagina	fallas	NR	Tiempo (ms)	hits	caracteres
imagen1	4	3	1869	2959	1866	109
imagen1	4	17	15962	23782	15945	938

imagen1	4	30	28032	42527	28002	1648
imagen1	4	63	58955	85842	58892	3467
imagen1	4	146	139263	203776	139117	8191
imagen1	8	3	1869	2763	1866	109
imagen1	8	17	15962	23653	15945	938
imagen1	8	30	28032	40963	28002	1648
imagen1	8	62	58955	85809	58893	3467
imagen1	8	147	139263	204657	139116	8191

paginas	512					
Imagen	Marcos Pagina	fallas	NR	Tiempo (ms)	hits	caracteres
caso2-parrots	4	3	1869	2802	1866	109
caso2-parrots	4	17	15962	23553	15495	938
caso2-parrots	4	30	28032	41563	28002	1648
caso2-parrots	4	63	58955	85913	58892	3467
caso2-parrots	4	146	139263	203937	139117	8191
caso2-parrots	8	3	1869	2746	1866	109
caso2-parrots	8	17	15962	23560	15945	938
caso2-parrots	8	30	28032	41845	28002	1648
caso2-parrots	8	62	58955	85829	58893	3467
caso2-parrots	8	147	139263	204158	139116	8191

paginas	1024					
Imagen	Marcos Pagina	fallas	NR	Tiempo (ms)	hits	caracteres
imagen1	4	2	1869	2760	1867	109
imagen1	4	9	15962	23069	15953	938
imagen1	4	15	28032	41032	28017	1648
imagen1	4	32	58955	86296	58923	3467
imagen1	4	73	139263	205908	139190	8191
imagen1	8	2	1869	2810	1867	109
imagen1	8	9	15962	22936	15953	938
imagen1	8	15	28032	41514	28017	1648
imagen1	8	32	58955	87351	58923	3467
imagen1	8	73	139263	206114	139190	8191

paginas	1024					
Imagen	Marcos Pagina	fallas	NR	Tiempo (ms)	hits	caracteres
caso2-parrots	4	2	1869	2752	1867	109
caso2-parrots	4	9	15962	23164	15953	938

caso2-parrots	4	15	28032	41144	28017	1648
caso2-parrots	4	32	58955	86943	58923	3467
caso2-parrots	4	73	139263	206002	139190	8191
caso2-parrots	8	2	1869	2896	1867	109
caso2-parrots	8	9	15962	22991	15953	938
caso2-parrots	8	15	28032	41698	28017	1648
caso2-parrots	8	32	58955	87288	58923	3467
caso2-parrots	8	73	139263	206965	139190	8191

paginas	2048					
Imagen	NR	Marcos Pagina	Tiempo (ms)	fallas	hits	caracteres
imagen1	1869	4	2763	2	1867	109
imagen1	15962	4	23748	5	15957	938
imagen1	28032	4	41356	8	28024	1648
imagen1	58955	4	86925	16	58939	3467
imagen1	139263	4	255568	37	139226	8191
imagen1	1869	8	2741	2	1867	109
imagen1	15962	8	23548	5	15957	938
imagen1	28032	8	41507	8	28024	1648
imagen1	58955	8	87360	16	58939	3467
imagen1	139263	8	255742	37	139226	8191

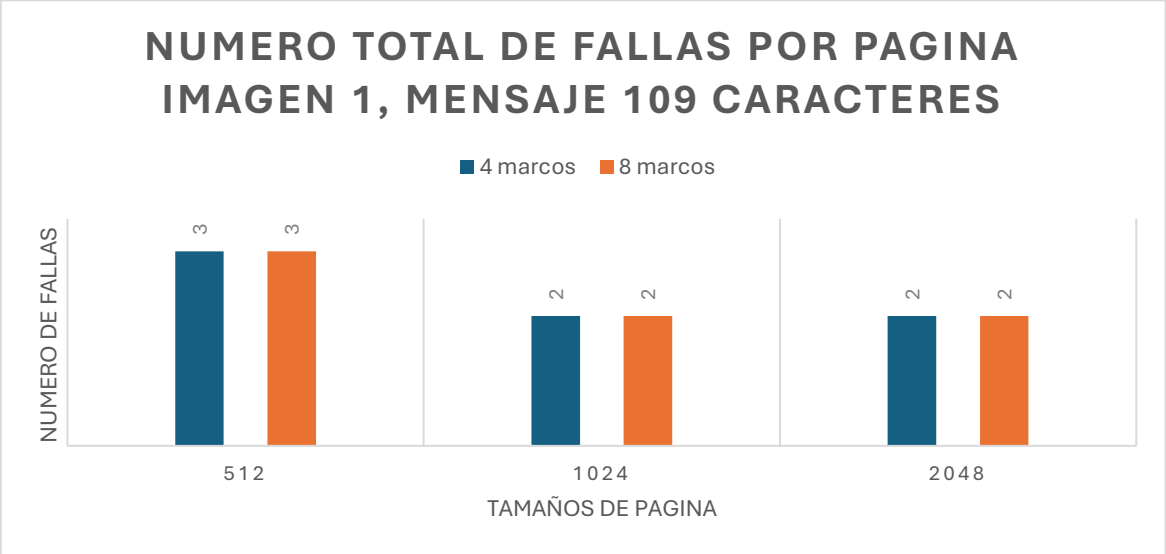
paginas	2048					
Imagen	NR	Marcos Pagina	Tiempo (ms)	fallas	hits	caracteres
caso2-parrots	1869	4	2742	2	1867	109
caso2-parrots	15962	4	23727	5	15957	938
caso2-parrots	28032	4	41335	8	28024	1648
caso2-parrots	58955	4	86904	16	58939	3467
caso2-parrots	139263	4	255547	37	139226	8191
caso2-parrots	1869	8	2720	2	1867	109
caso2-parrots	15962	8	23527	5	15957	938
caso2-parrots	28032	8	41486	8	28024	1648
caso2-parrots	58955	8	87339	16	58939	3467

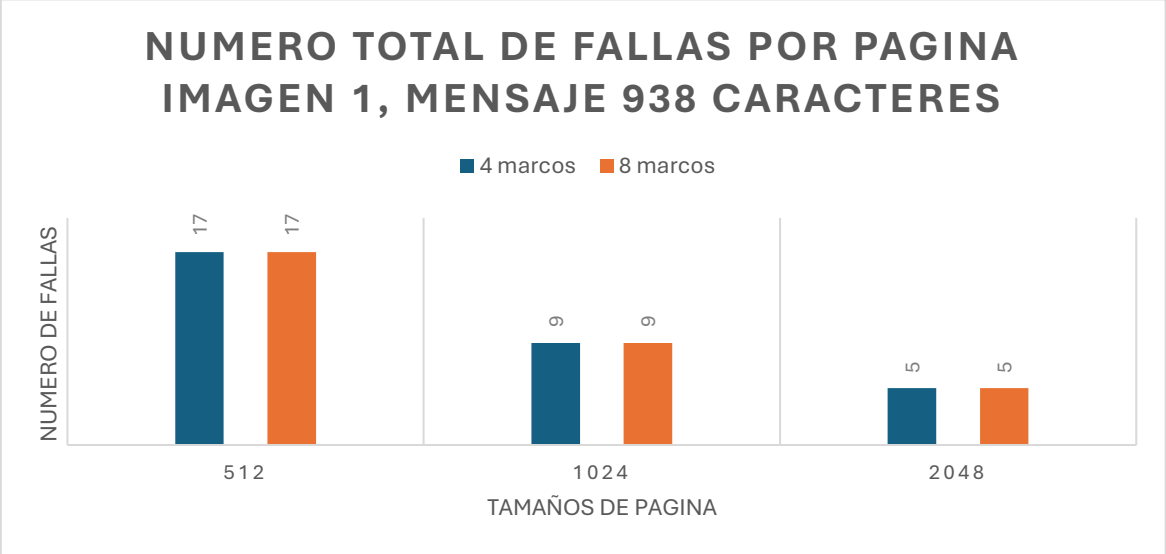
caso2-parrots	139263	8	255721	37	139226	8191

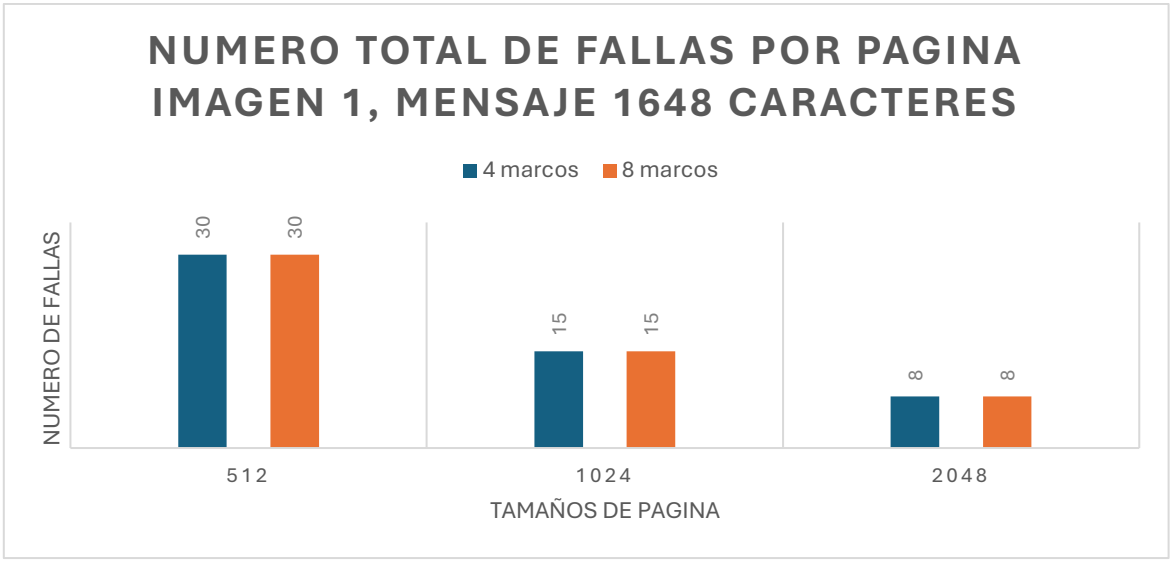
Inicialmente notamos que, para ambas imágenes, no se varía mucho, más que el tiempo, mientras que el número de referencias, de misses (fallas), de hits, no cambiaban entre ellas. Lo cual se debe a que lo que genera cambios en las referencias, hits y misses, es cambiar el mensaje principalmente, y cambiar los tamaños de página. Siendo así no importa en estos casos si la imagen era más grande que otra, porque se les analizó con respecto a los mismos mensajes.

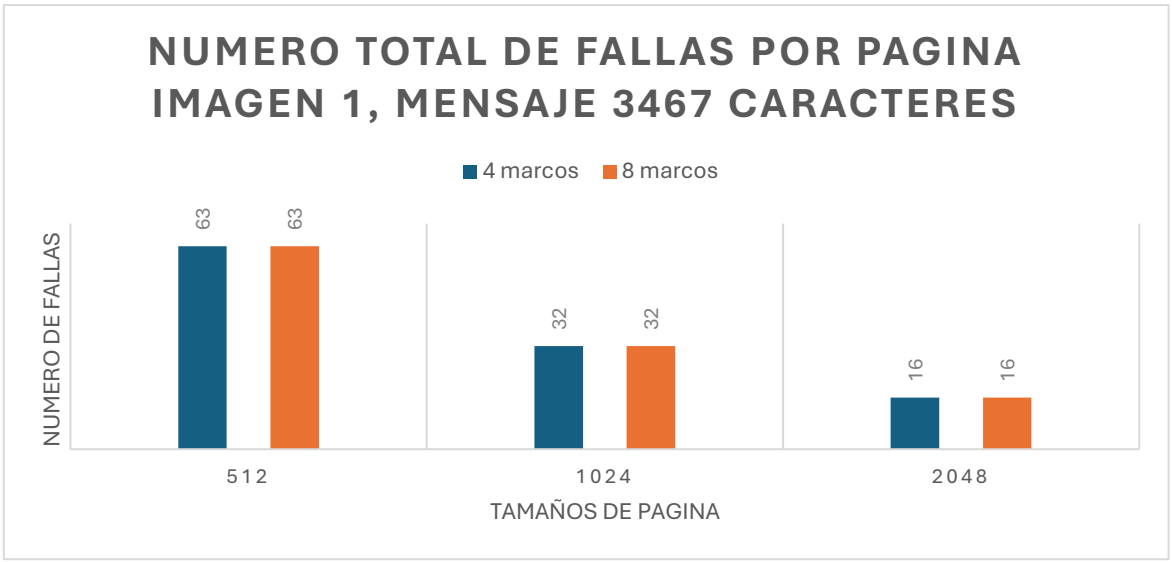
Ahora bien, como las imágenes dieron resultados muy similares. Para analizar las fallas dependiendo de tamaño de pagina y caracteres del mensaje, solo lo consideraremos para la imagen1.

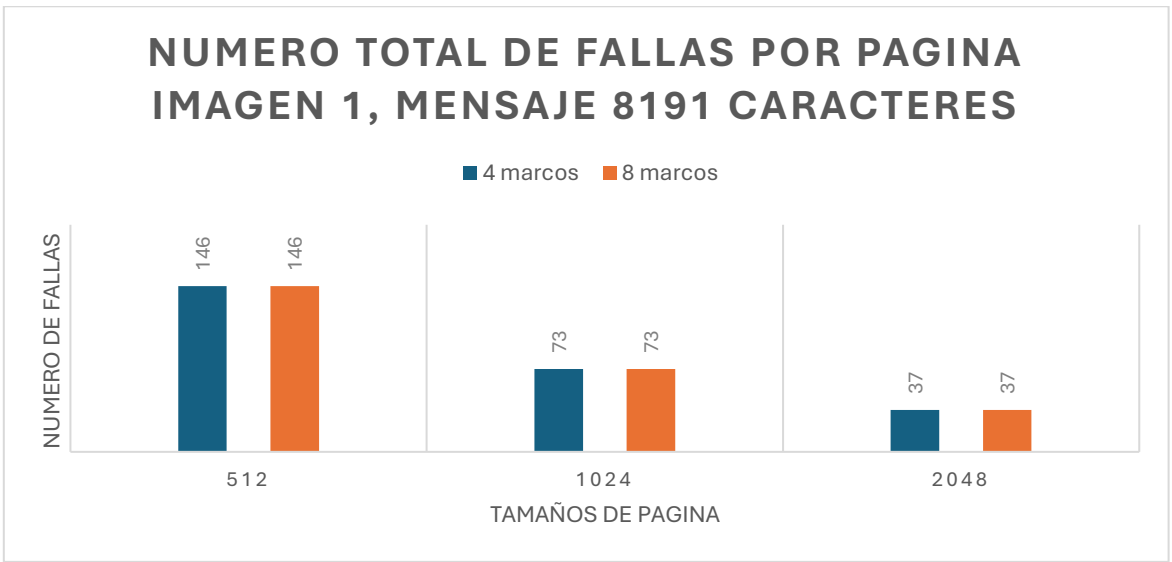
Graficas de fallas con respecto a caracteres, imagen, tamaños de página.

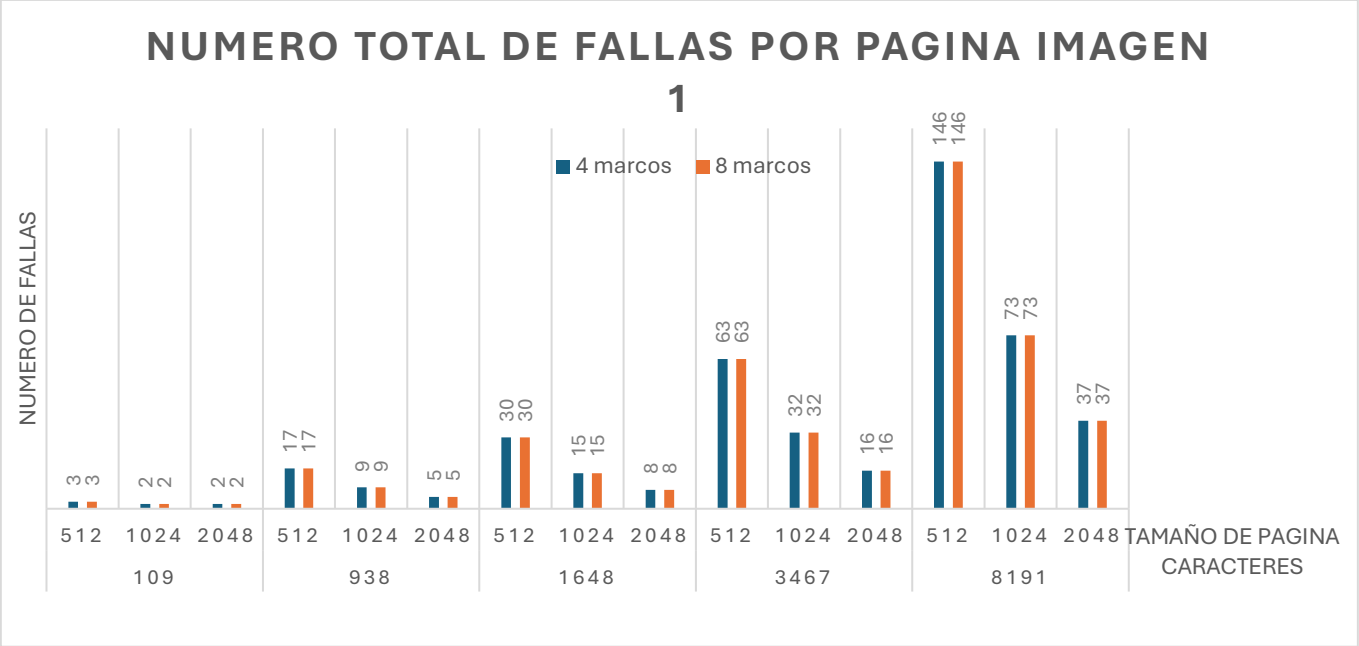





NUMERO TOTAL DE FALLAS POR PAGINA
IMAGEN 1, MENSAJE 1648 CARACTERES
4 marcos
8 marcos
NUMERO DE FALLAS
30
30
15
15
8
8
512
1024
2048
TAMAÑOS DE PAGINA


NUMERO TOTAL DE FALLAS POR PAGINA
IMAGEN 1, MENSAJE 3467 CARACTERES
4 marcos
8 marcos
NUMERO DE FALLAS
63
63
32
32
16
16
512
1024
2048
TAMAÑOS DE PAGINA

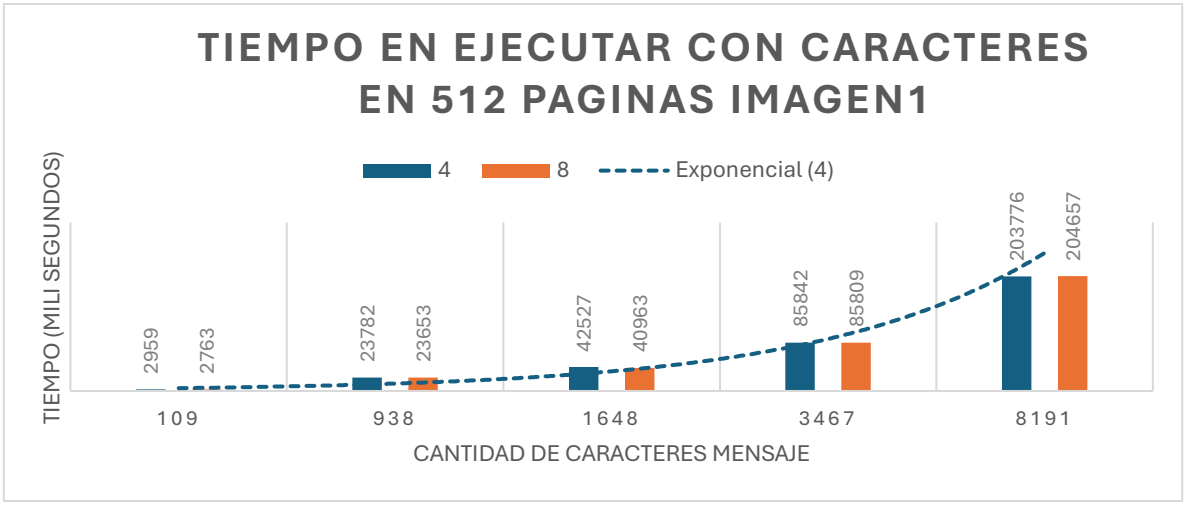

NUMERO TOTAL DE FALLAS POR PAGINA
IMAGEN 1, MENSAJE 8191 CARACTERES
4 marcos
8 marcos
NUMERO DE FALLAS
146
146
73
73
37
37
512
1024
2048
TAMAÑOS DE PAGINA

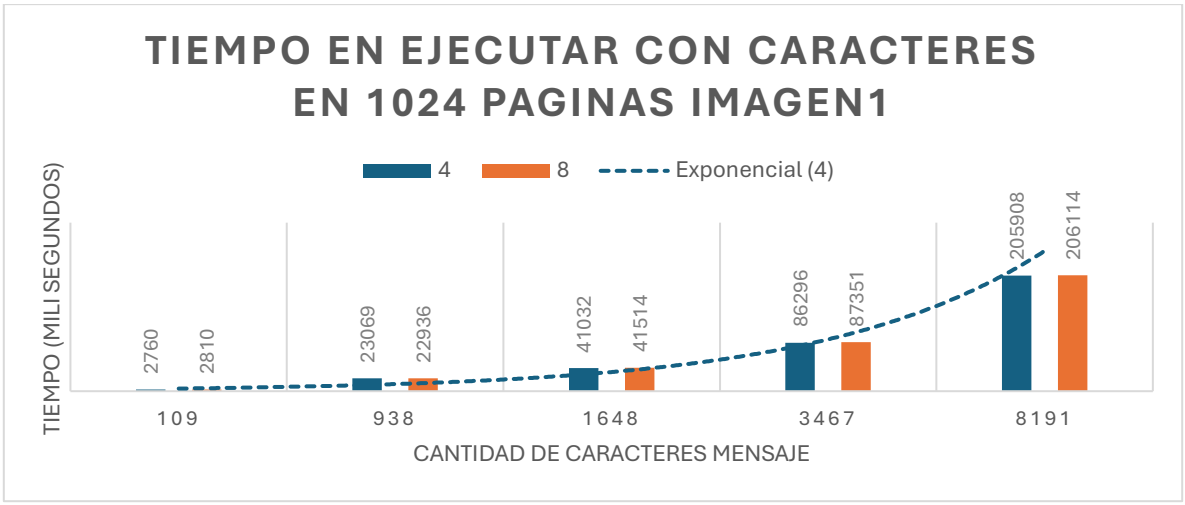


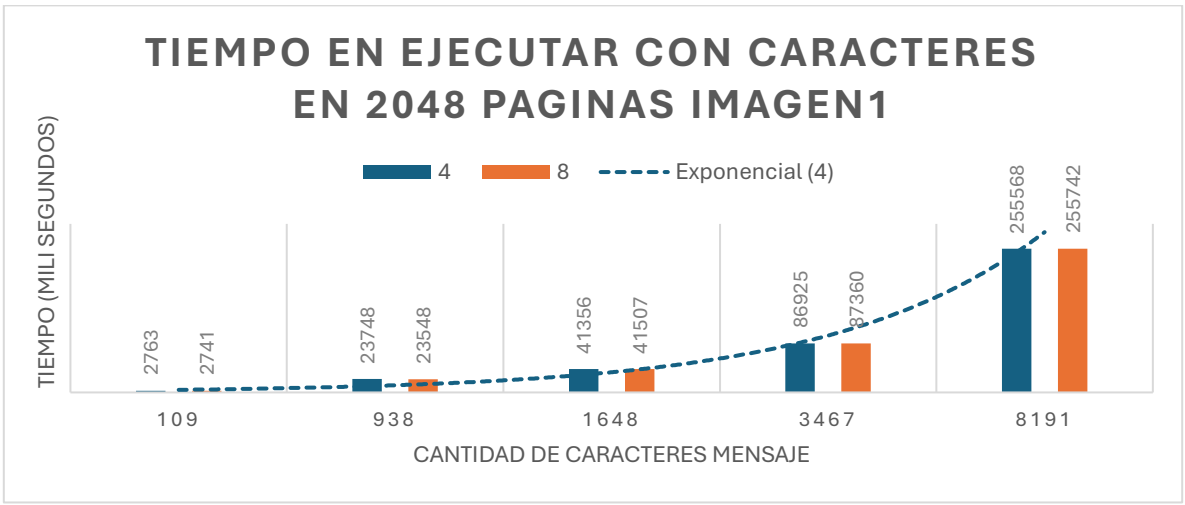
Al analizar el comportamiento de los misses, es decir los fallos, con respecto al tamaño de páginas, los marcos, y la cantidad de caracteres del mensaje, se puede definir que:

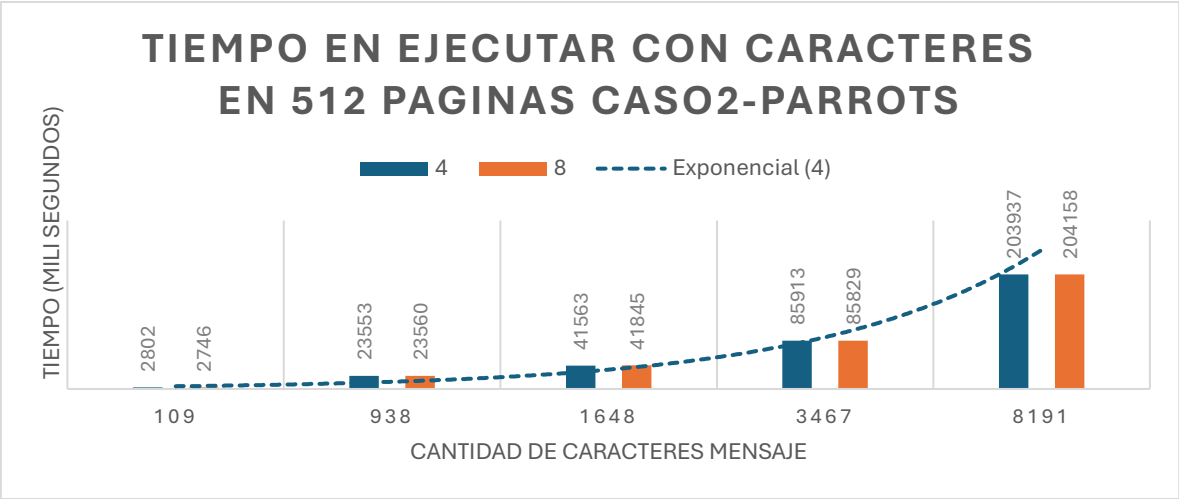
1. Entre mayor sea el numero de tamaño de página, van a ser menores las cantidades se misses, ya que puede haber una mayor cantidad de datos por página, entonces es mucho menos probable que el sistema necesita cargar nuevas paginas desde el almacenamiento. Lo cual es acorde con la teoría y en los experimentos se puede evidenciar este comportamiento.
2. El numero de misses no cambia al modificar el numero de marcos de pagina ya que el algoritmo de reemplazo porque el patrón de acceso a las paginas es el mismo, el algoritmo sigue reemplazado sin importar los marcos de página.
3. Entre mayor la cantidad de caracteres del mensaje escondido, mayor es la cantidad de misses, ya que hay muchos más accesos a memoria, y a mayor numero de paginas para extraer los bits distribuido sen la imagen, lo que conlleva la cantidad de fallos de página cada vez que una página es reemplazada.

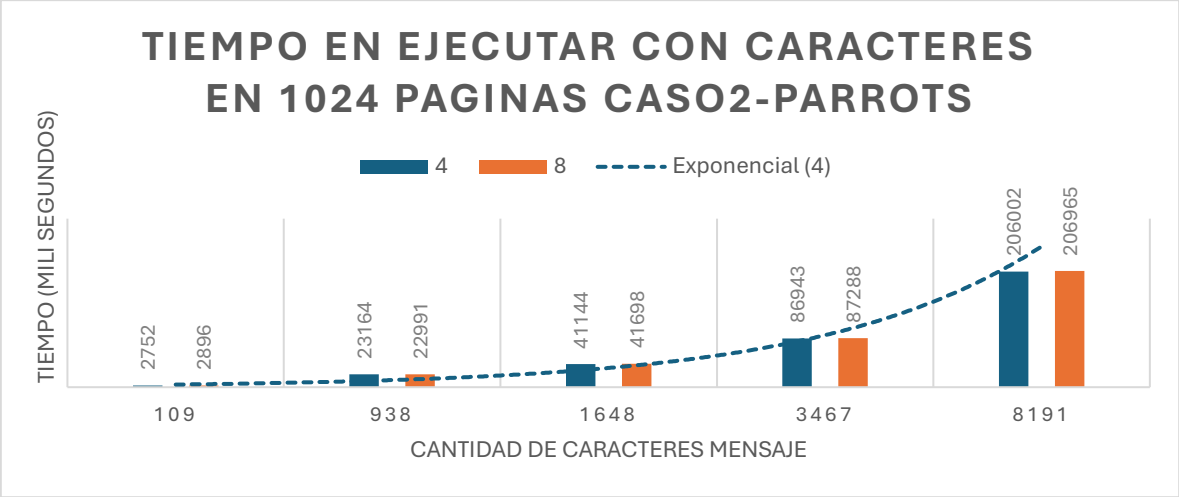
Graficas de tiempo

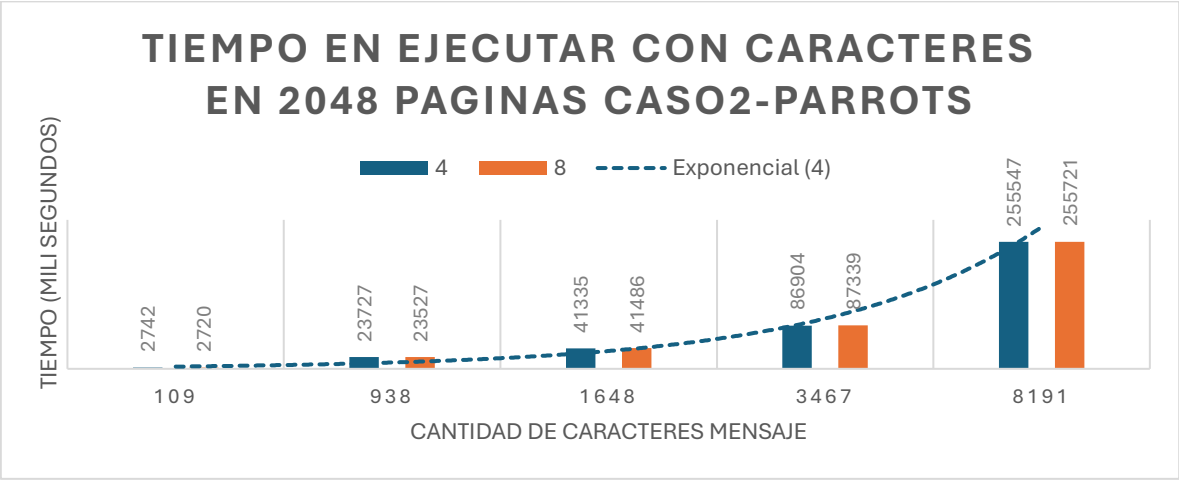

TIEMPO EN EJECUTAR CON CARACTERES EN 512 PAGINAS IMAGEN1
4
8
Exponencial (4)
TIEMPO (MILI SEGUNDOS)
2959
2763
23782
23653
42527
40963
85842
85809
203776
204657
109
938
1648
3467
8191
CANTIDAD DE CARACTERES MENSAJE


TIEMPO EN EJECUTAR CON CARACTERES EN 1024 PAGINAS IMAGEN1
4
8
Exponencial (4)
TIEMPO (MILI SEGUNDOS)
2760
2810
23069
22936
41032
41514
86296
87351
205908
206114
109
938
1648
3467
8191
CANTIDAD DE CARACTERES MENSAJE


TIEMPO EN EJECUTAR CON CARACTERES EN 2048 PAGINAS IMAGEN1
4
8
Exponencial (4)
TIEMPO (MILI SEGUNDOS)
2763
2741
23748
23548
41356
41507
86925
87360
255568
255742
109
938
1648
3467
8191
CANTIDAD DE CARACTERES MENSAJE







De todas estas pruebas anteriores se pueden notas los siguientes factores:

1. Tomando en cuenta que los caracteres del mensaje escondido se incrementaron de manera exponencial aproximadamente, se reflejó este comportamiento mismo en el tiempo de ejecución de calculo de hits y misses. Esto es indicativo de que los marcos de memoria comienzan a llenarse y se producen más fallos de página, lo que aumenta el tiempo de cómputo.
2. Notamos que para 4 marcos y 8 marcos no hay mayor diferencia, sin embargo, hay una tendencia constante en mayoría de casos, la cual indica que el sistema es mucho más eficiente con 8 marcos de página que con 4, ya que puede manejar más páginas en la memoria sin necesidad de reemplazo frecuente.

¿Corresponden a los resultados que esperaba, con respecto al número de marcos asignados?

Sí, los resultados se alinean con lo que se espera desde una perspectiva teórica. En sistemas de paginación, contar con más marcos de página permite al sistema conservar un mayor número de páginas en memoria, lo que disminuye la necesidad de cargar y reemplazar páginas frecuentemente. Esto resulta en una menor cantidad de fallos de página y tiempos de ejecución más cortos.

Por otro lado, cuando hay menos marcos de página (como en el caso de tener solo 4), la memoria se llena más rápidamente, lo que obliga al sistema a reemplazar páginas con mayor frecuencia y, por ende, aumenta los fallos de página. Esto sucede porque el sistema debe estar constantemente cargando nuevas páginas desde el almacenamiento secundario, lo que incrementa los tiempos de ejecución. Teóricamente, esto se ajusta al comportamiento clásico de la Ley de localidad, donde el número de marcos influye directamente en cuántas páginas "locales" pueden permanecer en memoria, afectando así el rendimiento del sistema.

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

La localidad de referencia tiene un impacto directo en la cantidad de aciertos y fallas de página. Cuando un proceso accede repetidamente a un pequeño conjunto de páginas (alta localidad), se produce un aumento en el número de aciertos, dado que es probable que esas páginas ya estén en memoria. En cambio, si el acceso es disperso y abarca una gran cantidad de páginas diferentes (baja localidad), se observará un aumento en las fallas de página.

En este sistema, el algoritmo NRU, que se implementa a través del método encontrarMarcoNRU(), toma decisiones sobre el reemplazo de páginas basándose en su clasificación según la localidad (uso reciente y modificaciones). Con una alta localidad, el sistema puede mantener en memoria las páginas más importantes, lo que ayuda a disminuir las fallas.

Este comportamiento se simula en la segunda opción del programa, donde se utilizan diversas cantidades de marcos de página para analizar su efecto en el rendimiento.