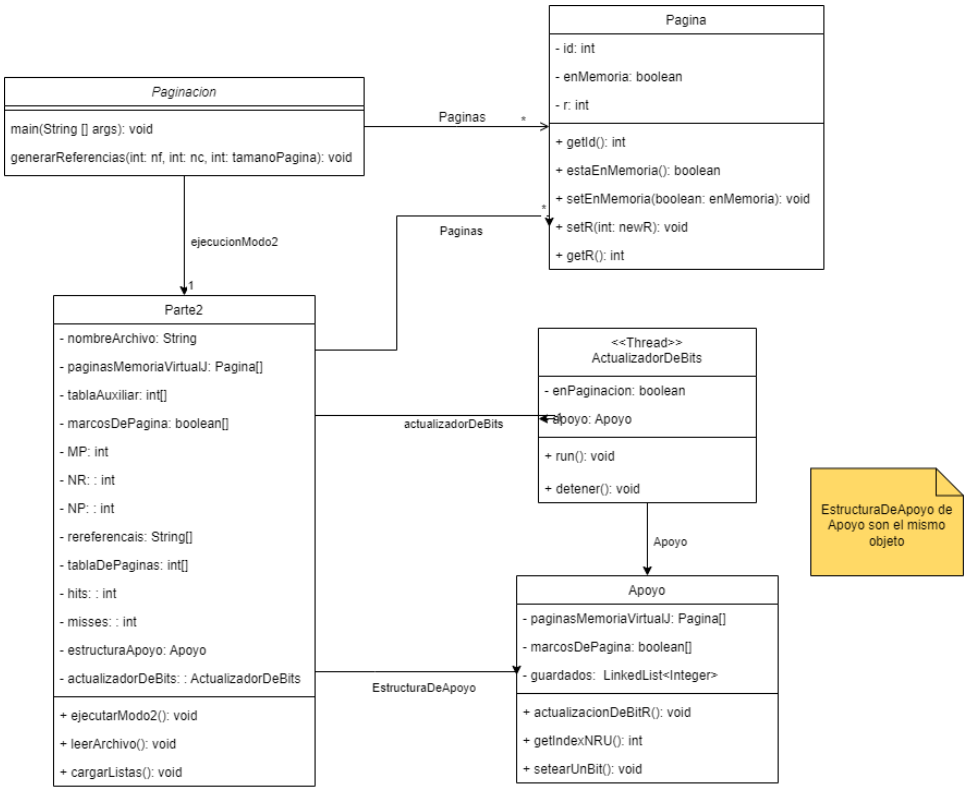
CASO 2 INFRAESTRUCTURA COMPUTACIONAL

Integrante 1: Juan David Duarte Yara - 202215070

Integrante 2: David Ernesto Zamora Cortes - 202113407

Integrante 3: Laura Valentina Lara Díaz -201912967

UML



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

Inicialmente guardamos en un mapa las páginas correspondientes a las diferentes entradas de cada matriz, es decir, de la matriz de entrada, el filtro y la matriz de salida. Cada matriz tiene su propio mapa donde se almacenan las páginas que corresponden a cada valor. Ese mapa tiene como llave lo que llamamos clave, que en realidad es un string que se compone de “fila-columna”. Entonces al final de la primera parte tenemos guardadas en 3 mapas diferentes las páginas a las que corresponden los valores de cada matriz. Adicionalmente usamos un contador que aumenta de 4 en 4 por cada entrada. Esto es porque cada entrada es un entero de 4 bytes. Con ese contador y el tamaño de página podemos calcular la página que corresponde a cada entrada y el desplazamiento. Sin embargo, solo se guarda la página con una clave, no el desplazamiento. Luego de haber guardado todas las páginas para cada valor, iteramos de la misma forma en la que el filtro se mueve sobre la matriz base, y para cada una de las multiplicaciones que se hacen al aplicar el filtro escribimos las referencias. Para escribirlas solo hace falta obtenerlas de nuestros mapas usando la fila y la columna como clave. Adicionalmente, dentro de esas iteraciones también escribimos las referencias a la matriz de

resultados. Finalmente, y fuera de esas iteraciones del filtro sobre la matriz, escribimos las referencias que ocurren al aplicar los valores de 0 o de 255 sobre los bordes de la matriz de resultados. Los desplazamientos para cada referencia los calculamos con la fila y la columna del dato que se le calculará la referencia y teniendo en cuenta otras entradas que ya estaban guardadas.

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).

En primer lugar, se emplean listas para representar la memoria virtual, que es una lista con objetos de tipo “Pagina” y la memoria real que es una lista de booleanos, en donde el índice indica el id de la página y el valor indica si el marco de página está siendo ocupado por una página. También se usa una lista para la tabla de páginas cuyo índice indica la página en memoria virtual y valor indica a que posición de marco de página esta almacenado dicha página, si no está en ningún marco contiene el valor de “–1” Estas listas se actualizan cuando se carga un proceso en memoria, se accede a una página, o se mueve una página entre memoria real y virtual **(paginasMemoriaVirtual, TablaDePaginas y paginasMemoriaReal)**. Además, se utilizan arreglos y variables para gestionar diferentes parámetros del sistema, como el tamaño de la memoria física (MP), el número de referencias (NR), el número de páginas (NP), entre otros.

El algoritmo NRU se implementa mediante las clases **Apoyo** y **ActualizadorDeBits**. La clase **Apoyo** contiene métodos para actualizar el bit R de las páginas y seleccionar páginas para ser reemplazadas según la categorización del algoritmo. Específicamente, La estructura de lista **guardados** en la clase **Apoyo** se utiliza para mantener un registro de las páginas que han sido referenciadas para que el bit R cambie de estado durante un intervalo de tiempo específico. Esta lista se inicializa como una lista enlazada vacía cuando se crea una instancia de la clase **Apoyo** y en esta lista se almacenan las ID’s de las páginas a las que se les tiene que actualizar el bit R. Cuando se pide la actualización de bits R las páginas indicadas en esta lista son actualizadas y el contenido de la lista es borrado (**clear**). La función **referenciarPagina** es importante para la actualización de los bits R de las páginas. Cuando una página es referenciada durante la ejecución del algoritmo de paginación, se llama a esta función con el número de la página como argumento. La función verifica si el estado actual es "Actualizando" (lo cual indica que ya se está realizando una actualización) y si es así, espera hasta que el estado cambie. Luego, agrega el número de la página a la lista **guardados**, registrando así que esa página ha sido referenciada recientemente.

La función **actualizacionDeBitR** se encarga de actualizar los bits R de las páginas según la información almacenada en **guardados**. Recorre todas las páginas en **paginasMemoriaVirtualJ** y verifica si el número de la página está presente en **guardados**. Si lo está, se establece el bit R de esa página en 1. Después de actualizar los bits R de todas las páginas relevantes, se limpia la lista **guardados** para iniciar un nuevo intervalo de referencias. Finalmente, se establece el estado en "Esperando" y se notifica a cualquier hilo en espera que el proceso de actualización ha finalizado.

En la función **ejecutarModo2** se realiza la simulación del sistema de paginación, donde se recorren las referencias de memoria, se actualizan los bits R según el intervalo especificado y se aplica el algoritmo NRU para gestionar la memoria física y virtual. Se hace uso de la clase **Pagina** para representar las páginas y sus atributos, como el bit R.

En la clase **ActualizadorDeBits** se extiende de la clase Thread y en esta se está constantemente pidiendo la actualización del bit R de las páginas que son guardadas para que sean actualizadas

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

Para la ejecución del modo 2, se utilizan dos Thread, el Thread main que estará ejecutando codigo de la clase Parte2.java, se encarga de ir actualizando el estado de la tabla de páginas y los marcos de página en RAM y un segundo Thread, objeto instancia de la clase ActualizadorDeBits.java, se encarga de la actualización del bit R. Estas dos clases comparten a un objeto instancia de la clase Apoyo.java, acá se encuentras las funciones importantes para las actualizaciones, estas funciones que manejan los datos compartidos por los Threads usan exclusion mutua, por tanto, tiene las etiquetas de synchronized. Para que no hallan mal alteraciones de datos. También se tiene que cada iteración de la clase main se ejecuta cada un milisegundo, mientras que el actualizador de bits lo hace cada 4 milisegundos, ya que como criterio se tomó que un pulso de reloj equivale a un milisengundo.

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

Datos recopilados para 16B con dimensiones de matriz 4x4, 8x8, 16x16, 32x32

16 B	4x4						
Marcos asignados	Total de referencias	Hits	% Hits	Fallas	tiempo ejecucion (ns)	tiempo hits (ns)	tiempo fallas (ns)
4	88	59	67.05	29	290001770	2640	880000000
8	88	77	87.5	11	110002310	2640	880000000
12	88	78	88.64	10	100002340	2640	880000000
16	88	78	88.64	10	100002340	2640	880000000

16 B	8x8						
Marcos asignados	Total de referencias	Hits	%Hits	Fallas	tiempo ejecucion	tiempo hits	tiempo fallas
4	712	345	48.46	367	3670010350	21360	1469934592
8	712	530	74.44	182	1820015900	21360	1469934592
12	712	583	81.88	129	1290017490	21360	1469934592
16	712	620	87.08	92	920018600	21360	1469934592

16 B	16x16						
Marcos asignados	Total de referencias	Hits	% Hits	Fallas	tiempo ejecucion (ns)	tiempo hits (ns)	tiempo fallas (ns)
4	3784	1189	31.42	5295	25950035670	113520	37840000000

8	3784	1948	52.48	1836	18360058440	113520	37840000000
12	3784	2318	61.26	1466	14660069540	113520	37840000000
16	3784	2742	72.46	1042	10420082260	113520	37840000000
32	3784	3196	84.46	588	5880095880	113520	37840000000
64	3784	3574	94.45	210	2100107220	113520	37840000000
128	3784	3554	96.56	130	1300109620	113520	37840000000

16 B	32x32						
Marcos asignados	Total de referencias	Hits	%Hits	Fallas	tiempo ejecucion (ns)	tiempo hits (ns)	tiempo fallas (ns)
4	17224	5078	29.48	12146	1.2146E+11	516720	1.7224E+11
8	17224	7296	42.36	9928	99280218880	516720	1.7224E+11
12	17224	8553	49.66	8671	86710256590	516720	1.7224E+11
16	17224	8966	52.06	8258	82580268980	516720	1.7224E+11
32	17224	11854	68.8	5370	53700355620	516720	1.7224E+11
64	17224	13867	80.51	3357	32130420330	516720	1.7224E+11
128	17224	15626	90.72	1598	15980468780	516720	1.7224E+11

Datos recopilados para 32B con dimensiones de matriz 4x4, 8x8, 16x16, 32x32

32 B	4x4						
Marcos asignados	Total de referencias	Hits	% Hits	Fallas	tiempo ejecucion	tiempo hits	tiempo fallas
4	88	74	84.09	14	140002220	2640	880000000
8	88	82	93.18	6	60002460	2640	880000000
12	88	82	93.18	6	60002460	2640	880000000
16	88	82	93.18	6	60002460	2640	880000000

32 B	8x8						
Marcos asignados	Total de referencias	Hits	%Hits	Fallas			
4	712	528	74.16	184	1469934592	21360	7120000000
8	712	664	93.26	48	1469934592	21360	7120000000
12	712	685	96.21	27	1469934592	21360	7120000000
16	712	635	89.19	77	770019050	21360	7120000000

32 B	16x16						
Marcos asignados	Total de referencias	Hits	% Hits	Fallas			
4	3784	2139	56.53	1645	16450064170	113520	814705664
8	3784	2980	78.75	804	8040089400	113520	814705664
12	3784	3329	87.98	455	4550099870	113520	814705664
16	3784	3499	92.47	285	2850104970	113520	814705664

32 B	32x32						
Marcos asignados	Total de referencias	Hits	% Hits	Fallas	tiempo ejecucion	tiempo hits	tiempo fallas
4	17224	7545	43.81	9679	95520230160	516720	1.7224E+11
8	17224	10262	59.58	6962	69620307860	516720	1.7224E+11
12	17224	11594	67.31	5630	56300347820	516720	1.7224E+11
16	17224	13076	75.92	4148	41480392280	516720	1.7224E+11

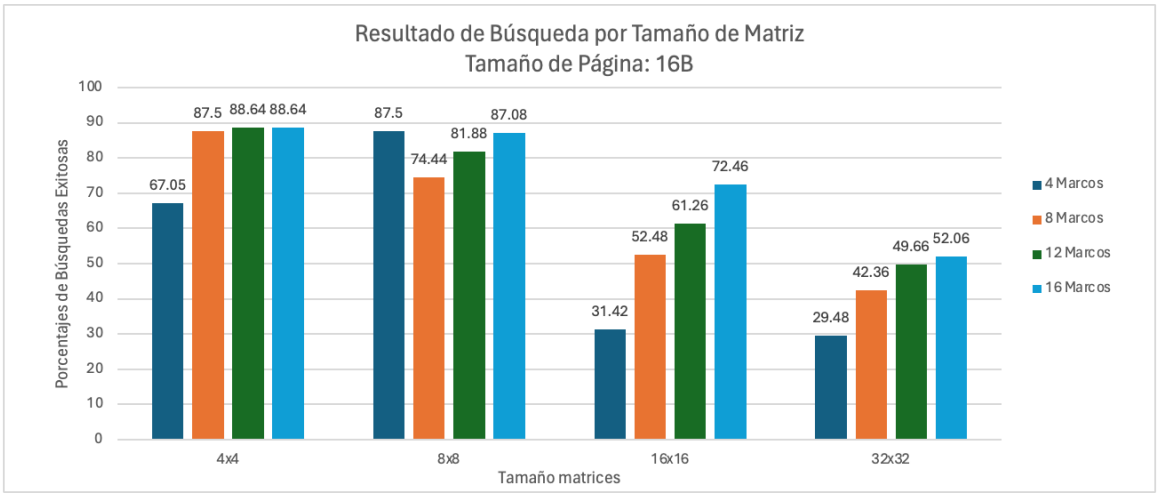
Una comparación específica del %Hits para 16B:

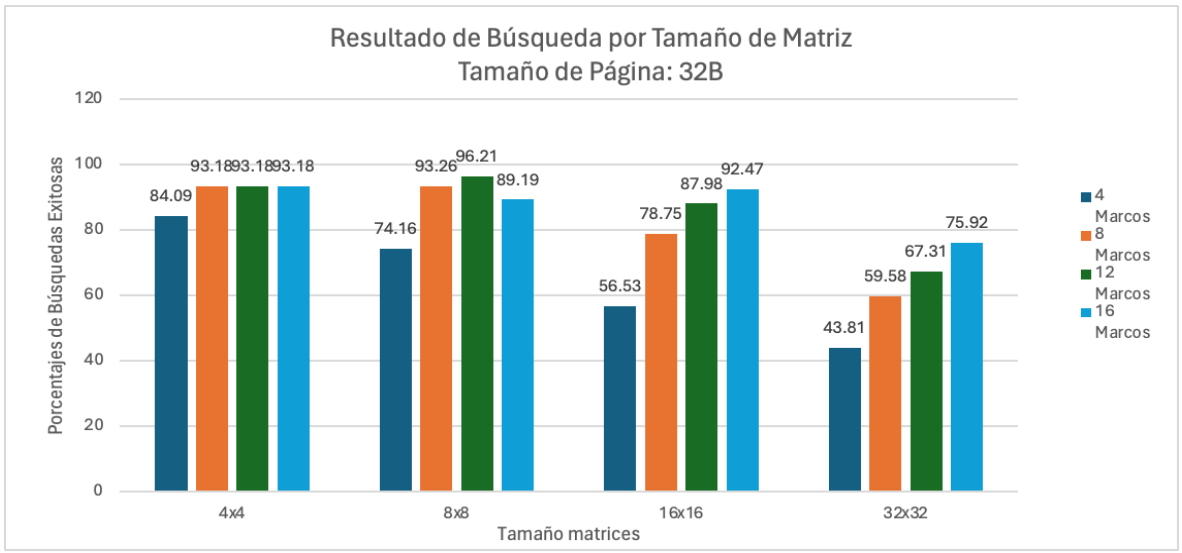
	4 Marcos	8 Marcos	12 Marcos	16 Marcos
Tamaño matriz	%HITS	%HITS	%HITS	%HITS
4x4	67.05	87.5	88.64	88.64
8x8	87.5	74.44	81.88	87.08
16x16	31.42	52.48	61.26	72.46
32x32	29.48	42.36	49.66	52.06

Una comparación detallada del %Hits para 32B:

	4 Marcos	8 Marcos	12 Marcos	16 Marcos
Tamaño matriz	%HITS	%HITS	%HITS	%HITS
4x4	84.09	93.18	93.18	93.18
8x8	74.16	93.26	96.21	89.19
16x16	56.53	78.75	87.98	92.47
32x32	43.81	59.58	67.31	75.92

5. Una serie de gráficas que ilustren el comportamiento del sistema. Para eso muestre gráficas donde fije tamaño de página y grafique tamaño de matriz vs. Marcos vs. Porcentaje de hits. La gráfica al final del enunciado ilustra el tipo de gráfica que buscamos. Genere gráficas que muestren los datos para diferentes tamaños de matrices.

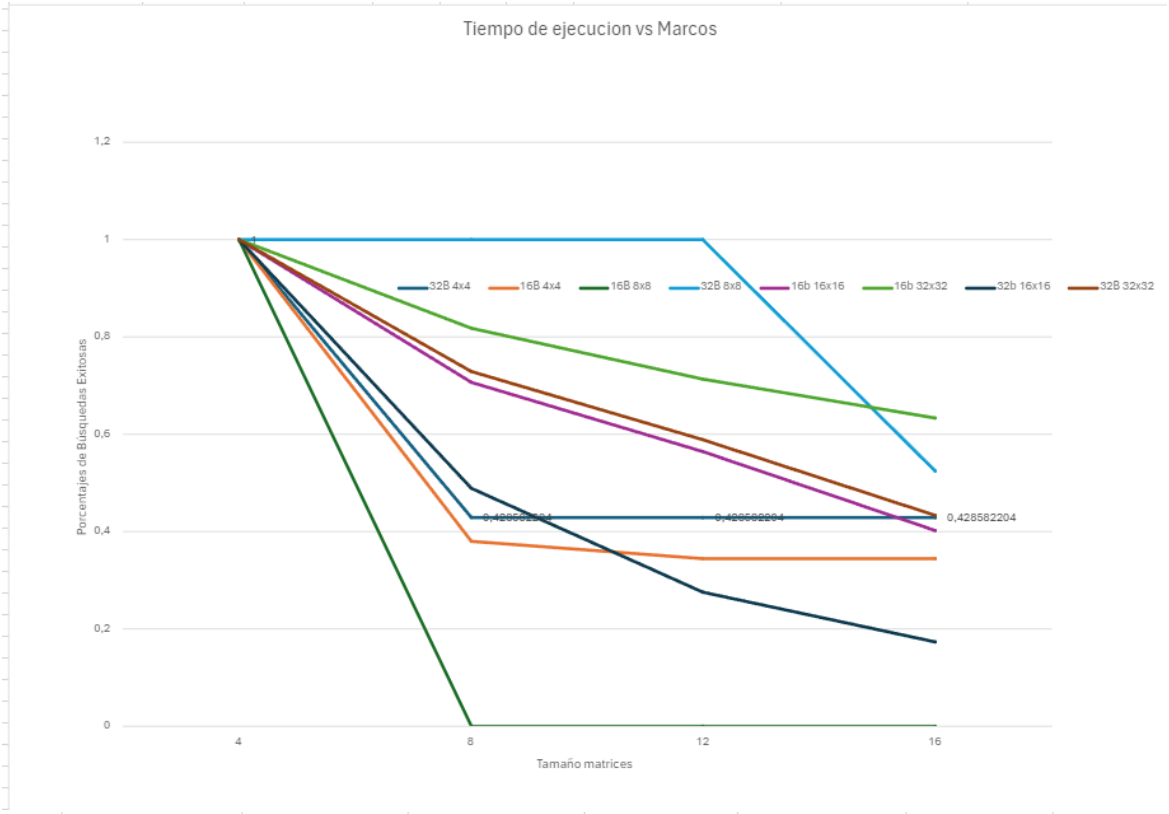


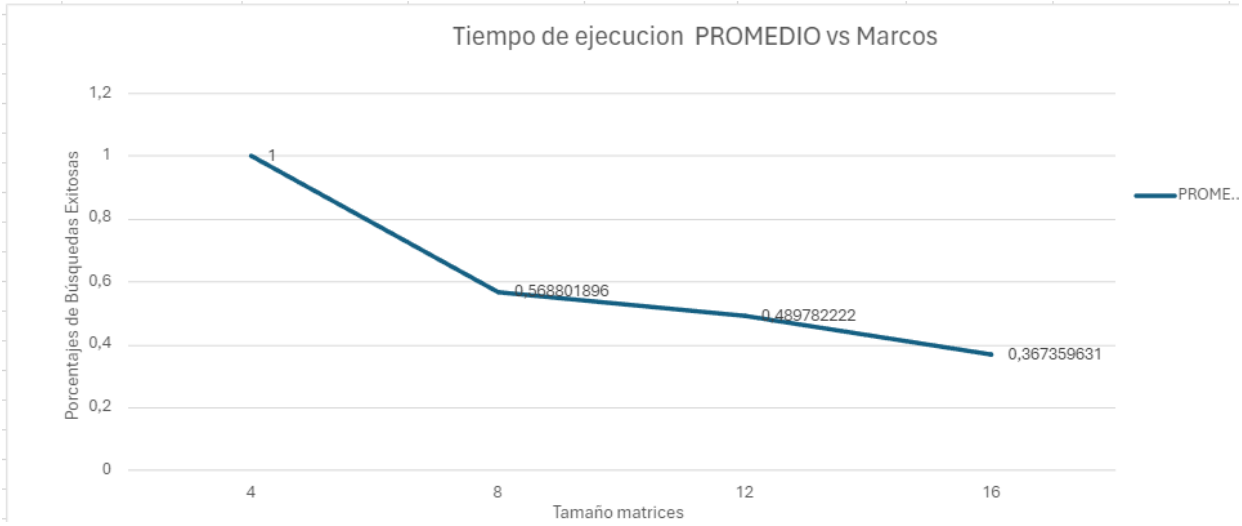


Se puede observar por las gráficas que en cuanto aumenta el número de marcos también aumenta el porcentaje de éxito. Sin embargo, a medida que aumenta el tamaño de las matrices disminuye el tamaño de porcentaje de éxito.

6. Adicione las gráficas de tiempo.

Las gráficas presentadas exhiben los tiempos asociados con cada tabla. Para asegurar una comparación equitativa y facilitar la visualización de las diferencias y similitudes entre ellas, los tiempos de cada tabla han sido normalizados. Esto se logró dividiendo los tiempos registrados en cada tabla por el valor máximo encontrado en esa tabla específica. Como resultado de este proceso de normalización, todos los conjuntos de datos comienzan en 1. La primera grafica muestra esto para cada una de las tablas y la segunda grafica es el tiempo promedio de ejecución normalizado. Podemos ver que a medida que aumentan los marcos de página el tiempo de ejecución baja. Esto se debe a que hay menos fallas.





7. Escriba su interpretación de los resultados: ¿tienen sentido? Justifique su respuesta.

Es claro que, si las páginas son más grandes, hay menos fallos de página porque hay más registros por página y menos páginas en uso, por lo que se jugaría con menos índices en la tabla de páginas para asignarle a los marcos.

También es visible que mientras más grandes son las matrices con las que se trabaja, más registros hay, evidentemente porque hay más datos. Luego si el número de registros aumenta, pero los marcos de página se mantienen iguales, se obtienen menos hits de páginas, esto se debe a que, si aumentan los registros, se aumenta el número de páginas a las que son destinadas un mismo número de bits.

Finalmente se tiene que si el número de marcos de página aumenta y los demás datos se mantienen constantes, aumenta el número de hits, pero no aumenta de manera lineal, por lo que aumenta bien

hasta algún punto y empezara a aumentar muy poco a partir de ahi, se debe encontrar un punto de equilibrio para implementar en las maquinas que se usan