Gabriela Vargas - 202013830

Mariana Díaz - 202020993

Juan Nicolás García Esquibel – 201717860

Caso 2

Estructuras de datos utilizadas/ Esquema de sincronización:

La solución del caso propuesto consiste en 6 clases, donde se logró obtener los cálculos solicitados y simular el comportamiento de un sistema que resuelve direcciones virtuales y el proceso de cargar las páginas en memoria RAM, teniendo en cuenta los fallos de página y el tiempo que toma esta ejecución. Se entrará en detalle para cada una, es decir, se explicará su respectivo funcionamiento y estructuras, pero las clases son las siguientes: TLB, TP, ThreadAging, ThreadLoader, RAM y App.

La primera corresponde al Translation Lookaside Buffer, al cual se le determina un tamaño, un HashMap de referencias y un ArrayList para llevar control de una cola ordenada. La razón del HashMap se basa en cómo se asemeja en su estructura y funcionamiento al TLB, debido que se tiene una referencia, en este caso, siendo la Key, y el Value es lo que llamamos marcoRAM. Luego, tenemos dos métodos sincronizados, ya que se debe controlar el acceso a ellos (en la clase RAM se entenderá más) y estos son: agregarReferencia, que como su nombre lo indica añade al hashmap una entrada dada su referencia y el marco de RAM, aunque si la TLB ya está en su límite, primero, debe eliminar la primera página en la cola, considerando que es FIFO; y el otro método, eliminarReferencia, que se encarga precisamente de eso dada una referencia como parámetro.

A continuación, la tabla de Páginas (TP), de igual manera, cuenta con un HashMap de referencias y un objeto para la TLB. También cuenta con dos métodos synchronized. Por un lado, buscarReferencia, que lo hace a partir del parámetro referencia y devuelve su respuesta, y, por otro lado, agregarReferencia, que cuando cumple cierta condición, inicialmente tiene que eliminar la página del marco menos usado y al terminar esto, se agrega la nueva referencia a tal marco y de igual manera, se agrega a la TLB.

Previo a aclarar la clase de RAM, es importante referirse a ThreadAging, que tal como se indica en su nombre, extiende de Thread, tiene una instancia de RAM y la función en su método run es ejecutar el algoritmo de envejecimiento cada milisegundo (simulado con un sleep). Adicionalmente, está ThreadLoader, así mismo tiene su instancia de RAM y una ArrayList de referencias, y su propósito es cargar las referencias y actualizar el estado de la RAM, TLB y la tabla de páginas cada 2 milisegundos y esta información la imprime en pantalla. Llegado a este punto, RAM tiene atributos para el número de marcos, marco menos usado, nueva referencia, ArrayList diferentes para marcos disponibles y ocupados, instancias de TLB y la TP, un HashMap para bits de marcos y variables llamadas direcciones, datos y numFallas, con sus respectivos métodos get. Su método para el algoritmo de envejecimiento es synchronized, como ya se vió anteriormente, este es llamado por el ThreadAging y como comparte datos con el otro Thread, tal como el marco menos usado, esto lo obliga a ser sincronizado. En resumen,

el algoritmo de envejecimiento va mirando cada marco y preguntándole su número de bits, almacenando el menor, con el objetivo de tener establecido el marco menos usado.

Por último, el método cargarReferencia, que se llama por el ThreadLoader, que se crea con un Array que va cargando en cada iteración y este se genera a partir de la lectyra del archivo, y también se requiere que sea sincronizado, este es el encargado de verificar si la referencia está en la TP, entonces si no la encuentra, aumenta el número de fallas, luego comprueba si hay marcos disponibles, si no hay, se agrega la referencia en el marco menos usado y se hace el set de esa nueva referencia. Por el contrario, cuando hay, se solicita un marco a la RAM, se actualizan y finalmente, a la tabla de páginas se le agrega. Se debe tener en cuenta que, si la referencia ya está en la TP, se busca el marco y se agrega en la TLB.

Tabla con los datos recopilados:

Tiempos de Carga Datos Proceso con Localidad Alta				
	TLB_4	TLB_8	TLB_16	TLB_32
8	2530023130	2530023130	2530023130	2530023130
16	1980024780	1970024810	2000024720	2010024690
32	1170027210	1210027090	1210027090	1170027210

Los datos obtenidos para esta tabla estuvieron en un margen de error menor al 3% con respecto a los valores de referencia

Tiempos de Carga Datos Proceso con Localidad Baja				
	TLB_4	TLB_8	TLB_16	TLB_32
8	6620010860	6640010800	6640010800	6630010830
16	5270014910	5270014910	5300014820	5280014880
32	2750022470	2710022590	2760022440	2680022890

Los datos obtenidos para esta tabla estuvieron en un margen de error menor al 2% con respecto a los valores de referencia

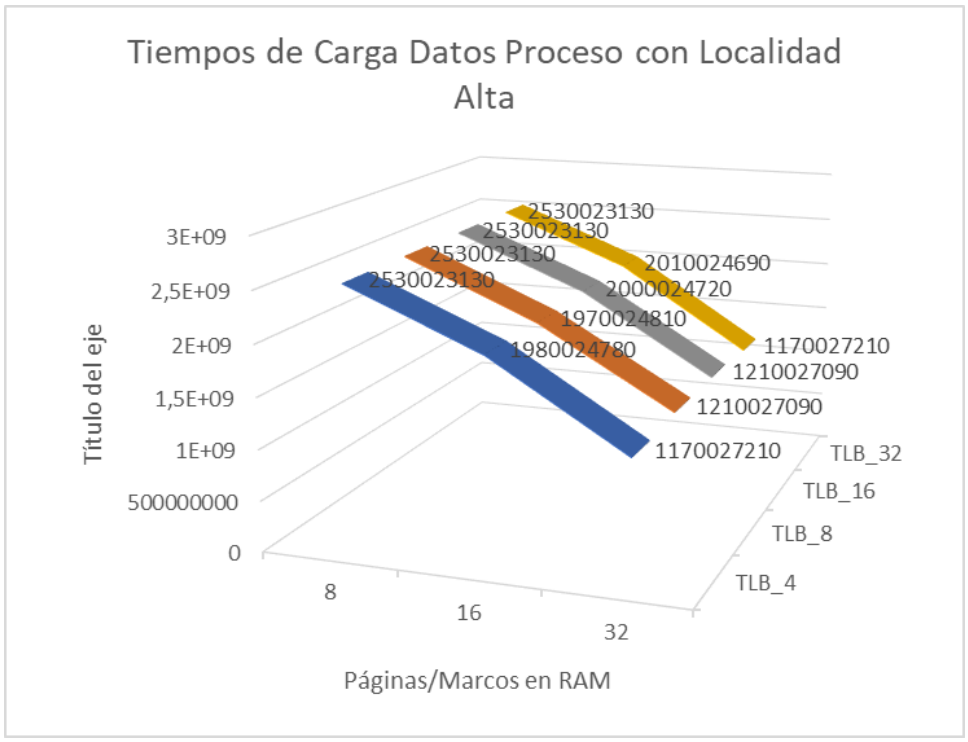
Tiempos Traducción Dirección Virtual Localidad Alta				
	TLB_4	TLB_8	TLB_16	TLB_32
8	19634	17070	16664	16838
16	17418	14372	13706	13590
32	15824	12450	10236	9298

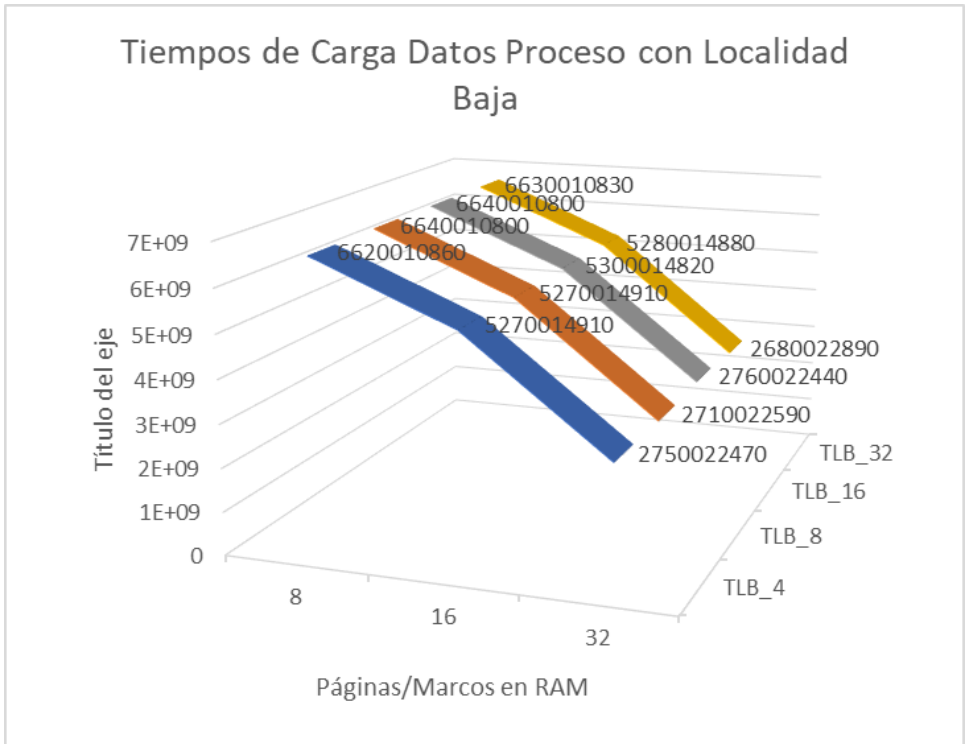
Los datos obtenidos para esta tabla estuvieron en un margen de error menor al 5% con respecto a los valores de referencia

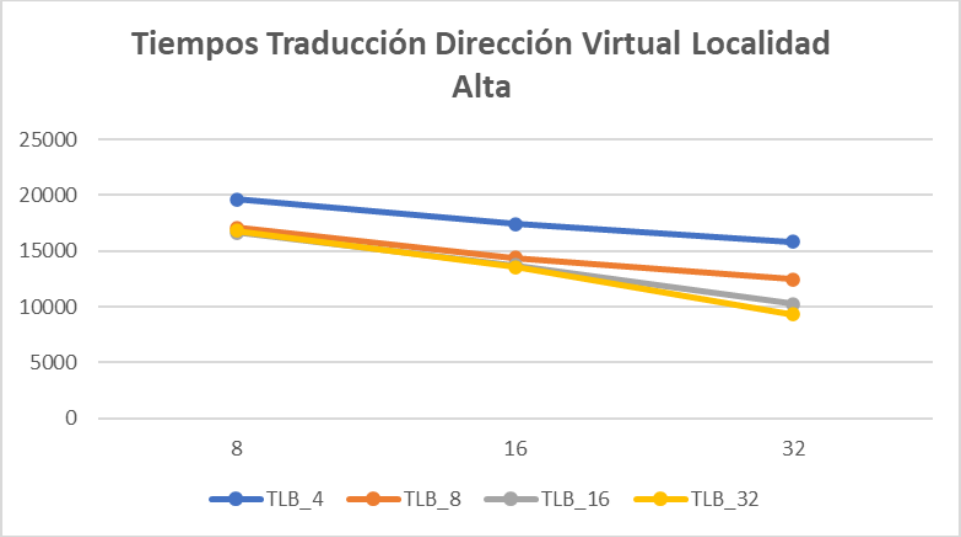
Tiempos Traducción Dirección Virtual Localidad Baja				
	TLB_4	TLB_8	TLB_16	TLB_32
8	41846	40560	40560	40560
16	38582	34320	32904	32672
32	30840	23360	21040	17882

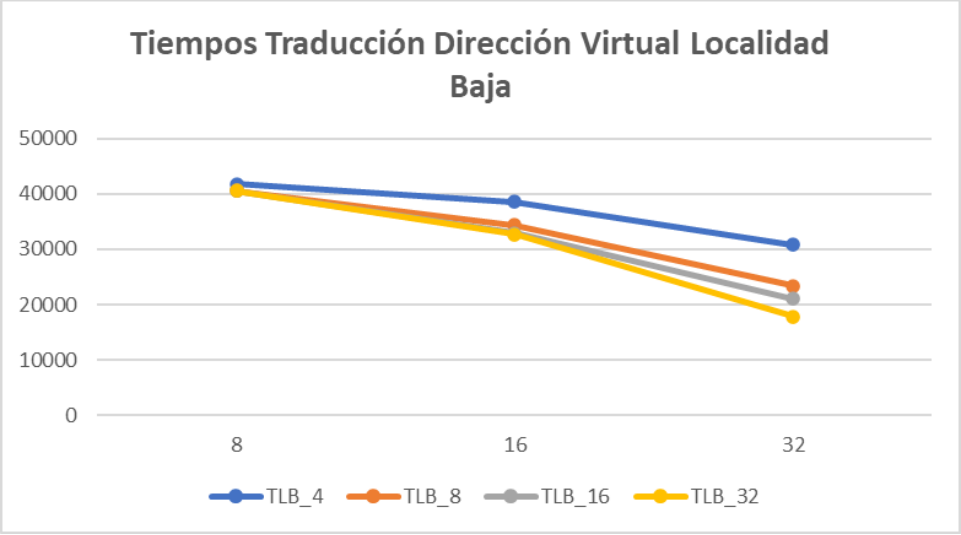
Los datos obtenidos para esta tabla estuvieron en un margen de error menor al 15% con respecto a los valores de referencia

Gráficas que ilustran el comportamiento del sistema:









Interpretación de resultados:

Para llegar a estos resultados, se usó como valores de entrada los siguientes: TLB: 4,8, 16, 32; MP: 8, 16, 32; Nombre de archivo, ej_Alta_64paginas.txt, ej_Baja_64 paginas.txt. Con esto, se tabuló por cada uno para lograr graficar satisfactoriamente cada uno de los casos. Al ver los resultados, podemos concluir que los resultados arrojados no fueron muy similares a los resultados que se encontraban en el pdf del caso, sin embargo, debido a la concurrencia, la variación que hubo fue de un porcentaje bajo. Por otro lado, también se pudo ver que tienen sentido ya que entre mayor marcos de página, existirán menos fallos debido a qué hay una mayor capacidad de almacenamiento lo que para la RAM es mucho mejor debido a que el tiempo y el uso de recursos se ve optimizado.