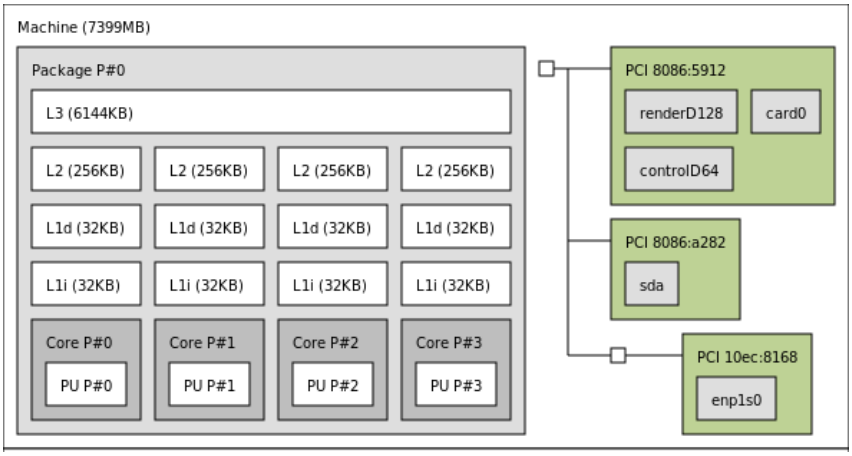
David Palomo Marcos, Pablo Sánchez Redondo
G1362 - Pareja 3

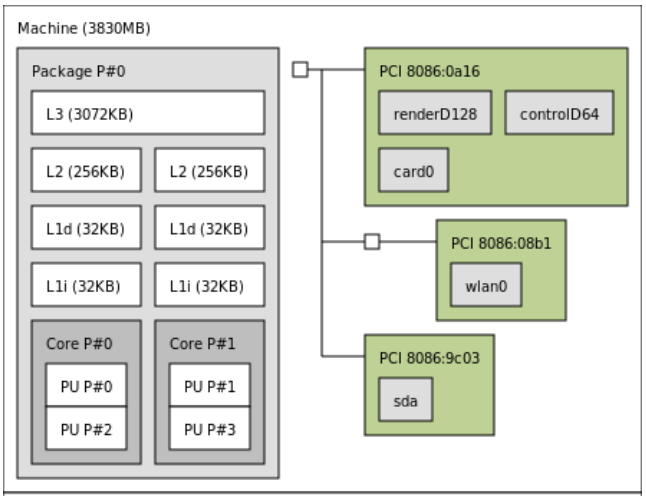
[ARQO – PRÁCTICA 3]

EJ-0: Información sobre la caché del sistema



Como podemos ver, en los ordenadores de los laboratorios existen:
- Una caché de instrucciones de nivel 1 (L1i) de 32 KB
- Una caché de datos de nivel 1 (L1d) de 32 KB.
- Una caché de nivel 2 (L2) de 256 KB.
- Una caché de nivel 3 (L3) de 6 MB.

Estas memorias caché son asociativas. En el nivel 1 hay distinción entre instrucciones y datos. Hemos realizado el mismo análisis en nuestro ordenador portátil, con los siguientes resultados:

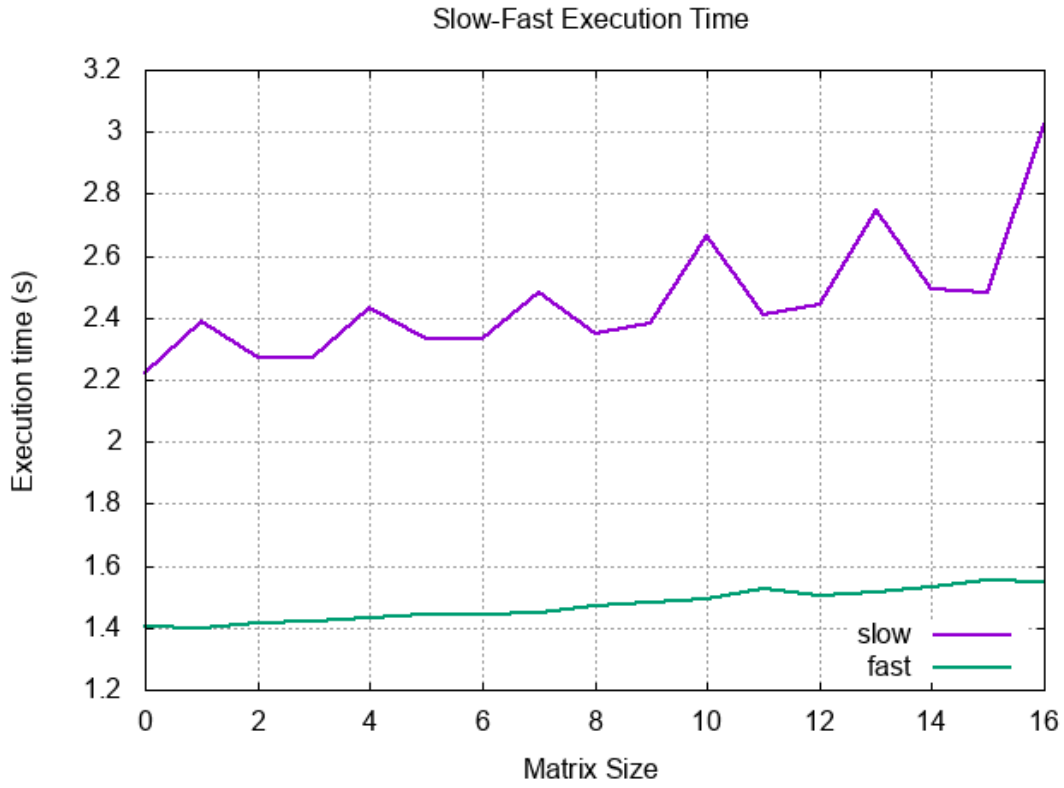


Las cachés del portátil son parecidas, pero la caché de nivel 3 es de 3 MB en lugar de 6 MB.

EJ-1: Memoria caché y rendimiento

La toma de medidas se realiza varias veces para lograr mayor precisión y limpieza, y para evitar los efectos que puedan producir otros procesos que vayan intercalados con el nuestro.

En nuestro caso, utilizamos 12 repeticiones y después calculamos la media de los datos obtenidos. La ejecución genera los siguientes resultados:

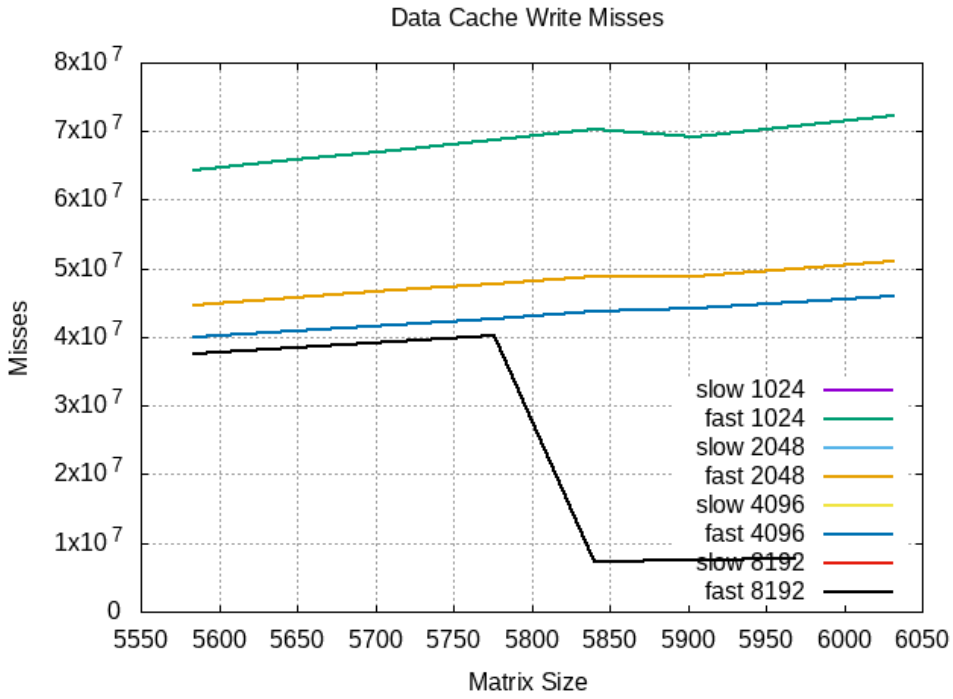


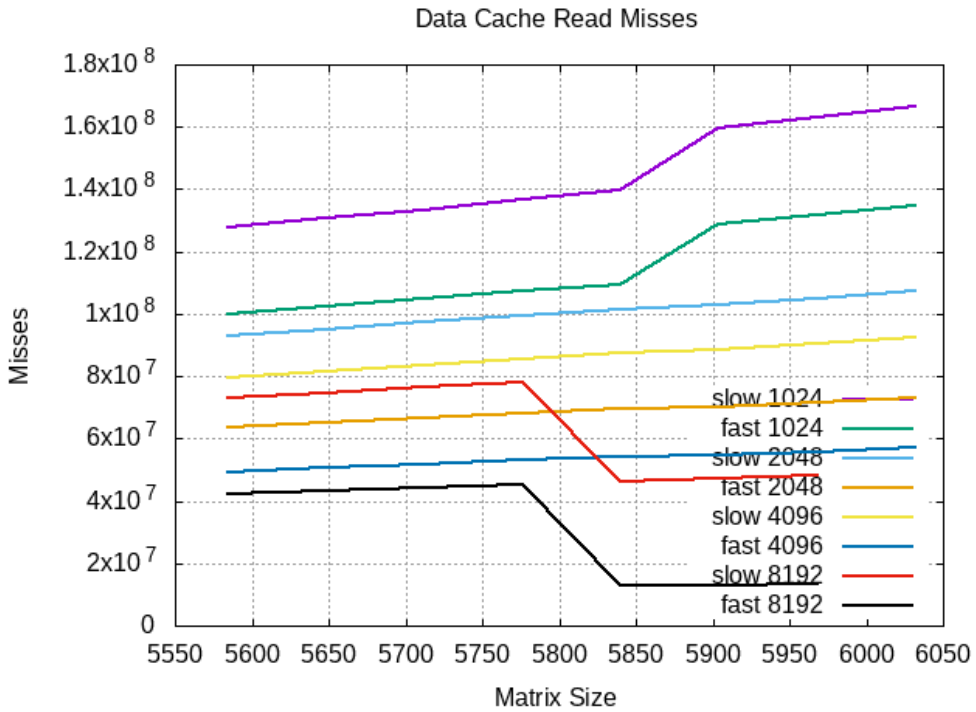
Como esperábamos, el programa “slow” tarda más tiempo que el programa “fast”: aproximadamente el doble. Sin embargo, en nuestra gráfica no se observa el efecto que menciona el enunciado, ya que para tamaños pequeños los tiempos de los programas no están próximos. Es posible que sea debido a los rangos utilizados.

La matriz se guarda en memoria por filas, es decir, que los elementos de la misma fila de la matriz se encuentran consecutivos en memoria. Esto da ventaja al programa “fast”, que puede acceder a los datos que necesita sin cambiar el bloque cargado.

EJ-2: Tamaño de la caché y rendimiento

Las gráficas obtenidas representan los fallos de lectura y de escritura de datos:





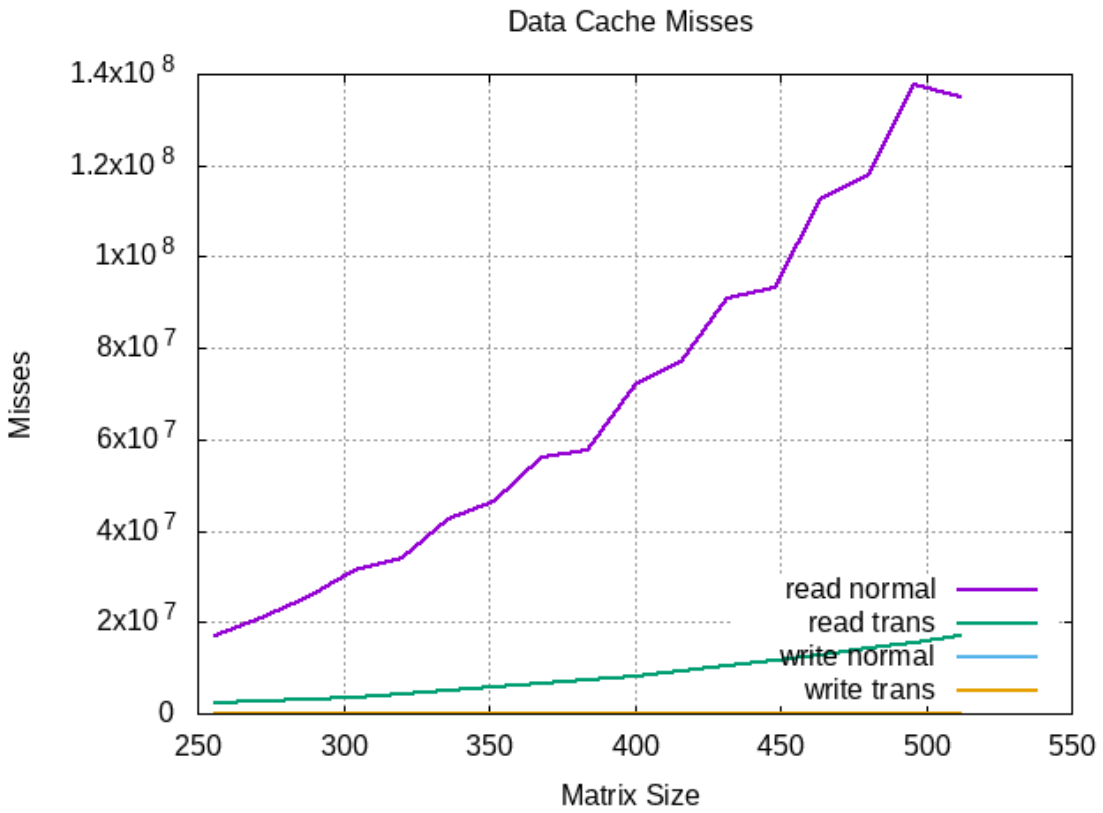
En la gráfica de escritura observamos que los fallos disminuyen cuanto mayor es el tamaño de la caché, pero para un mismo tamaño, “slow” y “fast” tienen los mismos fallos. Es lógico, ya que los dos escriben el mismo número de veces (una por elemento de la matriz).

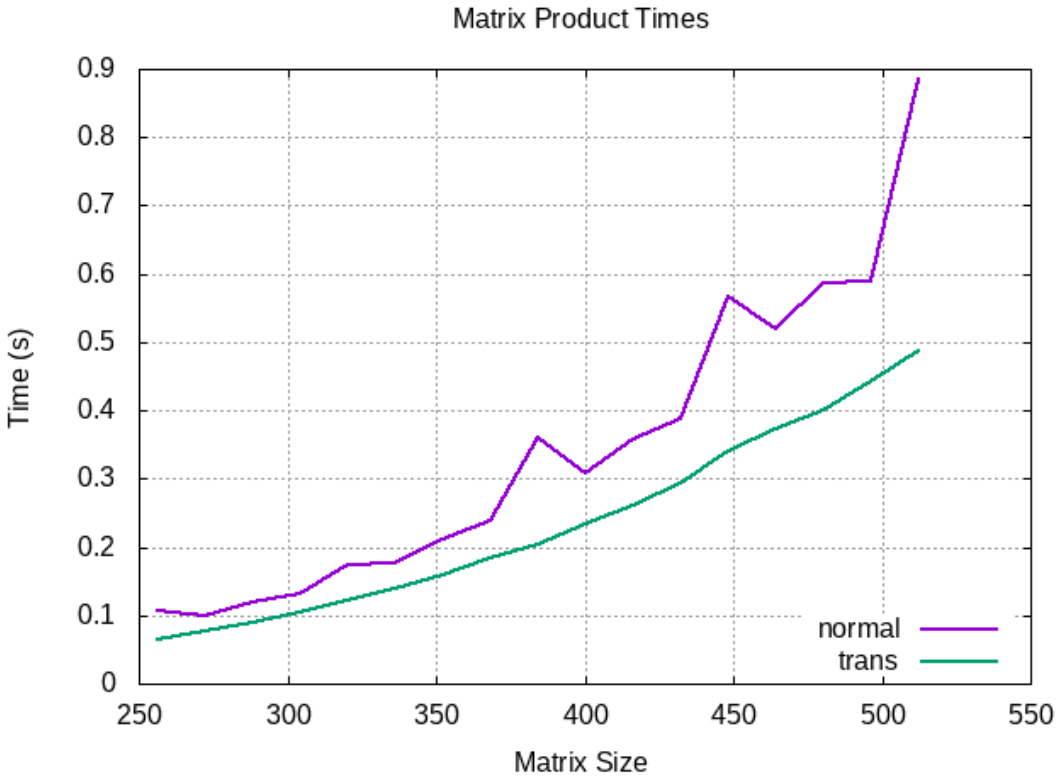
Analizando la gráfica de lectura, vemos que cuanto mayor es el tamaño de la caché, menor es el número de fallos en ambos programas, ya que caben más bloques simultáneamente en la caché y se reemplazan menos.

Como es lógico, el número de fallos aumenta con el tamaño de las matrices, y para el mismo tamaño de caché, el programa “slow” produce más fallos de lectura que “fast”,

Como ya se dijo antes, esto es debido a que “fast” (que recorre las matrices por filas) carga un bloque en caché y, como los próximos datos que necesita están en ese mismo bloque, no tiene que cargar otro hasta que lo haya leído entero. En cambio, “slow” carga un bloque distinto por cada dato leído de la columna.

EJ-3: Caché y multiplicación de matrices





En primer lugar, destacar que aunque hemos intentado mostrar la ejecución del apartado para los valores especificados, no hemos logrado que el script termine con éxito (posiblemente por timeout). Los resultados harían más evidente la diferencia entre ambos tipos de multiplicación (crecimiento exponencial), pero con valores más pequeños también se aprecia diferencia.

Procediendo al análisis, en la gráfica de fallos de caché (página anterior) observamos que se dan los mismos efectos que en el apartado anterior pero a una escala superior. Multiplicar matrices implica realizar más accesos a memoria que sumar sus elementos. Los fallos de escritura vuelven a ser los mismos para las dos versiones (en ambas se escribe una vez por cada elemento de la matriz resultado). Se observa claramente que la multiplicación traspuesta ofrece resultados muy superiores a los de la multiplicación normal.

Los tiempos reflejan lo mismo: tarda menos la multiplicación traspuesta que la normal. El motivo es una vez más la diferencia entre el recorrido por filas y por columnas. La multiplicación de matrices normal recorre una matriz por filas y otra por columnas, sumando los productos de los elementos, y al realizar recorrido por columnas, no es eficiente. En cambio, al hacer la traspuesta estamos eliminando la necesidad de recorrer la segunda matriz por columnas: ambas matrices se recorren por filas, con lo que se reduce el tiempo necesario y los fallos de caché.