Práctica3: Memoria caché y rendimiento

**Ejercicio 0: información sobre la caché del sistema.**

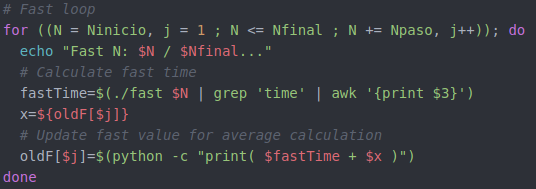


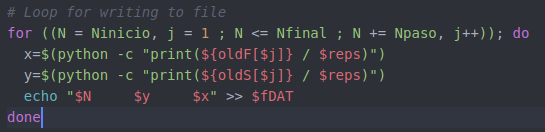
En esta primera imagen podemos ver que la cache de los equipos docente posee 3 niveles de cache (aunque aparezcan 4 se ve esa no existe, pues su tamaño y todo es 0). A su vez, el nivel 1 esta dividido en 2, I y D. el atributo SIZE nos indica el tamaño de cada nivel, ASSOC nos indica que es asociativa, y LINESIZE nos muestra el tamaño de cada línea de la cache.

|  |  |
| --- | --- |
|  | Con este comando podemos observar también que hay 3 niveles de cache, pero de esta manera se ve que cada Core tiene cierta porción de cada nivel menos el nivel 3, que es común para todos los Cores. |

**Ejercicio 1: Memoria caché y rendimiento**

Para obtener los datos de este ejercicio, creamos un script en bash que primero reserva un array de 0s en memoria para los tiempos de fast y de slow, de tamaño 16 (para los 16 números entre 16144 y 17168 con pasos de 64). Luego creamos un bucle para iterar 20 veces sobre los bucles que calculan los tiempos fast y slow con la N correspondiente, y los suma al valor actual del array de tiempos en la posición correspondiente. Las sumas están hechas con una linea de código python. (Ejemplo abajo)

 Hacemos la media de cada uno de los N tiempos con otro comando python. (Ejemplo abajo)

 Por ultimo, mandamos los datos a GNUPlot para sacar las graficas de los tiempos (See slow\_fast\_time.png)

La grafica (slow\_fast\_time.png) tiene ese aspecto porque como en el slow

Además de la gráfica mencionada, se pide en la memoria explicar el método seguido para la obtención de los datos, y una justificación del efecto observado. ¿Por qué para matrices pequeñas los tiempos de ejecución de ambas versiones son similares, pero se separan según aumenta el tamaño de la matriz?¿Cómo se guarda la matriz en memoria, por filas (los elementos de una filasestán consecutivos) o bien por columnas (los elementos de una columna son consecutivos)?