***Práctica 3 – Caché Blocking***

**9. El siguiente ejemplo permite comprender mejor donde se encuentran los fallos de memoria caché cuando se usa la versión naïve del producto de matrices. Sea el producto de dos matrices de tamaño 3x3, donde la matriz A se encuentra almacenada a partir de la dirección 1000h, la matriz B de la dirección 2000h y la matriz C de 3000h. Además, el sistema de memoria donde se ejecuta el código tiene una memoria cache para datos, con 4 bloques de 2 palabras cada uno de ellos, y organización asociativa por conjuntos de 2 vías. Contesta las siguientes preguntas:**

**a) Obtén la traza de referencias de memoria generadas al ejecutar el código usando el Listado 2, considerando que los accesos de escritura se comportan igual que los de lectura, y completa la siguiente tabla con la evolución de la memoria caché (añade todas las líneas que sean necesarias):**

**b) Calcula la tasa de fallos al acceder a las tres matrices, y las tasas de fallo particulares para A, B y C. Explica razonadamente porque son distintas.**

**10. Compila el código incluido en las carpetas Naive y Blocking, con la opción –O3 dentro del Makefile para que el compilador optimice el código al máximo. A continuación, usa perf para rellenar la tabla de eventos hardware y calcula los factores de mejora. Anota también el tiempo que tarda cada versión.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Evento** | **Naive** | **Blocking** | **Factor** |
| Branches | 18.852.072 | 18.718.028 | 1.007% |
| Branch-misses | 141.265 | 114.296 | 1.235% |
| Bus-cycles | 203.793.476 | 73.319.054 | 2.779% |
| Cache-misses | 17.001.604 | 947.382 | 17.945% |
| Cache-references | 40.479.704 | 42.410.513 | 0.951% |
| Cpu-cycles | 304.728.841 | 144.020.222 | 2.115% |
| Instructions | 118.769.868 | 126.052.704 | 0.942% |
| Tiempo | 3.662358e-01 | 1.3380255e-01 | 2.737% |

**11. ¿Cuáles son los eventos más representativos de la mejora?**

**12. Aparte de los eventos anteriores, hay otros relacionados con la caché como por ejemplo L1-dcache-loads, LLC-loads, dTLB-load-misses, etc. Estos eventos permiten determinar con más exactitud que está ocurriendo en los diferentes niveles del sistema de memoria. Anota los valores de todos los eventos relacionados con la caché de datos, para las versiones naïve y cache blocking, y justifica razonadamente su variación.**

**13. Cambia el factor de BLOCKSIZE y vuelve a compilar y ejecutar el programa. Prueba con todos los múltiplos de 2 entre 1 y 32 y representa gráficamente el tiempo de ejecución (o el factor de mejora) con respecto al factor de blocking. Explica razonadamente qué ocurre1.**

**14. Ejecuta ahora cada versión con diferentes tamaños de matrices y representa gráficamente la mejora (aceleración) obtenida con respecto al tamaño de la matriz. Puedes probar con tamaños de 64, 128, 256, 512 y 1024.**

**15. Para los distintos tamaños de matrices y factores de blocking, anota los eventos hardware relacionados con la caché y comprueba como varían.**

**16. Recupera la gráfica del ejercicio 7 (el modelo Roofline) y situa esta nueva versión en ella. Ten en cuenta que la intensidad operacional ha podido variar por lo que tienes que volver a calcularla, aparte de su rendimiento.**