Práctica 5

Coherencia de caché y medidas del rendimiento

1 Introducción

El objetivo de esta práctica es comprobar los efectos de la memoria caché sobre la ejecución de un programa haciendo uso de los contadores internos de un procesador.

2 Efecto de la longitud de los datos en la caché

El tamaño de las cachés, el tipo de política de reemplazo, la longitud de la línea de caché, etc., son factores que pueden tener una gran influencia en la velocidad de ejecución de los algoritmos según el tamaño de los datos (matrices o vectores) y su disposición en memoria.

Lo que se pretende en este apartado es comprobar la relación que existe (a veces son relaciones *patológicas*) entre la longitud de los datos y el comportamiento de la caché, y como consecuencia de ello, su impacto en el rendimiento del sistema.

Para poder medir tanto la velocidad de ejecución (que nos dará idea del rendimiento) así como los accesos a la caché (aciertos o *hits,* y fallos o *misses*) vamos a utilizar dos contadores del procesador utilizado (R10000) que contabilizan una serie de sucesos que han ido ocurriendo durante la ejecución del programa. Los sucesos que pueden ser detectados por cada contador son los que se exponen en la siguiente tabla:

Contador 1	Contador 2
0 = Cycles	16 = Cycles
1 = Issued instructions	17 = Graduated instructions
2 = Issued loads	18 = Graduated loads
3 = Issued stores	19 = Graduated stores
4 = Issued store conditionals	20 = Graduated store conditionals
5 = Failed store conditionals	21 = Graduated floating point instructions
6 = Decoded branches	22 = Quadwords written back from primary data cache
7 = Quadwords written back from scache	23 = TLB misses
8 = Correctable scache data array ECC errors	24 = Mispredicted branches
9 = Primary instruction cache misses	25 = Primary data cache misses
10 = Secondary instruction cache misses	26 = Secondary data cache misses
11 = Instruction misprediction from scache way prediction table	27 = Data misprediction from scache way prediction table
12 = External interventions	28 = External intervention hits in scache
13 = External invalidations	29 = External invalidation hits in scache
14 = Virtual coherency conditions	30 = Store/prefetch exclusive to clean block in scache
15 = Graduated instructions	31 = Store/prefetch exclusive to shared block in scache

Para poder utilizar estos contadores y medir así el rendimiento de nuestros programas, lo que haremos nosotros es incorporar en nuestra rutina en C, unas sencillas funciones que primero ponen estos contadores a cero (los dos que se quieran ver). Posteriormente se invocará el trozo de código a evaluar y por último, siempre dentro del propio programa, se leerán estos contadores para poder realizar estadísticas y gráficas (utilizando el gnuplot por ejemplo).

Estas funciones se explican solas en el siguiente programa de ejemplo donde simplemente se pretende medir los ciclos que tarda en ejecutarse un trozo de código en función de la longitud de los vectores que intervienen:

```
#define N     512   /* Longitud máxima que alcanzan los vectores */
#define E0      0   /* Primer grupo de contadores */
#define E1     26   /* Segundo grupo de contadores */

typedef long double tipo;

tipo *a,*b,*c;      /* Vectores de prueba */
int i,n;
long long c0,c1;    /* Para recoger la cuenta */

main()
{
 for (n=4;n<N;n+=4)    /* Aumenta de 4 en 4 */
 {
   a=(tipo*)calloc(n,sizeof(tipo));   /* Se crean los vectores cada vez */
   b=(tipo*)calloc(n,sizeof(tipo));
   c=(tipo*)calloc(n,sizeof(tipo));
   start_counters(E0,E1);             /* Inicializa contadores */
   for (i=0;i<n;i++)                  /* Rutina de prueba */
       c[i]=a[i]*b[i];
   read_counters(E0,&c0,E1,&c1);      /* Se leen los contadores */
   printf("\n%10d%12lld",n,c1);       /* Se escriben por stdout */
   free(a);
   free(b);
   free(c);
 }
}
```

La salida de este programa se puede redireccionar a un fichero de manera que se pueda ver la gráfica con **gnuplot**. Para ver la gráfica con gnuplot simplemente se llama al programa (gnuplot) y una vez dentro visualizamos el fichero con el comando plot 'fichero' donde lo que se pone dentro de las comillas simples es el nombre del fichero con los datos. Para realizar gráficas más elaboradas podéis consultar con **man gnuplot**.

Como las funciones `start_counters()` y `read_counters()` se encuentran en la librería libperfex, habrá que compilar con la opción **–lperfex** de **cc** para que funcione bien. Todo esto se realizará en las máquinas de los laboratorios basadas en el procesador R10k.

Lo que habrá que hacer en esta práctica es:

- Cambiar el programa anterior para detectar las variaciones en cuanto a tiempo de ejecución (contador 0) dependiendo de la longitud del vector. (Realizar la gráfica de tiempos en función del tamaño del vector).
- Cambiar de nuevo el programa para ver las variaciones en cuanto a fallos de caché tanto la primaria como la secundaria, dependiendo siempre de la longitud del vector. Eventualmente pueden hacer falta longitudes grandes, sobre todo para ver los fallos en la caché secundaria
- Realizar un programa que demuestre el efecto del *trashing* debido al reemplazo de mapeado directo.
- Se deben probar otros contadores para ver otras características del sistema. (Rendimiento en MIPS y MFLOPS, fallos de la caché, fallos de página, etc.)

El objetivo es realizar el mayor número de análisis sobre el procesador para obtener el mayor número de características del sistema con la ayuda de los contadores internos del procesador. Los resultados obtenidos habrá que anotarlos para luego razonarlos en la memoria que habrá que presentar antes del examen, sacando el mayor número de conclusiones sobre el sistema.

Algunas de las características que se puede intentar averiguar a partir de los resultados obtenidos son:

- Longitud en bytes de la línea de caché.
- Tamaño de la caché primaria.
- Tamaño de la caché secundaria.
- Averiguar si la caché es toda de mapeado directo o es asociativa por conjuntos, y en este segundo caso de cuantos conjuntos se dispone.

- Rendimiento de la máquina en MIPS y en MFLOPS.

Sobre algunos de estos parámetros es posible que no se puedan obtener valores fiables, pero es interesante realizar el análisis e intentarlo, seguro que al menos se alcanza un valor cercano al real. (Los valores reales se pueden obtener de las especificaciones de la máquina o del procesador, así se pueden comparar.)