Ejercicios de jerarquía de memoria

J. Daniel García Sánchez (coordinador)
David Expósito Singh
Javier García Blas
Óscar Pérez Alonso
J. Manuel Pérez Lobato

Arquitectura de Computadores
Grupo ARCOS
Departamento de Informática
Universidad Carlos III de Madrid

1. Ejercicios recomendados

Se recomienda la realización del los siguientes ejercicios del libro:
Fuente: Computer Architecture: A Quantitative Approach. 5 Ed
Hennessy and Patterson. Morgan Kaufmann. 2012.

- Apéndice B: B.1, B.2, B.3, B.4, B.5, B.6, B.7, B.8, B.9, B.10, B.11, B.12, B.13, B.14.
- Capítulo 2: 2.1, 2.2, 2.3, 2.8, 2.9, 2.10, 2.11, 2.12, 2.21, 2.22, 2.23.

2. Ejercicios de examen

Ejercicio 1 *Examen de junio de 2015.*

Suponga que dispone de un computador con 1 ciclo de reloj por instrucción (CPI) cuando todos los accesos a memoria están en la cache. Los únicos accesos a datos son load y store, y suman el 25 % del total de instrucciones. La penalización por fallo es de 50 ciclos de reloj y la tasa de fallo del 5 %.

Determine el speedup que se obtiene cuando no hay ningún fallo de caché con respecto al caso en que se producen fallos de caché. Asuma que todas las instrucciones se encuentran en memoria caché.

Ejercicio 2 *Examen de octubre de 2014.*

Sea la siguiente definición de variables globales:

```
const unsigned int max = 1024 * 1024;
double x[max];
double y[max];
double z[max];
double vx[max];
double vy[max];
double vz[max];
```

Y la siguiente función:

```
void actualiza_posiciones(double dt) {
  for (unsinged int i=0; i<max; ++i) {
    x[i] = vx[i] * dt + x[i];
    y[i] = vy[i] * dt + y[i];
    z[i] = vz[i] * dt + z[i];
  }
}
```

Suponga que se dispone de una caché L1 asociativa por conjuntos de 4 vías con un tamaño de 32 KB y un tamaño de línea de 64 bytes. La caché L2 es asociativa por conjuntos de 8 vías con un tamaño de 1 MB y un tamaño de línea de 64 bytes. La política de remplazo es LRU. Todas las cachés se encuentran inicialmente vacías.

Los arrays se almacenan de forma consecutiva en memoria y el primero ellos se encuentra en una dirección múltiplo de 1024. Asuma que el argumento de función **dt** y las variable índice **i** se encuentran asignadas a registros.

1. Determine la tasa de aciertos de las caché L1 y L2 para la ejecución de la función **actualiza_posiciones()**. ¿Cuál será la tasa global de aciertos?
2. Modifique el código aplicando la optimización de fusión de arrays.
3. Repita los cálculos del primer apartado para el código resultante del apartado segundo.
4. Si se asume que un acierto en la caché L1 requiere 4 ciclos, y en la caché L2 requiere 14 ciclos y la penalización por traer un bloque de memoria principal a la caché de nivel 2 es de 80 ciclos ¿cuál será el tiempo medio de acceso en cada uno de los dos casos?

Ejercicio 3 *Examen de octubre de 2014.*

Sea la siguiente definición de variables globales:

```
const unsigned int max = 1024 * 1024;
double x[max];
double y[max];
double z[max];
double vx[max];
double vy[max];
double vz[max];
```

Y la siguiente función:

```
void actualiza_posiciones(double dt) {
  for (unsinged int i=0; i<max; ++i) {
    x[i] = vx[i] * dt + x[i];
  }
  for (unsinged int i=0; i<max; ++i) {
    y[i] = vy[i] * dt + y[i];
  }
  for (unsinged int i=0; i<max; ++i) {
    z[i] = vz[i] * dt + z[i];
  }
}
```

Suponga que se se dispone de una caché L1 asociativa por conjuntos de 4 vías con un tamaño de 32 KB y un tamaño de línea de 64 bytes. La caché L2 es asociativa por conjuntos de 8 vías con un tamaño de 1 MB y un tamaño de línea de 64 bytes. La política de remplazo es LRU. Todas las cachés se encuentran inicialmente vacías.

Los arrays se almacenan de forma consecutiva en memoria y el primero ellos se encuentra en una dirección múltiplo de 1024. Asuma que el argumento de función **dt** y las variable índice **i** se encuentran asignadas a registros.

1. Determine la tasa de aciertos de las caché L1 y L2 para la ejecución de la función **actualiza_posiciones()**. ¿Cuál será la tasa global de aciertos?

2. Modifique el código aplicando la optimización de fusión de bucles.

3. Repita los cálculos del primer apartado para el código resultante del apartado segundo.

4. Si se asume que un acierto en la caché L1 requiere 4 ciclos, y en la caché L2 requiere 16 ciclos y la penalización por traer un bloque de memoria principal a la caché de nivel 2 es de 80 ciclos ¿cuál será el tiempo medio de acceso en cada uno de los dos casos?

Ejercicio 4 *Examen de octubre de 2013.*

Dado el siguiente fragmento de código:

```
struct particula {
  double x, y;
  double ax, ay;
  double vx, vy;
  double masa;
  double carga;
};

void mueve(particula p[], int n, double t) {
  for (int i=0; i<n; ++i) {
    p[i].x += p[i].vx * t + 0.5 *p[i].ax * t * t;
  }
  for (int i=0; i<n; ++i) {
    p[i].y += p[i].vy * t + 0.5 * p[i].ay * t * t;
  }
}
```

La función **mueve()** se ejecuta para un array de 1000 partículas, en un sistema con una memoria caché de L1 de datos de 32 KB asociativa por conjuntos de 4 vías y tamaño de bloque de 64 bytes. La caché se encuentra inicialmente vacía.

Asuma que el array p[] está alineado a una dirección múltiplo de 64. Asuma que los argumentos **n** y **t**, así como la variable de índice **i** se encuentran asignadas a registros.

Se pide determinar:

1. El número de fallos de cache.

2. La tasa media de fallos.

3. Proponga un código alternativo usando la técnica de fusión de bucles. ¿Cuál sería la nueva tasa de fallos?

4. Proponga un código alternativo reorganizando el código en múltiples arrays paralelos y sin fusión de bucles ¿Cuál sería la nueva tasa de fallos?

5. ¿Cree que es conveniente aplicar la fusión de bucles en el tercer apartado? Justifique su respuesta.

Ejercicio 5 *Examen de octubre de 2013.*

Dado el siguiente fragmento de código:

```
for (i=0; i<64; i++)
  for (j=0; j<1024; j=j+2 )
    v[i][j] = v[i][j] * v[i][j+1] + b[i]
```

Se ejecuta en una arquitectura con una cache de tamaño 256 KB, con tamaño de bloque 64 Bytes y de palabra de 8 Bytes en el que la memoria caché es totalmente asociativa y utiliza un algoritmo de reemplazo LRU. Siendo **v** y **b** matrices de números reales de 8 byte almacenadas por filas (estilo C) y tamaños 64×1024 (**v**), 64 (**b**). Se supondrá que las variables índice se almacenan en registros y que al empezar el código la caché está vacía.

Se pide determinar:

1. El número de fallos de cache.
2. La tasa media de fallos.
3. Razone si se produce alguna vez reemplazo de una línea de cache precargada. No es necesario identificar qué reemplazos hay sino razonar la existencia o no de reemplazos.
4. Razone si la técnica de optimización de intercambio de bucles mejoraría la tasa media de fallos de la caché (no es necesario calcularla).

Ejercicio 6 *Examen de octubre de 2013.*

Dado el siguiente fragmento de código:

```
int a[100];
int b[100];
for (i=0;i<100;i++)
  a[i]=a[i]+5;
for (i=0;i<100;i++){
  if (i>90)
    c=c+1;
  else
    b[i]=a[i]*3;
}
```

Se pide:

1. Describir las optimizaciones de caché del compilador que permitan mejorar el tiempo de acceso a la memoria para este código y reescríbalo de acuerdo a dichas optimizaciones.
2. Considere un computador con una memoria caché cuya capacidad de línea es 32 bytes. Inicialmente la caché está vacía. ¿Cuál es la tasa de fallos de la caché en la ejecución del programa optimizado obtenido en el apartado anterior? Considere sólo fallos forzosos de los vectores **a** y **b** (no existen fallos de capacidad o conflicto).
3. Considerar ahora un computador con una memoria multinivel L1 y L2 ambas unificadas. El tiempo de ciclo es de 1ns y CPI=1.3. Para la caché de nivel 1 se asume una tasa de fallos del 10 % y una penalización de 10 ns. Para la caché de nivel 2 la tasa de aciertos es del 95 % y la penalización de L2 es 80ns. Se asume un tiempo de acceso a la caché L1 de 1ns. Se pide calcular el tiempo de ejecución de un programa con CI instrucciones donde el 50 % de las instrucciones son de Lectura/Escritura.

Ejercicio 7 *Examen de junio de 2011.*

Sea el siguiente fragmento de código:

```
double a[256][256], b[256][256], c[256][256], d[256][256];
//...
for (int i=0;i<256;++i)
  for (int j=0;j<256;++j) {
    a[i][j] = b[i][j] + c[i][j]
  }
}
for (int i=0;i<256;++i)
  for (int j=0;j<256;++j) {
    d[i][j] = b[i][j] - c[i][j]
  }
}
```

Se desea ejecutar este código en un computador que tiene una caché de nivel 1 totalmente asociativa de 16KB y política de sustitución LRU con tamaño de línea de 64 bytes. Los fallos de caché de nivel 1 requieren 16 ciclos de reloj. Por otra parte la caché de nivel 2 siempre genera aciertos en este código.

Asuma que los fallos de escritura en la caché de nivel 1 se envían directamente un búfer de escritura y no generan ningún ciclo de espera.

Se pide

1. Determine la tasa de aciertos del segmento de código, asumiendo que las variables **i** y **j** se asignan a registros del procesador y que las matrices **a**, **b**, **c** y **d** se encuentran totalmente en la caché de nivel 2.

2. Determine el tiempo acceso a memoria asumiendo que los accesos a la caché de nivel 1 requieren un ciclo de reloj.

3. Proponga una transformación de código de las que puede generar un compilador para mejorar la tasa de aciertos, mostrando el código resultante en lenguaje C.

4. Determine la nueva tasa de aciertos y el tiempo medio de acceso a memoria resultantes.

Ejercicio 8 *Examen de mayo de 2011.*

Dada una arquitectura con dos niveles caché con las siguientes características:

Memoria	Tiempo de acceso (ns)	Tasa de aciertos
L1	2	0.8
L2	8	0.9
RAM	100	1

El ordenador ejecuta un programa que reside completamente en memoria (no hay accesos a disco). Se pide:

1. Asumiendo que el 100 % de los accesos a memoria son operaciones de escritura, calcular de forma justificada el tiempo medio de acceso a memoria para (a) una política de escritura inmediata (write-through) y (b) una política de post-escritura (write-back) en las memorias caché L1 y L2.

2. Considerando los dos niveles de memoria caché L1 y L2 como una única caché global se pide calcular el tiempo medio de acceso y la tasa de aciertos de esta caché global.

3. Dado el siguiente código:

```
for (i=0;i<1000;i=i+32){
  a[i]=a[i+8]+a[i+16];
}
```

El tamaño de cada entrada de a es de 8 bytes y el de bloque de 64 bytes. El índice del bucle se almacena en un registro del procesador. Se pide comentar razonablemente el efecto en el rendimiento del empleo de una técnica de caché multibanco que emplea 4 bancos. ¿Qué repercusión tendría el empleo de esta técnica en el tiempo de cada acceso y en el ancho de banda de la caché para este código?