Laboratorio Número 2

Arquitectura de Computadoras

Esp. Ing. Esteban Volentini (evolentini@herrera.unt.edu.ar)
Ing. Lucas Castillo (lecastillodelacroix@herrera.unt.edu.ar)

http://www.microprocesadores.unt.edu.ar/arqcom/

¿De qué se trataba este LAB?

- Evaluación de distintas configuraciones
  - Aritmética de cálculo de direcciones.
  - Ubicación de un bloque de memoria.
- Trazas de memoria
  - Relación entre el código ensamblador de un programa y los accesos a memoria que genera.
  - Generación de trazas.
- Tipos de fallos
  - Inevitables, Capacidad y Conflicto.
- Impacto en la performance de un programa

Planteo del problema

- Ejercicio del Segundo Parcial en 2008.
- Caché de datos
- Asociativo puro
- Cuatro bloques de cuatro palabras
- Algoritmo de reemplazo LRU
- Multiplicación de una matriz A por un vector B y guardando el resultado en un vector C

Planteo del problema

- Matriz A
  - Contiene 16 x 16 elementos de una palabra
  - Esta organizada por columnas en posiciones consecutivas de memoria
  - Ocupa 1024 bytes a partir de 0x10000000
- Vector B
  - Contiene 16 elementos de una palabra
  - Ocupa 64 bytes a partir de 0x10000400
- Vector C
  - Contiene 16 elementos de una palabra
  - Ocupa 64 bytes a partir de 0x10000440

Planteo del problema

- El código para la multiplicación es el siguiente:

```
for(i = 0; i <= 15; i++) {
    for(j = 0; j <= 15; j++) {
        r1 = A[i, j];
        r2 = B[j];
        r3 = C[i];
        r4 = r1 * r2;
        r5 = r3 + r4;
        C[i] = r5;
    }
}
```

Planteo de la solución

a) Calcular la tasa de fallos que producirá el programa en la caché. Calcule qué fracción de esta tasa se debe a fallos compulsivos, de capacidad y de conflicto.

- Analizar cuántos accesos a memoria tiene un ciclo de los lazos.
- Calcular la cantidad de fallos que generará cada uno de estos accesos y extrapolarlo según la cantidad de veces que se ejecuta.
- Clasificar los fallos.

La solución en detalle

```
for(i = 0; i <= 15; i++) {
    for(j = 0; j <= 15; j++) {
        r1 = A[i, j];
        r2 = B[j];
        r3 = C[i];
        r4 = r1 * r2;
        r5 = r3 + r4;
        C[i] = r5;
    }
}
```

- Observamos que cada uno de los lazos se ejecuta 16 veces.
- ¿Cuántos accesos se realizan en cada iteración del lazo interno del programa?

La solución en detalle

```
for(i = 0; i <= 15; i++) {
    for(j = 0; j <= 15; j++) {
        r1 = A[i, j];
        r2 = B[j];
        r3 = C[i];
        r4 = r1 * r2;
        r5 = r3 + r4;
        C[i] = r5;
    }
}
```

- Los accesos a C no dependen del lazo interno
- El segundo acceso del lazo nunca falla
- Un fallo cada 4 iteraciones del lazo externo
- Se producen 4 fallos en total

La solución en detalle

```
for(i = 0; i <= 15; i++) {
    for(j = 0; j <= 15; j++) {
        r1 = A[i, j];
        r2 = B[j];
        r3 = C[i];
        r4 = r1 * r2;
        r5 = r3 + r4;
        C[i] = r5;
    }
}
```

- Los accesos a B dependen del lazo interno
- Un fallo cada 4 iteraciones del lazo interno
- Se producen 64 fallos en total

La solución en detalle

```
for(i = 0; i <= 15; i++) {
    for(j = 0; j <= 15; j++) {
        r1 = A[i, j];
        r2 = B[j];
        r3 = C[i];
        r4 = r1 * r2;
        r5 = r3 + r4;
        C[i] = r5;
    }
}
```

- La matriz A esta ordenada por columnas

$a_{0,0}$	$a_{1,0}$	...	$a_{15,0}$	$a_{0,1}$	$a_{1,1}$	...	$a_{15,1}$

- Un fallo cada acceso a un elemento
- Se producen 256 fallos en total

La solución en detalle

```
for(i = 0; i <= 15; i++) {
    for(j = 0; j <= 15; j++) {
        r1 = A[i, j];
        r2 = B[j];
        r3 = C[i];
        r4 = r1 * r2;
        r5 = r3 + r4;
        C[i] = r5;
    }
}
```

1) i=0, j=0: $C_{0..3}$, $B_{0..3}$, $A_{0..3,0}$ Fallan A, B y C
2) i=0, j=1: $C_{0..3}$, $B_{0..3}$, $A_{0..3,1}$, $A_{0..3,0}$ Falla A
4) i=0, j=3: $C_{0..3}$, $B_{4..7}$, $A_{0..3,4}$, $A_{0..3,3}$ Fallan A y B
17) i=1, j=0: $C_{0..3}$, $B_{0..3}$, $A_{0..3,0}$, $A_{0..3,16}$ Fallan A y B

La solución en números

- Total de fallos
  $T_f = 4 + 64 + 256 = 324$
- Total de accesos
  $T_a = 16 * 16 * 4 = 1024$
- Tasa de fallos
  $t_f = \frac{T_f}{T_a} = \frac{324}{1024} = 31,64\,\%$

La solución en detalle

- Es un caché asociativo puro
- No hay fallos por conflicto
- Son inevitables 1 fallo cada 4 elementos
- En A son inevitables 64 de los 256 fallos
- En B son inevitables 4 de los 64 fallos
- En C son inevitables los 4 fallos

La solución en números

- Total de fallos inevitables
  $T_i = 64 + 4 + 4 = 72$
- Tasa de fallos inevitables
  $t_f = \frac{T_i}{T_f} = \frac{72}{324} = 22,22\,\%$
- Total de fallos por capacidad
  $T_c = (256 - 64) + (64 - 4) = 192 + 60 = 252$
- Tasa de fallos por capacidad
  $t_c = \frac{T_c}{T_f} = \frac{252}{324} = 77,77\,\%$

Planteo de la solución

b) Proponga una nueva forma de organizar los datos en memoria para reducir los fallos. Calcule nuevamente la tasa de fallos. Calcule qué fracción de esta tasa se debe a fallos compulsivos.

- Analizar una mejor forma de organizar los datos.
- Recalcular la tasa de fallos y clasificación de los mismos.

La solución en detalle

- Organizar la matriz A por filas
- Coincide con el recorrido del programa
- Solo 1 fallo cada 4 accesos
- Se producen 64 fallos en lugar de 256
- Son todos fallos inevitables

La solución en números

- Nuevo total de fallos
  $T'_f = 64 + 64 + 4 = 132$
- Nueva tasa de fallos
  $t'_f = \frac{T'_f}{T_a} = \frac{132}{1024} = 12,89\,\%$
- Nueva tasa de fallos inevitables
  $t_f = \frac{T_i}{T'_f} = \frac{72}{132} = 54,54\,\%$
- Nuevo total de fallos por capacidad
  $T'_c = (64 - 4) = 192 + 48 = 60$
- Nueva tasa de fallos por capacidad
  $t'_c = \frac{T'_c}{T'_f} = \frac{60}{132} = 45,45\,\%$

Planteo de la solución

c) Para la organización original de datos, proponga un cambio de programa para bajar la cantidad de fallos a un número similar a (b). Calcule nuevamente la tasa de fallos y la fracción de fallos inevitables.

- Cambiar el orden de los lazos para recorrer la matriz por filas y aprovechar su organización en memoria.
- Se obtienen los mismos resultados que en la solución del apartado (b).

Planteo de la solución

d) Si existiera la posibilidad de duplicar el tamaño del caché, indique qué convendría duplicar.
   - Cantidad de vías.
   - Cantidad de bloques.
   - Tamaño del bloque

   Justifique su respuesta.

- Ahora la mayor parte de los fallos son fallos inevitables
- La mejor alternativa es duplicar el tamaño del bloque.

Conclusiones Finales

- Había que analizar un código para saber que accesos generaban fallos en el caché.
- Hay que considerar la estructura del caché
- Influye la organización de los datos
- Y la implementación del algoritmo
- Clasificación de los tipos de fallos
- Criterio para disminuir los fallos
- En este laboratorio se mostraron cuestiones estructurales de los cachés

Conclusiones

- Próximas evaluaciones:
  - Practico 7 - Miércoles 6 de Junio.