

Temario-AC.pdf



almoga



Arquitectura de Computadores



2º Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación
Universidad de Granada

Les Roches

Despierta el líder que llevas
dentro en Les Roches.

Saber más →



WUOLAH

Fran será lo que quiera.



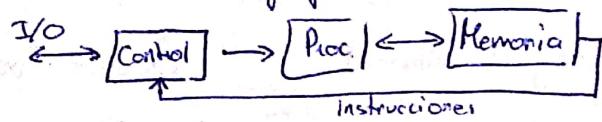
#NoTeApuntesAWuolah

Les Roches

Tema 1

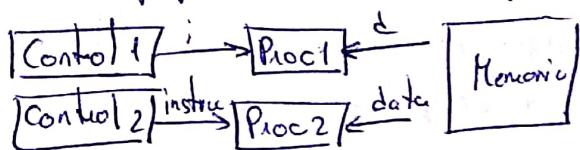
■ UMA, NUMA y to' esa polla.

Computadores SISD: Un único flujo de instrucciones y un único flujo de datos.



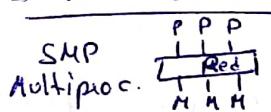
Computadores SIMD: Único flujo de instrucciones, varios de datos.

" MIMD: Varios flujos de instrucciones y de datos.

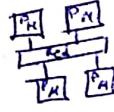


Computadores MIMD: Varios flujos de instrucciones, solo uno de datos. No existen

SMP Y CC-NUMA



Multi-computador



Diferencia SMP y CC-NUMA:

Los dos son multiprocesadores, es decir, comparten el espacio de direcciones físicas.

SMP la memoria se encuentra centralizada a igual distancia de todos los procesadores. Mientras que en CC-NUMA cada procesador tiene cerca un conjunto de sus direcciones, los módulos no están a igual distancia de todos los procesadores.

UMA Y NUMA

UMA → Uniform Memory Access. Se accede con la misma latencia desde cualquier procesador a memoria.

NUMA → Non Uniform Memory Access. Acceso no uniforme.

■ FÓRMULAS

| | | | | |
|------------|--|---------------------------|--|--|
| Tiempo CPU | $T_{CPU} = NI \cdot CPI \cdot T_{ciclo}$ | $T_{ciclo} = \frac{1}{f}$ | $CPI = \frac{Ciclos_proj}{NI} = \sum_i^n CPI \cdot I_i$ | $CPI \rightarrow \text{nº de ciclos}$ $IPE \rightarrow \text{nº instr. en un instante}$ |
|------------|--|---------------------------|--|--|

| | | | |
|-----------------------------------|---|--|--|
| $MIPS = \frac{f}{CPI \cdot 10^6}$ | $MIPS_{max} = \frac{f \cdot IPC_{max}}{10^6}$ | $MFLOPS = \frac{Op_FP}{T_{CPU} \cdot 10^6}$ | $MFLOPS_{max} = \frac{f \cdot (Op_FP/ciclo_{max})}{10^6}$ |
|-----------------------------------|---|--|--|

■ Ganancia Prestaciones

$$S = \frac{T_1}{T_p} = \frac{V_p}{V_1}$$

■ Ley de Amdahl

$$S_p < \frac{P}{1 + \frac{1-p}{f} \cdot (p-1)}$$

$f = \frac{\% \text{ de tiempo}}{\text{sin mejora}}$

■ Productividad

$$P(n) = \frac{1}{T_p}$$

■ Cuestiones Ley de Amdahl

- P : es el factor de incremento de prestaciones del recurso que se mejora.
- f : es la fracción de tiempo antes de la mejora en el que no se usa el recurso mejorado
- La máxima ganancia de velocidad que se puede conseguir, por mucho que se mejore el recurso es $\frac{1}{f}$.
- P : puede ser mayor que 1.

■ Ejercicios tablas

| Tipo instrucción | CPI | NI |
|----------------------------|--------|-----------------|
| LOAD, STORE, entradas, etc | ciclos | porcentaje o... |

* Si dice algo del reloj en el ejercicio hay que poner $T_{ciclo} = \frac{1}{f} \cdot T_{ciclo}$, en 0 .

A partir de esos datos, usar la fórmula oportuna

• En el de un circuito segmentado hacer

• $T = \text{mejor etapas} + \text{retardo}$

Ganancia $\Rightarrow S = \frac{T_{CPU}}{T_{min}}$ → esto lo dice el enunciado.

■ Dependencias

RAW: $a = b + c$ // b escribe en a
 $c = a + c$ // B_2 lee a

WAW: $a = b + c$ // b escribe a
 $c = d + e$ // B_2 escribe a

WAR: $b = a + 1$ // B_1 lee a
 $a = d + e$ // B_2 escribe a

B_1 = bloque

■ Ejercicios MIPS y MFLOPS

Haciendo uso de las fórmulas anteriores, teniendo en cuenta que para los MIPS necesitamos f = frecuencia

$$CPI = \text{nº CPI} \cdot NI \text{ de la tabla}$$

• Tener cuidado con los 10^6

Y para los MFLOPS, mira el código, las operaciones de coma flotante, si están en un bucle se multiplican por N y listo.

Tema 2

Ley de Gustafson

Ganancia de prestaciones. Con T paralelo constante $\rightarrow S(p) = p(1-f) + f$. Dónde f es la fracción de tiempo donde no hay paralelización.

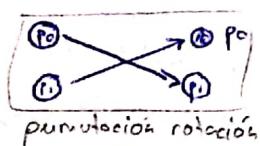
Cuanto mayor sea $1-f$, mayor será la escalabilidad.

Eficiencia $(p, n) = \frac{S(p, n)}{p}$, n es el tamaño del problema, p es el nº de procesadores.

Comunicación paralelización

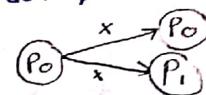
- Com. múltiple uno a uno: Componentes que envían un único mensaje y componentes que reciben un único mensaje.

Si todos envían y reciben se implementa una permutación.



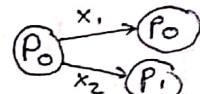
- Comunicación uno a todos: Un proceso envía y todos reciben.

- Difusión: Todos reciben el mismo mensaje



x = mensaje completo

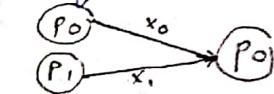
- Dispersion: Cada uno recibe un mensaje diferente



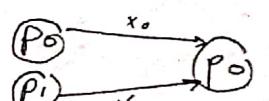
$x = (x_1, x_2)$

- Comunicación todos a uno: Todos los procesos envían mensaje a un único proceso.

- Reducción: Los mensajes se combinan en uno solo



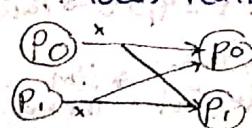
- Acumulación: Los mensajes se reciben uno tras otro.



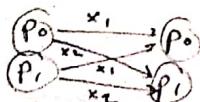
$x = (x_0, x_1)$

- Comunicación todos a todos: Todos realizan envío a todos.

- Todos difunden:



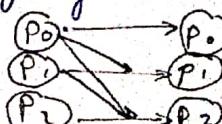
- Todos dispersan



- Todos combinan o reducción y extensión: Se aplica reducción a todos los procesos.

Barrera: Punto de sincronización donde un grupo tiene que alcanzar para continuar la ejecución.

- Recorrido: Todos envían un mensaje y reciben cada uno el resultado de reducir un conjunto de esos mensajes.



Les Roches

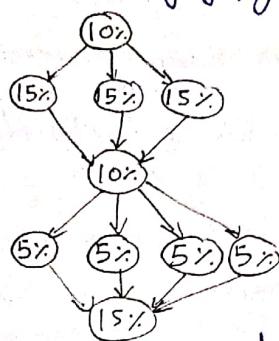
OpenMP y MPI

OpenMP hace uso de la programación, estableciendo la comunicación entre procesos mediante el acceso a variables compartidas.

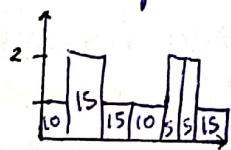
• MPI hace uso del envío de mensajes. Las funciones `send` y `receive` permiten enviar y recibir datos de forma síncrona o asíncrona en función de si `send` bloquea el acceso o no.

Ejercicios

⑤ Dado el grafo, y un $T_s = 60s$. Calcula el T_p y la ganancia:



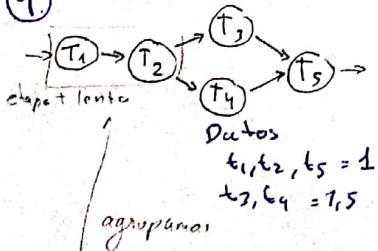
b) Para 2 procesadores



$$T_p = 0,1 + 0,15 + 0,15 + 0,1 + 0,05 + 0,05 + 0,15 = 0,75 \cdot 60 = 45s$$

$$S_{(2)} = \frac{60}{45} = 1,33$$

⑨



La mayor ganancia en prestaciones para:

a) 5 procesadores

$$T_s = 3t + 1,5 \cdot 2t = 6t \rightarrow \text{Para } n \text{ bloques} = 6nt$$

$$T_p = 3t + 1,5t = 4,5t \rightarrow \text{Para } n \text{ bloques} = 4,5nt$$

$$S_{(5)} = \frac{6nt}{3t + 1,5nt} \rightarrow \lim_{n \rightarrow +\infty} \frac{6nt}{3t + 1,5nt} = \frac{6}{3 + 1,5} = 4 //$$

b) Para 4 procesadores

$$T_s = 6nt$$

$$T_p = 4,5t + 2t(n-1) = \frac{2,5t + 2nt}{2,5t + 2nt} \text{ esto es lo interesante}$$

$$S_{(4)} = \frac{6nt}{2,5t + 2nt} \Rightarrow \lim_{n \rightarrow +\infty} \frac{6nt}{2,5t + 2nt} = \frac{6}{2} = 3 //$$

Los n bloques
se aplican a la
etapa más lenta, tanto
en el caso a) como en
el b).

Ganancias

- Lineal \rightarrow El grado de paralelismo debe ser ilimitado, siempre se debe poder dividir el código entre los p procesadores disponibles

$$S_p = \frac{T_{\text{secu}}}{T_{\text{para}}} = p$$

- Superlineal $\rightarrow (S_p > p)$ Se da en el caso de que al aumentar el $nº$ de procesadores aumentamos también otros recursos (cache, memoria, etc).

Tema 3

Conceptos

TMT \rightarrow temporal multithreading :

- Ejecutan varios threads en el mismo core a la vez.
- Gestión de threads por hardware
- Un thread por ciclo.

SMT \rightarrow Simultaneous Multithreading

- Varios threads en paralelo en un core superscalor
- Varios threads por ciclo.

Protocolos de gestión de cache

MSI : Un bloque de memoria puede tener 3 estados,

(M) Modificado \rightarrow La única copia válida del bloque en todo el sistema. La cache debe proporcionar el bloque si al espia el bus observa que algún componente la solicita en invalidado si algún nodo solicita una copia exclusiva para su modificación.

(S) Shared \rightarrow Todas las copias del bloque están actualizadas. La cache debe invalidar su copia si al espia el bus observa que algún nodo solicite una copia exclusiva del bloque para su invalidación.

(I) Invalido \rightarrow El bloque se ha invalidado o bien, no está en cache. Se encuentra en este estado si esto no es la última actualización del bloque.

MESI : Igual que MSI pero se añade un nuevo estado.

(E) Exclusivo: El paquete solo está en una cache, por lo que evitan invalidar bloques en otra cache para que no haya peligro de incoherencia.

■ Modelos de consistencia relajados.

Pueden permitir que en el código ejecutado en un procesador se relaje el orden entre dos accesos a distintas direcciones.

- Modelos que relajan $W \rightarrow R$: Permiten que las lecturas adelanten a las escrituras, pero evitando problemas de dependencias RAW.
- Modelos que relajan $W \rightarrow R$ y $W \rightarrow W$
- Que relajar cualquier ordenación: $W \rightarrow W$, $W \rightarrow R$, $R \rightarrow W$, $R \rightarrow R$.
 - de ordenación débil: se basa en mantener el orden entre accesos sólo en los puntos de sincronización del código
 - consistencia de liberación: distingue entre dos tipos de operaciones; adquisición y liberación.

■ Primitivas de sincronización

- Test & Set (x)

$\{$
temp = x
 $x = 1$
return temp; $\}$

Lee en una variable local el contenido actual de x .
Escribe un 1 en x .
Devuelve el valor de temp.

- Fetch & Operation (Add, Or...) (x, a)

$\{$
temp = x
 $x = x + a$ // o cualquier otra operación
return temp; $\}$

Lee el valor de x .
Realiza la operación.
Devuelve temp.

- Compare & Swap (a, b, x)

$\{$
if ($a == x$)
swap (x, b); $\}$

Si x coincide con a , intercambia x con b .

* Combinación

lock (a)

while (test & set (a) == 1) { $\}$

lock (a)

while (fetch & or ($a, 1 == 1$)) { $\{$
temp (temp); $\} lock (a);$

lock (a)

$b = 1$
do compare & swap ($0, b, a$);
while ($b == 1$); $\}$

■ Cerrajos, ejemplo

```
1 for (i = iThread; i < n; i += iThread)
2     sum += a[i];
3     lock (a); // Adquiere EH
4     sum += sum; // SC
5     unlock (a); // Libera EH
```

Les Roches

■ Entradas de un directorio

Ejercicio 1/ Multiprocesador NUMA con 16 nodos 4GB por nodo y líneas de caché de 64Bytes. ¿Cuántas entradas del dir de memoria?

$$4\text{GB} = 2^2 \cdot 2^{30}\text{B} = 2^{32}\text{B}$$

$$64\text{B} = 2^6\text{B}$$

$$\frac{2^{32}}{2^6} = 2^{26} \text{ entradas}$$

¿Cuántos bits tiene cada entrada?

$$16\text{ nodos} + 1\text{ bit estado} = 17\text{ bits}$$

■ Valores de un código El ejercicio 1 de la relación resuelto del tema 3 lo explica mejor

③ ¿Qué valores puede tomar R si no se respecta $W \rightarrow W$ e inicialmente $X=Y=0$?

$$\begin{array}{l|l} P_1 \{ & P_2 \} \\ \begin{array}{l} X=2; //a \\ Y=1; //b \end{array} & \begin{array}{l} R=1; //c \\ \text{if } (Y=1) \\ R=X; //d \end{array} \end{array}$$

Los posibles valores son 0, 1 y 2:

$$\cdot 0: a \rightarrow d \rightarrow c \rightarrow b$$

$$\cdot 1: c \rightarrow a \rightarrow d \rightarrow b$$

$$\cdot 2: a \rightarrow b \rightarrow c \rightarrow d$$

■ Parallelizar código

① a) Paralleliza el siguiente código de forma que se haga una asignación dinámica y se utilice Fetch & Add, teniendo en cuenta que no se garantiza $W \rightarrow R$.

for (i=0; i<100; i++)
 Código que usa i

\rightarrow
 n = Fetch & Add (i, 1);
 while (n < 100)
 código para n;
 n = Fetch & Add (i, 1);

Tema 4

■ Relacionado con los ej. del código máquina

• Una ventana de instrucciones es una cola de registros donde se almacenan las instrucciones que han sido decodificadas y están a la espera de ser emitidas.

- centralizada, si almacena las instrucciones pendientes

- distribuida, solo almacena de un tipo.

• Emisión alineada. No se introducen nuevas instrucciones en la ventana hasta que esté vacía.

• No alineada. Se pueden introducir nuevas si hay espacio disponible.

■ Superescalares y VLIW

Tanto un procesador superescalares como uno VLIW son procesadores segmentados que aprovechan el paralelismo entre instrucciones.

La diferencia principal es que mientras que el superescalares incluye elementos hardware para realizar la planificación de instrucciones dinámicamente, en el VLIW el paralelismo es ejecutado mediante software.

■ Buffer de renombramiento vs reordenamiento

- Renombramiento. Están en los superescalares y permiten asignar almacenamiento temporal a los datos. Así evitan dependencias entre las instrucciones.

- Reordenamiento. Permite implementar la finalización ordenada de las instrucciones después de su ejecución (ROB).

■ Ejercicios: predicción de saltos

⑧ Dado:

Penalizaciones si falla: penaliza con 5 ciclos.

a) Predicción fija (siempre se considera no)

$$P = 11 \times 5 = 55 \text{ ciclos de penalización.}$$

b) Predicción estática (si es negativo se toma)

$$P = 7 \times 5 = 35 \text{ ciclos}$$

c) Dinámica con 2 bits (11)

Predicción:

Ej: S S N N N S S N S N S N S S S S N
11 P P 1 P P P P P P P P 1 1 1 P

• Fallar en ejecución de tablas.

• Tablas con ROB.

• Código máquina.

• Entender mejor la predicción de saltos.