Arquitecturas Avanzadas

Solución al examen de Septiembre de 2001

Fernando Pardo, 24 de septiembre de 2001

Teoría

Las preguntas 2, 3, 5, 6 y 7 han sido explicadas en clase. Las respuestas que aquí se dan no tienen por qué recoger la totalidad de lo que se pregunta, sólo se han incluido aquellas partes que requerían pensar y no se encuentran de forma explícita en la teoría.

Pregunta 1

Para ver la escalabilidad de un sistema hay que calcular su Eficiencia en función del número de procesadores. La eficiencia *E(n)* es el factor de mejora (*speed-up*) *S(n)* dividido entre el número de procesadores *n*. Para calcular *S(n)* se calcula el tiempo de ejecución con un procesador *T(1)* y se divide por el tiempo de ejecución con varios procesadores *T(n)*. El cálculo de la eficiencia del sistema se hace de la siguiente manera:

$$T(1) = \Delta\left(\frac{1}{A} + C\right)$$

$$T(n) = \Delta\left(\frac{1}{nA} + nC\right)$$

$$S(n) = \frac{T(1)}{T(n)} = \frac{\Delta\left(\frac{1}{A} + C\right)}{\Delta\left(\frac{1}{nA} + nC\right)} = \frac{n(1+AC)}{1+ACn^2}$$

$$E_A(n) = \frac{S(n)}{n} = \frac{(1+AC)}{1+ACn^2}$$

$$E_B(n) = \frac{S(n)}{n} = \frac{(1+BC)}{1+BCn^2}$$

Es fácil comprobar, a la vista de las dos últimas expresiones, que si $A>B$ entonces $E_A(n)<E_B(n)$ por lo tanto el sistema A es menos escalable que el sistema B.

Pregunta 4

Las combinaciones posibles suponiendo una memoria secuencialmente consistente son las siguientes:

1	a1 a2 b1 b2	P1 mata a P2
2	a1 b1 a2 b2	Las condiciones nunca se cumplen y no muere nadie
3	a1 b1 b2 a2	Las condiciones nunca se cumplen y no muere nadie
4	b1 b2 a1 a2	P2 mata a P1
5	b1 a1 b2 a2	Las condiciones nunca se cumplen y no muere nadie
6	b1 a1 a2 b2	Las condiciones nunca se cumplen y no muere nadie

Es imposible que, dada una memoria secuencialmente consistente, ambos procesos mueran. El motivo es que la instrucción condicional viene siempre después de la asignación, entonces cuando ambos procesadores ejecutan la condicional ya han debido ejecutar las asignaciones cuyos resultados serán vistos en ese orden por todos los procesadores, por lo tanto no es posible que ambos ejecuten las instrucciones condicionales con éxito.

En cambio, si la memoria es consistente de procesador o PRAM, sí que es posible que ambos mueran, ya que este modelo de consistencia sólo exige la secuencialidad en las instrucciones de un procesador, pero no la secuencialidad entre instrucciones de procesadores diferentes, por lo tanto, es posible que ambos procesadores ejecuten la instrucción condicional al mismo tiempo, leyendo las variables del otro como cero, enviando por consiguiente la instrucción de *kill*. No es algo que pasara de forma sencilla pero podría ocurrir.

Problema

Primera parte

No hay encadenamiento ni demasiados recursos. La división en convoys y los tiempos de arranque correspondientes son lo que siguen:

12	LV	V1,R1
12	LV	V2,R2
12	ADDV LV	V2,V2,V1 V3,R3
6	SUBV ADDV	V2,V2,V3 V3,V3,V1
6	SUBV	V2,V3,V1
12	MULTV SV	V1,V2,V3 R1,V2
12	SV MULTV	R2,V1 V3,V3,V4
12	SV	R3,V3

Por lo tanto hay 8 convoys y 6 instrucciones en coma flotante. El tiempo de arranque será:

Tarranque=12+12+12+6+6+12+12+12=84

Tn=|n/64|(15+84)+8n

Tn(n>>)=9.55n+99

A partir de aquí es sencillo calcular el rendimiento para vectores con un número grande de elementos:

R∞=6*200/9.55=125.65 MFLOPS

Para calcular Nv necesitamos saber el tiempo que necesita un escalar para realizar las operaciones anteriores sobre un elemento, pero esto es un dato que nos dan, por lo tanto, el tiempo invertido por un escalar en procesar un vector de longitud n es 20n. Igualando lo que tar el escalar por elemento y lo que tarda el vectorial se puede calcular Nv (suponemos que Nv<64):

20Nv=99+8Nv y despejando:

Nv=99/(20-8)=8.25

Así que a partir de 9 elementos el procesador vectorial es más rápido que el escalar (para este caso concreto).

Segunda parte

Ahora ya se permiten más cosas, pero hay restricciones, como el número de cauces de escritura a los registros, que son las que principalmente nos marcan el número de convoys:

12+6=18	LV	V1,R1	12
	LV	V2,R2	12
	ADDV	V2,V2,V1	6
	LV	V3,R3	12
6+6+7+12=31	SUBV	V2,V2,V3	6
	ADDV	V3,V3,V1	6
	SUBV	V2,V3,V1	6
	MULTV	V1,V2,V3	7
	SV	R1,V2	12
	SV	R2,V1	12
7+12=19	MULTV	V3,V3,V4	7
	SV	R3,V3	12

El número de convoy es de 3 y el Tarranque=18+31+19=68, con lo que el tiempo de ejecución por elemento para n grande queda:

Tn(n>>)=4.297n+83

Y el Rendimiento cuando n tiende a infinito será:

R∞=6*200/4.297=279 MFLOPS

Para calcular el $N_{1/2}$ suponemos en primer lugar que $N_{1/2}$<=64, por lo que el tiempo por elemento queda como Tn(n<=64)=3n+83. El rendimiento para $N_{1/2}$ es la mitad de R∞, por lo tanto:

$R(N_{1/2})=6*200*N_{1/2} /(3*N_{1/2} + 83)=279/2$

Despejando tenemos:

$N_{1/2}$ = 83*139.5/(1200-3*139.5)=14.8

Que podemos redondear a 15 que es el entero más cercano