Ampliación de Estructura de Computadores
Soluciones de los ejercicios del Tema 1

Aquí se muestra la resolución de los ejercicios básicos propuestos en el boletín de ejercicios 1 de la asignatura Ampliación de Estructura de Computadores. El objetivo es que el alumno pueda comprobar si la resolución del ejercicio que ha efectuado es correcta o no, y aprender de la solución propuesta. Del mismo modo, se dan los resultados de los ejercicios adicionales.

Ejercicio 1

Se trata de calcular en ambos casos el tiempo de CPU usando la fórmula explicada en la teoría:

$$T_{CPU} = NI \times CPI \times T_{ciclo}$$

A partir de los resultados se determina qué máquina es más rápida y se calcula el factor de mejora.

Apartado a

Empezamos por la configuración original (sin aplicar la optimización que describe el enunciado):

$$T_{CPU\text{-}orig} = NI_{orig} \times CPI_{orig} \times T_{ciclo\text{-}orig}$$

donde $CPI_{orig} = 0.33 \times 2 + 0.21 \times 3 + 0.12 \times 3 + 0.24 \times 3 + 0.10 \times 12 = 3.57$, así pues nos queda que:

$$T_{CPU\text{-}orig} = NI_{orig} \times 3.57 \times T_{ciclo\text{-}orig}$$

Para la configuración alternativa el tiempo de ciclo se incrementa un 25% con respecto al original, y los ciclos que tardan las operaciones FP y las cargas se modifican. Tenemos que:

$$\begin{aligned} T_{ciclo\text{-}alt} &= 1.25 \times T_{ciclo\text{-}orig} \\ CPI_{alt} &= 0.33 \times 2 + 0.21 \times 4 + 0.12 \times 3 + 0.24 \times 3 + 0.10 \times 6 = 3.18 \end{aligned}$$

El número de instrucciones no cambia ($NI_{alt} = NI_{orig}$), por lo que:

$$T_{CPU\text{-}alt} = NI_{alt} \times 3.18 \times T_{ciclo\text{-}alt} = NI_{orig} \times 3.18 \times 1.25 \times T_{ciclo\text{-}orig} = NI_{orig} \times 3.975 \times T_{ciclo\text{-}orig}$$

Por lo tanto, la configuración original es 1.1134 veces $(3.975/3.57)$ más rápida que la alternativa propuesta, o lo que es lo mismo: un 11.34%.

Apartado b

En este caso hay que volver a calcular el CPI de la configuración alternativa teniendo en cuenta que el número de ciclos de las cargas no se ve afectado:

$$\begin{aligned} CPI_{alt} &= 0.33 \times 2 + 0.21 \times 3 + 0.12 \times 3 + 0.24 \times 3 + 0.10 \times 6 = 2.97 \\ T_{CPU\text{-}alt} &= NI_{orig} \times 2.97 \times 1.25 \times T_{ciclo\text{-}orig} = NI_{orig} \times 3.7125 \times T_{ciclo\text{-}orig} \end{aligned}$$

La configuración original sigue siendo más rápida, esta vez en un factor 1.04 $(3.7125/3.57)$.

Ejercicio 2

Los MIPS se pueden calcular como:

$$MIPS = \frac{Freq(MHz)}{CPI}$$

El CPI con el compilador estándar sería:

$$CPI_{std} = 0.43 \times 1 + 0.21 \times 2 + 0.12 \times 2 + 0.24 \times 2 = 1.57$$

Con lo que los MIPS serían:

$$MIPS_{std} = \frac{50}{1.57} = 31.85$$

El CPI con el compilador optimizado sería:

$$CPI_{opt} = \frac{0.215 \times 1 + 0.21 \times 2 + 0.12 \times 2 + 0.24 \times 2}{0.215 + 0.21 + 0.12 + 0.24} = 1.73$$

Con lo que los MIPS serían:

$$MIPS_{opt} = \frac{50}{1.73} = 28.96$$

El programa optimizado sería más lento según los MIPS, pero claramente podemos pensar que su tiempo de ejecución sería menor, ya que ejecuta menos instrucciones que el programa compilado con el compilador estándar.

Ejercicio 3

Inicialmente, la relación coste/rendimiento de la máquina es C/R. Vamos a calcular cómo se incrementa el rendimiento por un lado y el coste por otro con el cambio. Si el rendimiento se incrementa en mayor medida que el coste, o ambos aspectos se ven mejorados en igual proporción, entonces el cambio es interesante desde el punto de vista de la relación coste/rendimiento. Si el coste se incrementa en mayor medida que el rendimiento, entonces el cambio no es aconsejable.

Para calcular la mejora en el rendimiento que se obtiene con el cambio aplicamos la ley de Amdahl, considerando que la CPU se usa el 50% del tiempo:

$$Speedup = \frac{1}{1 - 0.5 + 0.5/5} = 1.667$$

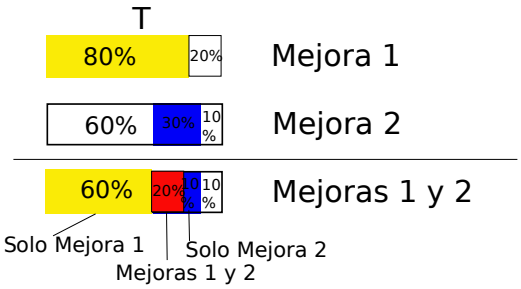
Lo que significa que el rendimiento tras el cambio $R' = 1.667 \times R$.

Por otro lado, la CPU supone $1/3$ del coste del sistema y se ha multiplicado por 5 su precio, con lo que el coste tras el cambio $C' = 2/3 \times C + 1/3 \times C \times 5 = 7/3 \times C = 2.334 \times C$

Concluimos, pues, que el cambio no es interesante, ya que el coste se incrementa en mayor proporción que el rendimiento (incremento de 2.334 veces en el coste para obtener un factor de mejora en el rendimiento de 1.667).

Ejercicio 4

Veamos primeramente, cuál es la situación que nos encontramos al aplicar cada mejora de forma individual y después consideraremos la aplicación de las 2 simultáneamente. Sea T el tiempo total de ejecución, la primera mejora afecta al 80% de T (reduce en un factor de 4 esta parte). La segunda mejora afecta al 30% de T. Cuando se usan conjuntamente hay un solapamiento del 20% (que se ve acelerado en un factor de 6). Esto se ha reflejado en el dibujo de abajo, en el cual se ha marcado en amarillo la fracción del tiempo original que se ve afectada por la primera mejora, en azul la parte afectada por la segunda y en rojo la parte afectada por ambas, situación que se muestra en la tercera barra:



Así pues, aplicando la Ley de Amdahl y considerando el factor de mejora de cada una de las fracciones de la barra anterior tenemos que:

$$Speedup = \frac{1}{1-(0.6+0.2+0.1)+0.6/4+0.2/6+0.1/2} = 3.003$$

Ejercicio 5

Vamos a obtener primero los ciclos empleados por cada tipo de instrucción antes de que se apliquen las mejoras y la fracción de ciclos totales en la ejecución de cada tipo de instrucción. Los resultados se muestran en la tabla siguiente. Respecto a la fracción de ciclos hay que tener en cuenta que el tiempo total de ejecución (en ciclos) será la suma de los 4 tiempos individuales: 20mill+600mill+350mill+60mill = 1030 millones de ciclos. Y con esto se obtiene fácilmente la fracción de ciclos correspondiente a cada tipo de instrucción.

Tipo	Cantidad	CPI	Factor mejora	Ciclos	Fracc. ciclos
Suma	10 millones	2 ciclos	2.0	20 millones	2%
Multiplicación	30 millones	20 ciclos	1,3	600 millones	58%
Memoria	35 millones	10 ciclos	3.0	350 millones	34%
Saltos	15 millones	4 ciclos	4.0	60 millones	6%

Con estos datos podremos usar la ley de Amdahl para calcular el efecto que cada una de las 4 mejoras propuestas, de forma independiente, tienen en el rendimiento global del procesador:

$$Speedup_{suma} = \frac{1}{(1-f)+\frac{f}{S}} = \frac{1}{(1-0.02)+\frac{0.02}{2}} = 1.01$$

$$Speedup_{mult} = \frac{1}{(1-0.58)+\frac{0.58}{1.3}} = 1.15$$

$$Speedup_{memoria} = \frac{1}{(1-0.34)+\frac{0.34}{3}} = 1.29$$

$$Speedup_{saltos} = \frac{1}{(1-0.06)+\frac{0.06}{4}} = 1.05$$

De forma que el orden de los tipos de instrucciones de mayor a menor impacto en el rendimiento global será: instrucciones de memoria, multiplicaciones, saltos y sumas.

Ejercicio 6

Las instrucciones de salto condicional emplean 2 ciclos y el resto 1 ciclo.

CPU A: 20% saltos, 20% comparaciones, 60% otras.

$Freq_A = 1,25 \times Freq_B$

Apartado a

Vamos a ver si B es más rápida que A:

$$Speedup = \frac{T_A}{T_B} = \frac{NI_A \times CPI_A \times Tc_A}{NI_B \times CPI_B \times Tc_B} = \frac{NI_A \times CPI_A \times Freq_B}{NI_B \times CPI_B \times Freq_A}$$

Como la CPU B ejecuta un 20% de instrucciones menos que la CPU de A (las comparaciones se unen a los saltos): $NI_B = 0,8 \times NI_A$

El $CPI_A = 0,2 \times 2 + 0,2 \times 1 + 0,6 \times 1 = 1,2$

El CPI de B es más complicado de calcular, ya que hay que *normalizar* el porcentaje (o ratios) de instrucciones al total de instrucciones que ejecuta B. En concreto, ejecuta un $\frac{0,2}{0,8} = 0,25$ de saltos y un $\frac{0,6}{0,8} = 0,75$ otras instrucciones. Por tanto, $CPI_B = 0,25 \times 2 + 0,75 \times 1 = 1,25$

Sustituyendo tenemos:

$$Speedup = \frac{NI_A \times 1,2 \times Freq_B}{0,8 \times NI_A \times 1,25 \times 1,25 \times Freq_B} = 0,96$$

Por tanto B no es más rápida que A. A es la más rápida.

Apartado b

$$Speedup = \frac{NI_A \times 1,2 \times Freq_B}{0,8 \times NI_A \times 1,25 \times 1,1 \times Freq_B} = 1,091$$

B es ahora un 9,1% más rápida que A.

Ejercicio 7

A partir de los datos que nos dan para las máquinas 1 y 2, y teniendo en cuenta que el único cambio entre ellas es la frecuencia de reloj a la que funcionan, podemos aplicar la Ley de Amdalh para calcular la fracción del tiempo (F) que se está utilizando el procesador:

$$Speedup = \frac{10}{9} = \frac{1}{1-F+F/(800/700)} \Rightarrow 1.11 = \frac{1}{1-F+0.875 \times F} \Rightarrow F = 80\%$$

Calculamos ahora el tiempo de ejecución en la máquina 3 (X):

$$\frac{10}{X} = \frac{1}{1-0.8+0.8/(1000/700)} \Rightarrow X = 7.6 \text{ s}$$

Suponiendo que la máquina 1 tiene una relación coste/rendimiento C/R, la máquina 2 incrementa el rendimiento en un factor 1.11 (10/9) y el costo en un factor 1.075 (4300/4000), con lo que el rendimiento aumenta en mayor medida que el coste. La máquina 3, por el contrario, aumenta el rendimiento en un factor 1.32 (10/7.6) mientras que coste se incrementa en un factor 1.375 (5500/4000). De esta forma, podemos concluir que la mejor opción desde el punto de vista coste/rendimiento es la máquina 2.

Ejercicio 8

Apartado a

Sabemos que $MIPS = \frac{Frecuencia \text{ (en MHz)}}{CPI}$.

Cuando se usa el coprocesador matemático para ejecutar las operaciones de punto flotante tenemos que CPI = 10, con lo que:

$$MIPS_{copro} = 16.67/10 = 1.667$$

Por el contrario, cuando se emplean las rutinas software el CPI es de 6:

$$MIPS_{rutinas} = 16.67/6 = 2.778$$

Vemos pues como los MIPS no son una buena métrica de rendimiento, ya que la configuración que más MIPS obtiene es precisamente la más lenta (más tiempo tarda).

Apartado b

Aplicamos ahora la siguiente fórmula:

$$MIPS = \frac{NI}{T_{CPU} \times 10^6}$$

Cuando usamos el coprocesador matemático:

$$1.667 = \frac{NI_{copro}}{1.08 \times 10^6} \Rightarrow NI_{copro} = 1.8 \times 10^6$$

Cuando usamos las rutinas software:

$$2.78 = \frac{NI_{rutinas}}{13.6 \times 10^6} \Rightarrow NI_{rutinas} = 37.81 \times 10^6$$

Apartado c

En la versión con el coprocesador podemos distinguir dos tipos de instrucciones: las de punto flotante (sabemos que son 195578) y las instrucciones enteras que conforman el programa. Puesto que sabemos el total de instrucciones ejecutadas y el número de instrucciones de punto flotante, podemos calcular cuántas instrucciones enteras tiene el programa:

$$InstrEnteras = NI_{copro} - 195578 = 1.8 \times 10^6 - 0.195578 \times 10^6 = 1.6 \times 10^6$$

Obviamente estas intrucciones enteras están presentes también en la versión del programa que emplea las rutinas software, por lo que el número de instrucciones en que se traducen las 195578 operaciones de punto flotante lo obtendremos restando al número total de instrucciones ejecutas en este caso ($NI_{rutinas}$) estas intrucciones enteras:

$$InstrPFenSW = 37.81 \times 10^6 - 1.6 \times 10^6 = 3.6 \times 10^7$$

Así pues, por cada operación de PF se requieren en media $3.6 \times 10^7 / 195578 = 185$ instrucciones enteras.

Apartado d

Los MFLOPS son millones de instrucciones de punto flotante por segundo. Sabemos que hemos necesitado 1.8 segundos para ejecutar un total de 195578 instrucciones de punto flotante, con lo que $MFLOPS = 0.195578/1.08 = 0.18$.

Ejercicio 9

Se trata de calcular en ambos casos el tiempo de CPU usando la fórmula explicada en la teoría:

$$T_{CPU} = NI \times CPI \times T_{ciclo}$$

A partir de los resultados se determina qué máquina es más rápida y se calcula el factor de mejora.
Empezamos por la configuración original (sin aplicar la optimizaciones que describe el enunciado):

$$T_{CPU\text{-}orig} = NI_{orig} \times CPI_{orig} \times T_{ciclo\text{-}orig}$$

donde $CPI_{orig} = 1$, así pues nos queda que:

$$T_{CPU\text{-}orig} = NI_{orig} \times 1 \times T_{ciclo\text{-}orig}$$

Para la configuración optimizada el tiempo de ciclo se incrementa un 5% con respecto al original, y se ejecutan $1/3$ menos de operaciones de carga y almacenamiento que suponen un 30%. Tenemos que:

$$T_{ciclo\text{-}opt} = 1.05 \times T_{ciclo\text{-}orig}$$
$$NI_{opt} = NI_{orig} - NI_{orig} \times 0.3/3$$

El CPI no cambia ($CPI_{opt} = CPI_{orig}$), por lo que:

$$T_{CPU\text{-}opt} = NI_{opt} \times 1 \times T_{ciclo\text{-}opt} = NI_{orig} \times 0.9 \times 1 \times 1.05 \times T_{ciclo\text{-}orig} = NI_{orig} \times 0.945 \times T_{ciclo\text{-}orig}$$

Por lo tanto, la configuración optimizada es 1.058 veces ($1/0.945$) más rápida que la configuración original, o lo que es lo mismo: un 5,8%.

Ejercicio 10

Teniendo en cuenta la primera opción para mejorar el rendimiento, aplicamos la ley de Amdahl, considerando que la aceleración es 1.176 y el factor de mejora igual a 2:

$$1.176 = \frac{1}{(1-D) + D/2}$$

Obtenemos $D = 30\%$

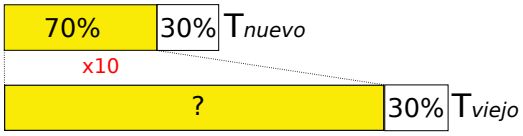
Teniendo en cuenta la segunda opción para mejorar el rendimiento, aplicamos la ley de Amdahl, considerando que la aceleración es 1.351, el factor de mejora del disco igual a 1.25 y el factor de mejora del coprocesador igual a 2:

$$1.351 = \frac{1}{(1-D-F) + D/1.25 + F/2}$$

Obtenemos $F = 40\%$.

Ejercicio 11

A este tipo de ejercicio también se le conoce como aplicación de la ley de Amdahl "inversa", en el sentido de que nos piden qué fracción del tiempo original (o viejo) es el que se ha visto afectado, en lugar de darnos dicha fracción y ver cómo se ha visto afectado el tiempo nuevo.



Tal y como se observa en la figura, el *tiempo nuevo* puede expresarse como:

$$T_{nuevo} = 0,30 \times T_{nuevo} + 0,70 \times T_{nuevo}$$

De forma similar, podemos decir que el *tiempo viejo* (que será mayor) puede expresarse como:

$$T_{viejo} = 0,30 \times T_{nuevo} + 0,70 \times T_{nuevo} \times 10 = 7,3 \times T_{nuevo}$$

Por tanto, el porcentaje del *tiempo viejo* que se ha visto afectado por la mejora es $\frac{7}{7,3} = 96\%$ y se ha mejorado el tiempo de ejecución en 7,3 veces.

Ejercicio 12

Apartado a

Opción 1:

$$Speedup_1 = \frac{1}{1 - f_a - 2f_a + \frac{f_a}{10} + 5 \times 2f_a} = \frac{1}{1 + 7f_a + \frac{f_a}{10}}$$

Opción 2:

$$Speedup_2 = \frac{1}{1 - f_b - 0,5f_b + \frac{f_b}{20} + 2 \times 0,5f_b} = \frac{1}{1 - 0,45f_b}$$

Opción 3:

$$Speedup_3 = \frac{1}{1 - f_a - f_a + \frac{f_a}{5} + 1,8f_a} = \frac{1}{1 - 2f_a + 2f_a} = 1$$

Apartado b

Como se observa, en la opción 1 se empeora el tiempo de ejecución, mientras que en la opción 3 se mantiene igual. Por tanto, la mejor opción es la 2 ya que es la única que realmente supone una mejora en el rendimiento final.

Ejercicio 13

Apartado a

Para calcular los MIPS podemos usar la fórmula simplificada:

$$MIPS = \frac{F(MHz)}{CPI} = \frac{100}{10} = 10\ MIPS$$

Para los MFLOPS, simplemente podemos tomar la fórmula de su definición:

$$MFLOPS = \frac{Num\_instr\_PF}{T_{cpu} \times 10^6} = \frac{3000}{10^{-3} \times 10^6} = 3\ MFLOPS$$

Apartado b

Los MIPS incluyen las instrucciones en coma flotante, así que si sabemos que tenemos 10 MIPS, es decir, 10 000 000 de instrucciones en un segundo, de las cuales 3 000 000 son instrucciones en coma flotante, resulta que tiene que haber 7 000 000 instrucciones puramente enteras en un segundo. Y si multiplicamos por los segundos que tarda el programa (0.001 s), nos da 7 000 instrucciones enteras.

Apartado c

Nos dicen que 1 instrucción en coma flotante se convierte en 9 instrucciones enteras por término medio. Esto nos permite calcular fácilmente el número de instrucciones enteras que tendría nuestro programa tras la traducción: $7\,000 + 3\,000 \times 9 = 34\,000$ instrucciones. Como también nos dicen que el CPI resultante es 4, podemos aplicar la fórmula del tiempo de ejecución para obtener la frecuencia de funcionamiento:

$$0.001\ s = NI \times CPI \times \frac{1}{F} = 34\,000 \times 4 \times \frac{1}{F}$$

y despejando F:

$$F = 134 \times 10^6 = 136MHz$$

Ejercicio 14

Apartado a

Llamamos A, B y C a las máquinas con rendimientos de 5, 15 y 25 MFLOPS respectivamente. Aplicando la ley de Amdahl a las máquinas A y B, podemos obtener la fracción de tiempo que la aplicación usa el hardware de punto flotante (el resto del tiempo podemos suponer que la máquina está realizando otras tareas o ejecutando instrucciones enteras):

$$\frac{T_A}{T_B} = \frac{20}{10} = \frac{1}{(1-f) + \frac{f}{S}} = \frac{1}{(1-f) + \frac{f}{15/5}}$$

donde S es la cantidad de mejora de la máquina B respecto a A como consecuencia de las diferencias en su hardware para la ejecución de las operaciones de punto flotante, y que se concreta en la relación entre sus respectivas frecuencias en MFLOPS $(15/5 = 3)$. Despejando obtenemos $f = 0.75$, que es la fracción de tiempo que la máquina A utiliza el hardware de punto flotante al ejecutar la aplicación.

Apartado b

A partir del apartado anterior, podemos calcular el tiempo de ejecución de la aplicación en la máquina C:

$$\frac{T_A}{T_C} = \frac{20}{T_C} = \frac{1}{(1-f) + \frac{f}{S}} = \frac{1}{(1-0.75) + \frac{0.75}{25/5}} \rightarrow T_C = 8s$$

Apartado c

Por último, suponiendo que pudiésemos acelerar al máximo el hardware de punto flotante (aceleración infinita), esto resultaría en que no tardaríamos nada en ejecutar la parte de la aplicación, lo que nos da el tiempo de ejecución mínimo que se podría alcanzar en base a mejorar esta parte de la máquina.

$$\frac{T_A}{T_{min}} = \frac{20}{T_{min}} = \frac{1}{(1-0.75)} \rightarrow T_{min} = 5s$$