Laboratorio Sesión 09: Coma flotante, precisión y rendimiento

Objetivo

El objetivo de esta sesión es introducir el uso y los problemas de precisión de los cálculos en coma flotante además de realizar medidas de rendimiento. Para recordar cómo se tratan en un procesador actual las operaciones en coma flotante podéis repasar la Sección 3.5 del libro Çomputer organization and design: the hardware/software interface"de David A. Patterson y John L. Hennessy.

Conocimientos Previos

MIPS vs. MFLOPS

Ya en la práctica 1 vimos los MIPS como forma de medir el rendimiento de un procesador. Sin embargo los MIPS son una unidad de medida que es extremadamente variable ya que el mismo código fuente se puede implementar con más o menos instrucciones: dependiendo del ISA del procesador, del compilador y de las optimizaciones un mismo programa que se ejecuta en el mismo tiempo puede dar lugar a medidas de más o menos MIPS. Para intentar paliar este tipo de problemas, al menos en los programas científicos, se utilizan como unidad de medida de la velocidad de un procesador los MFLOPS (Millones de Operaciones de Coma Flotante por Segundo). Las diferencias con los MIPS son principalmente dos:

- Mide las operaciones y no las instrucciones, es decir, una multiplicación siempre es una operación, independientemente de la cantidad de instrucciones que hagan falta para implementarla. Así pues esta medida es más independiente del código máquina que los MIPS.
- Las operaciones son de Coma Flotante y no enteras. Esto hace que los MFLOPS sean adecuados para medir el rendimiento de programas de cálculo numérico (que usan este tipo de operaciones), pero no para programas que no usan la coma flotante. Es más, un procesador enfocado a la ejecución de códigos enteros (y por tanto sin unidad hardware específica de coma flotante) puede ser muy rápido y tener una velocidad en MFLOPS muy lenta.

Además, como pasa con los MIPS, hay una gran diferencia entre la velocidad máxima teórica (o de pico) y la velocidad real a la que se pueden ejecutar las instrucciones (debido a la influencia de la memoria, etc.). Para hacer comparaciones entre computadores se suelen ejecutar un conjunto concreto de códigos como por ejemplo el benchmark `LINPACK` (usado en el `TOP500`).

Estudio Previo

1. Dado el número -23,75 exprésalo en el formato IEEE de simple precisión. A continuación transfórmalo al formato IEEE de doble precisión.
2. Dado el número 1048576,2 averiguad si se puede codificar de forma exacta en el formato IEEE de simple precisión. Si no se puede, calculad el error que se introducirá al expresarlo en dicho formato. Finalmente explica si el formato IEEE de doble precisión permitiría codificar dicho número de forma exacta o no.
3. Buscad en el manual de ensamblador del x86 qué hacen las siguientes instrucciones ensamblador: `flds`, `fmuls`, `fadds` y `fstps`.
4. Dado el siguiente código en alto nivel y dos posibles códigos equivalentes en ensamblador, así como los tiempos que tardan en ejecutarse en el mismo procesador, calculad los MIPS y MFLOPS de cada código así como el Speedup del segundo código respecto al primero.

Código C

```
for (i=0; i<256; i++)
  C[i] = C[i] + A[i] * B[i];
```

Código ASM 1 (2,6 μs)

```
80484a3: movl  $0x0,0x34(%esp)
80484ab: jmp   80484e2
80484ad: mov   0x34(%esp),%eax
80484b1: mov   0x34(%esp),%edx
80484b5: flds  0x804a840(,%edx,4)
80484bc: mov   0x34(%esp),%edx
80484c0: flds  0x804a040(,%edx,4)
80484c7: mov   0x34(%esp),%edx
80484cb: flds  0x804a440(,%edx,4)
80484d2: fmulp %st,%st(1)
80484d4: faddp %st,%st(1)
80484d6: fstps 0x804a840(,%eax,4)
80484dd: addl  $0x1,0x34(%esp)
80484e2: cmpl  $0xff,0x34(%esp)
80484ea: jle   80484ad
```

Código ASM 2 (2,2 μs)

```
80484ab: mov   $0x0,%eax
80484b0: mov   $0x804a840,%edx
80484b5: mov   $0x804a040,%ecx
80484ba: flds  (%ecx,%eax,4)
80484bd: fmuls 0x804a440(,%eax,4)
80484c4: fadds (%edx,%eax,4)
80484c7: fstps (%edx,%eax,4)
80484ca: add   $0x1,%eax
80484cd: cmp   $0x100,%eax
80484d2: jne   80484ba
```

Trabajo a realizar durante la Práctica

1. Para empezar esta práctica vamos a ver algunos de los problemas que nos puede dar el hecho de usar números en formato IEEE debido a los errores de precisión. Dada la siguiente operación:

   $$z = x^4 - 4y^4 - 4y^2$$

   Haced un programa que calcule el resultado para z de la anterior operación para los valores iniciales $x = 665857$ y $y = 470832$ en doble precisión.

2. A continuación calculad la misma operación en simple precisión.

3. Finalmente computad el resultado usando variables enteras largas (`long long`).

4. Explicad las diferencias observadas y razonad cuál es la solución correcta y por qué[1]. Para comprobar el resultado podéis utilizar una calculadora, un móvil, el programa *bc* o incluso la calculadora de *Google*. Ojo, sólo os aseguramos que *bc* da el resultado correcto.

5. A continuación vamos a ver la importancia de los MFLOPS como elemento de medida de la velocidad. Partiendo del programa `SumMulMat.c` que tenéis en el paquete de programas de la práctica, ejecutadlo en vuestro ordenador y calculad a cuantos MIPS y a cuantos MFLOPS se ejecuta. Para calcular las operaciones de coma flotante y el tiempo tened en cuenta la parte del programa que se indica en el código del mismo. Para calcular las instrucciones, aunque es una aproximación, podéis usar por las instrucciones totales del programa tal y como las indica la herramienta `valgrind`. Recordad no medir a la vez tiempo e instrucciones.

6. A continuación compilad optimizando el código, es decir, con la opción `-O2`. Calculad de nuevo a cuantos MIPS y MFLOPS se ejecuta en este caso. Calculad el Speedup que se ha obtenido en este caso.

[1] Para saber más del tema podéis consultar: David Goldberg. 1991. What every computer scientist should know about floating-point arithmetic. ACM Comput. Surv. 23, 1 (March 1991), 5-48.

Nombre: Carlos Sansón Martín Grupo: 22

Nombre: Agnés Masip Gómez

Hoja de respuesta al Estudio Previo

1. El número -23,75 en formato IEEE se expresa:

   En simple precisión: 0xC1BE0000

   En doble precisión: 0xC037000000000000

2. Dado el número 1048576,2:

   Se codifica exacto en simple precisión (S/N): N

   Error en simple precisión: 0,05

   Se codifica exacto en doble precisión (S/N): N

3. Las instrucciones ensamblador: `flds`, `fmuls`, `fadds` y `fstps` sirven para:

   La instrucción flds es para hacer un load con números de coma flotante. fmuls hace multiplicaciones en coma flotante, fadds hace sumas en coma flotante y fstps hace stores en coma flotante y hace un pop.

4. El primer código en ensamblador se ejecuta:

   MIPS: 1281,54 MFLOPS: 196,92

   El segundo código en ensamblador se ejecuta:

   MIPS: 819,1 MFLOPS: 232,73

   Speedup con respecto al primer código: 1,18

   Comenta de forma crítica los resultados anteriores:

   Como el speedup es superior a 1, eso quiere decir que el tiempo de ejecución del segundo código tiene mejor rendimiento.

Nombre: Carlos Sansón Martín Grupo: 22

Nombre: Agnés Mosip Gómez

Hoja de respuestas de la práctica

1. El resultado de la operación $z = x^4 - 4y^4 - 4y^2$ con $x = 665857$ y $y = 470832$ en doble precisión es: 11885568.0

2. El resultado de la operación $z = x^4 - 4y^4 - 4y^2$ con $x = 665857$ y $y = 470832$ en simple precisión es: -88673112.4480

3. El resultado de la operación $z = x^4 - 4y^4 - 4y^2$ con $x = 665857$ y $y = 470832$ en enteros es: 1

4. Calculado con medios (*bc*, *Google*, Calculadora, etc.) externos es:

   bc: 1

   Nombre Medio 2: Calculadora de Google

   Resultado Medio 2: 11885568

   Explica cuál y por qué es el resultado correcto de los anteriores. ¿Qué problemas genera lo que habéis descubierto y cuáles son las posibles soluciones?

   La opción correcta es el bc, ya que las otras maneras causan problemas al tener overflow y no manejarlo correctamente. Por eso, la solución sale bien al usar long long ya que hay espacio suficiente para que no haya overflow.

5. El programa en C de la práctica ejecuta:

   Instrucciones: 839 634 177 Segundos: 0,185

   Operaciones de Coma Flotante: 33 554 432

   MIPS: 4538,56 MFLOPS: 181,38

6. Optimizado el programa ejecuta:

   Instrucciones: 119 168 622 Segundos: 0,022

   Operaciones de Coma Flotante:

   MIPS: 5416,76 MFLOPS: 1525,20

   Speedup: 8,41