1 Conceptos teóricos.

1.1 Definición de métricas de rendimiento.

- **Tiempo de respuesta.** Tiempo transcurrido entre el comienzo y el final de un evento.
- **Productividad.** Cantidad total de trabajo realizado por unidad de tiempo.
- **Aceleración o speed-up.** Ganancia que se obtiene al mejorar alguna parte de un computador o de un programa.

$$\textbf{Aceleracion o speed} - \textbf{up}. = \frac{tiempo\ de\ ejecucion\ mejorado}{tiempo\ de\ ejecucion\ sin\ mejorar}$$

$$\textbf{Aceleracion global} = \frac{1}{1 - ITM + \frac{ITM}{speed-up}}$$

  donde, ITM= Intervalo de tiempo de la mejora.

- **Tiempo de CPU.** Mide el tiempo que la CPU está calculando, sin incluir el tiempo de espera para las E/S. Este tiempo se divide en tiempo de CPU de usuario, que es el tiempo empleado por la CPU en el programa, y tiempo de CPU del sistema, que es el tiempo empleado por el S.O. realizando tareas requeridas por el programa.
- **Rendimiento de la CPU.** Tiempo de CPU de usuario.

$$\textbf{Tiempo de CPU de usuario} = NI\ .\ CPI\ .\ tiempo\ de\ ciclo\ de\ reloj$$

  donde CPI se define como,

$$\textbf{CPI} = \frac{ciclos\ de\ reloj\ para\ un\ programa}{numero\ de\ instrucciones}$$

  y NI es el número de instrucciones para un programa.

- **MIPS (Millones de instrucciones por segundo).**

$$\textbf{MIPS} = \frac{numero\ de\ instrucciones}{tiempo\ de\ ejecucion\ .\ 10^6}$$

$$\textbf{MIPS} = \frac{frecuencia\ de\ reloj}{CPI\ .\ 10^6}$$

- **MFLOPS (Millones de operaciones en punto flotante por segundo).**

$$\textbf{MFLOPS} = \frac{numero\ de\ operaciones\ en\ punto\ flotante}{tiempo\ de\ ejecucion\ .\ 10^6}$$

1.2 Comparación de dos máquinas.

Para comparar dos máquinas se dice que la máquina X es un $n\%$ más rápida que la máquina Y.

$$\frac{Tiempo\ de\ ejecucion_Y}{Tiempo\ de\ ejecucion_X} = 1 + \frac{n}{100}$$

1.3 Medida de los distintos tiempos asociados con la ejecución de un programa.

(a) **time** (comando de tiempo del S.O. UNIX). Mide los tiempos asociados con la ejecución de un programa. Se le pone como argumento el nombre de un fichero ejecutable y devuelve los siguientes valores asociados con la ejecución de ese programa en conjunto:

- Tiempo de CPU de usuario: 00.0u
- Tiempo de CPU del sistema: 00.0s
- Tiempo de respuesta del sistema: 0:00
- Utilización de la CPU: 0%

(b) **clock()** (función de tiempo disponible en ANSI C). Esta función permite medir el tiempo de CPU de usuario de cualquier parte de un programa. Para poder llamarla se debe incluir en el programa el fichero "time.h". Veamos un ejemplo de cómo se utiliza:

```
# include "time.h".
main()
{
definición de variables del programa;
clock_t tiempo1,tiempo2;
double tiempo;
tiempo1=clock();
CUERPO DEL PROGRAMA;
tiempo2=clock();
tiempo=(double)(tiempo2-tiempo1)/(double)CLOCKS_PER_SEC;
}/*end of main*/
```

La función clock() devuelve una medida de tiempo en ticks. Cada máquina genera un número de ticks por segundo, que está contenido en la variable CLOCKS_PER_SEC. Por lo tanto para conseguir una medida de tiempo en segundos se necesita dividir el valor que se obtiene de clock() por CLOCKS_PER_SEC.

2 Realización de la práctica

Para la realización de esta práctica vamos a utilizar dos bucles del *bechmark* sintético *Whetstone*. El listado del programa que incluye dichos bucles aparece como anexo.

1. Medir el rendimiento de tu máquina para el programa que se da como anexo.
2. Medir el rendimiento de otra máquina distinta para el mismo programa. ¿Cuál de ellas obtiene mayor rendimiento? ¿Qué % es más rápida una que la otra para ese programa?
3. Medir el Tiempo de CPU de usuario correspondiente a la ejecución de las instrucciones "*for*" etiquetadas como L1 y L2 .
   Determinar que % representa el tiempo de cada uno de los lazos en relación al tiempo total del programa.
4. Vamos a centrar nuestra atención en la línea marcada como “línea a estudiar”:

   ```
   x=sqrt(exp(log(x+1)/t1));  /*!!!! línea a estudiar!!!! */
   ```

   En esta línea se hace el cálculo de la expresión matemática:

   $$\sqrt{e^{\frac{\log(x+1)}{t1}}}$$

   Una simplificación de esa expresión es:

   $$e^{\frac{\log(x+1)}{2t1}}$$

   Escribir de nuevo dicha línea para que calcule la anterior expresión y medir de nuevo el tiempo de CPU para el bucle L2. ¿Crees que el compilador ha detectado esa simplificación? ¿En qué % se mejora el rendimiento de dicho bucle? En qué % se mejora el rendimiento del programa completo?. (Haz un cálculo teórico y compáralo con la mejora experimental, que se obtiene si se mide de nuevo el rendimiento de la máquina para el programa completo mejorado).
5. Volvemos a modificar la línea del ejercicio anterior añadiendo una operación más de la forma:

   ```
   x =  log(exp(log(x+1)/(2*t1)));
   ```

   Mide el tiempo de CPU del lazo L2 de nuevo. Escribe la expresión matemática a la que corresponde esa línea. Hay una simplificación matemática muy clara. ¿Crees que el compilador la ha detectado? Justifica tu respuesta.

CUESTIONES TEÓRICAS:

Supongamos que en las dos máquinas anteriores varía la ejecución de la instrucción de salto condicional.

- CPU A: Hay una instrucción de comparación que inicializa el código de condición y es seguida por un salto que examina el código de condición.
- CPU B: En la instrucción de salto ya se incluye la comparación.

1. En ambas CPUs, la instrucción de salto condicional emplea dos ciclos de reloj y las demás instrucciones una. En la CPU A, el 20% de las instrucciones ejecutadas son saltos condicionales, como cada salto necesita una comparación, otro 20% de las instrucciones son comparaciones. Debido a que la CPU A no incluye la comparación en el salto, su ciclo de reloj es un 25% más rápido que el de la CPU B. ¿Qué CPU es más rápida?.
2. Haciendo un estudio de diseño se vio, que en las máquinas anteriores la diferencia de las duraciones de los ciclos de reloj se podrán reducir un 10%. ¿Qué CPU es más rápida ahora?.
3. Supongamos que se quiere mejorar la velocidad de la CPU en un factor de 5 (sin afectar al rendimiento de E/S) y esto supone incrementar su coste 5 veces. Supongamos también que la CPU se utiliza el 50% del tiempo y que el tiempo restante la CPU está esperando las E/S. Si la CPU supone un tercio del coste total del computador. ¿Es una buena inversión el incremento de velocidad de la CPU en un factor de 5 desde el punto de vista coste/rendimiento?