Evaluación de la eficiencia de un programa

Técnicas Experimentales
Práctica de Laboratorio #11

3 de mayo de 2013

Resumen

El objetivo de esta práctica de laboratorio es realizar mediciones sobre la ejecución de un programa que proporcionen información sobre el uso que se está haciendo de los recursos computacionales.

1. Motivación y Objetivos

Entre los factores que influyen en el rendimiento de un programa se puden citar los siguientes:

- *Hardware*. El hardware condiciona de manera muy importante en rendimiento de un programa, en general, depende de la potencia del conjunto de instrucciones de la máquina y de su velocidad.
- *Software*. El software es también un factor muy importante del rendimiento de un programa. Este rendimiento depende básicamente de la capacidad de los compiladores y de las bibliotecas o librerias utilizadas.
- *Contenido del programa*. El contenido del programa tambien es un factor que influye en su rendimiento. Por ejemlo, la cantidad de operaciones en punto flotante que se realizan contribuye de manera significativa al tiempo de cómputo, puesto que son muy costosas debido a que emplean mucho más tiempo de la ALU que las operaciones enteras.
- *Diseño de la aplicación*. Es importante ajustar el diseño del programa al tipo de máquina sobre el que se va a ejecutar de manera que organicen bien los datos y se saque el máximo rendimiento de la misma. Si se tiene software que no se adapta bien al hardware, el rendimiento del mismo será muy pobre.

1.1. Medidas de tiempo

El tiempo no es sólo tiempo en una computadora.

Considere las siguientes definiciones:

- El tiempo transcurrido (en inglés, *elapsed time*) es el tiempo que se puede medir mediante un cronómetro o un reloj de pared (en inglés, *wall clock time*).
- Mientras que el tiempo de CPU es la cantidad de tiempo que un programa mantiene a la Unidad Central de Proceso ocupada.
- El tiempo del sistema es el tiempo empleado en tareas del Sistema Operativo como las operaciones de Entrada/Salida.
- El concepto de tiempo de usuario es la diferencia entre el tiempo de CPU y el tiempo del sistema.

Si una máquina está ocupada por múltiples procesos concurrentes, el tiempo de CPU de un programa puede ser bastante diferente del tiempo transcurrido.

Python tiene un módulo `time` con algunas funciones útiles para medir el tiempo transcurrido y el tiempo de CPU. Por ejemplo, en el siguiente código se muestra el uso de los métodos `time` y `clock`.

```
import time
e0 = time.time()     # Unix epoch time
c0 = tieme.clock()   # CPU time
... # do tasks
elapsed_time = time.time() - e0
cpu_time = time.clock() - c0
```

Aunque las medidas de tiempo tienen una resolución más fina que los segundos, se deberían construir casos de prueba que empleen varios segundos para obtener resultados fiables.

Para medir la eficiencia de un conjunto de sentencias o de una expresión, el código se debería de ejecutar varias veces para que el tiempo de CPU total sea del orden de segundo. El módulo `timeit` tiene funcionalidades para ejecutar un segmento de código repetidamente.

```
>>> import timeit
>>> t = timeit.Timer('sin(1.2)', setup='from math import sin')
>>> t.timeit(10000000)
2.6692249774932861
>>> t = timeit.Timer('math.sin(1.2)', setup='import math')
>>> t.timeit(10000000)
3.7189419269561768
>>>
```

Se deduce que `sin(1.2)` se ejecuta aproximadamente un 30 % más rápido que `math.sin(1.2)`.

1.2. Información del Hardware

Junto con las medidas de tiempo con frecuencia es conveniente mostrar la información acerca del hardware sobre el cual se han realizado los experimentos. El siguiente código Python acceder al fichero `cpuinfo` y muestra la información.

```
import os

def CPUinfo():
# infofile on Linux machines:
  infofile = '/proc/cpuinfo'
  cpuinfo = {}
  if os.path.isfile(infofile):
    f = open(infofile, 'r')
    for line in f:
      try:
        name, value = [w.strip() for w in line.split(':')]
      except:
        continue
      if name == 'model name':
        cpuinfo['CPU type'] = value
      elif name == 'cache size':
        cpuinfo['cache size'] = value
      elif name == 'cpu MHz':
        cpuinfo['CPU speed'] = value + ' Hz'
      elif name == 'vendor_id':
        cpuinfo['vendor ID'] = value
    f.close()
  return cpuinfo

if __name__ == '__main__':
  print CPUinfo()
```

1.3. Información del Sistema Operativo y el Compilador

También es conveniente informar acerca del Sistema Operativo donde se han realizado los experimentos, así como sobre la versión del compilador que se ha utilizado. Python tiene el módulo `platform` con funcionalidades que permiten obtener esta información.

```
>>> import platform
>>> platform.uname()
('Linux', 'europa', '2.6.24-32-generic', '#1 SMP Tue Sep 25 17:26:23 UTC 2012', 'i686', '')
>>> platform.platform()
'Linux-2.6.24-32-generic-i686-with-debian-lenny-sid'
>>> platform.python_version()
'2.5.2'
>>> platform.python_build()
('r252:60911', 'Oct 12 2012 20:21:56')
>>>
```

2. Ejercicios propuestos

Crear un módulo en el que se importe la función que calcula el porcentaje de error en una identidad matemática implementada en la práctica anterior.

- Escriba un programa Python que le solicite al usuario un nombre de fichero y almacene en el mismo la información relativa al hardware, el Sistema Operativo y la versión del intérprete que se está ejecutando.
- Diseñe una experimento que le permita medir la eficiencia de la evaluación de una expresión frente a la de su equivalente. Por ejemplo, la evaluación de la expresión $(ab)^3$ en Python es un 40 % más rápida que la de la expresión a^3b^3.

3. Para saber más...

Diseñe un experimento que le permita comparar el tiempo que se emplea en realizar las siguientes tareas:

1. Aplicar la función `error` al listado de quince identidades matemáticas de las prácticas anteriores para cinco valores de '*umbral*' diferentes.
2. Abrir el fichero que contiene los resultados del experimento anterior y cargarlo en memoria.

Referencias

[1] Tutorial de Python. http://docs.python.org/2/tutorial/