CONCEPTES AVANÇATS DE SISTEMES OPERATIUS (CASO)

Facultat d'Informàtica de Barcelona, Dept. d'Arquitectura de Computadors, curs 2019/2020 – 2Q

Pràctiques de laboratori Mesures de rendiment

Material

~~La vostra instal·lació de **Linux (Ubuntu, Debian...)**~~. La **instal·lació de Linux** que tingueu cada estudiant disponible, a casa, o amb una connexió remota als laboratoris de la FIB.
Una **eina de missatgeria** que us permeti interactuar amb el vostre company/a de pràctiques. Donades les circumstàncies, **eviteu de totes totes les trobades presencials!**
~~Un disc dels antics (els haureu de compartir, planificats en RR).~~

Mesures de rendiment

Per mesurar el rendiment d'aplicacions i sistemes operatius habitualment usem diverses mètriques:
- Temps d'execució
- Acceleració (speedup): la relació entre el temps d'execució en seqüencial i el temps d'execució en paral·lel
- Ample de banda (bandwidth): la relació entre les dades transmeses i el temps que s'ha invertit en transmetre-les
- Latència: Cost d'iniciar operacions o comunicacions

Les mesures preses poden tenir una certa variabilitat, per això és bo repetir els experiments un cert número de cops. Per exemple, fem l'experiment 10 cops i...
- Temps d'execució: fem la mitjana dels temps obtinguts en els 10 experiments
- Speedup: fem les mitjanes de les 10 executions seqüencials i de les 10 execucions en paral·lel i l'speedup es calcula com a la seva relació
- Bandwidth: fem la mitjana dels temps d'execució dels 10 experiments i calculem el bandwidth o bé fem la mitjana dels diferents bandwidths obtinguts
- Latència: mitjana de les latències obtingudes en N experiments

També és bo calcular la desviació estàndard de les mesures, que ens donarà una idea de la seva variabilitat respecte la mitjana.

Eines de suport

L'eina més bàsica per a prendre mesures de temps en Linux/UNIX és **gettimeofday**(...). És una crida a sistema que retorna el temps transcorregut des de l'1 de gener del 1970[1]. Veure el seu manual ($man gettimeofday).

Amb aquesta crida podem obtenir el temps abans i després del codi que volem mesurar i fer la diferència.

Linux proporciona altres eines, en forma de comandes, que ens permeten analitzar l'execució del sistema o de les aplicacions: *top, ps, vmstat, iostat...*

En el codi de suport d'aquest laboratori us proporcionem un exemple que mostra com aconseguir la data/hora actuals i un altre per comprovar quina és la fórmula correcte per calcular diferències dels temps presos amb *gettimeofday*.

1 http://en.wikipedia.org/wiki/Unix_time

Exercicis

1. Escriviu un programa que calculi el temps que triga una crida a sistema senzilla. Per exemple, aquestes són interessants:

- **sbrk(0)**, que només ha de retornar el *break point*, apuntant a la primera adreça invàlida després del *heap*. Veieu la diferència provant també amb **sbrk(inc)**, amb inc > 0. Quin "inc" podeu posar sense comprometre el sistema?
- **sched_yield()**, que només suggereix al SO que si troba un altre flux per executar, canviï de context.
- **getpid()**, que retorna el pid del procés.
- **fork()/waitpid()**, per crear i esperar un procés fill.

  Per a fer-ho, feu programes que executin aquestes crides milions de vegades (desenes de milers en el cas del fork()) i preneu el temps.

Feu una taula amb els temps d'execució que obteniu. Per què els temps d'execució són tan diferents?

Syscall	sbrk(0)	sbrk(inc)	sched_yield()	getpid()	fork/waitpid
Execution time (in microsecs.)					

2. Podeu comprovar d'alguna manera que els programes executen realment la crida a sistema? (I no aprofiten el resultat retornat per la crida anterior)
3. A la darrera transparència del tema Virtualització-Sincronització-MesuresdeRendiment (transp. 50) us demanem que feu un programa que escrigui al disc 500MBytes, mesurant el temps que triga a fer-ho. Des d'un usuari no privilegiat (no root), executeu el vostre programa sobre un fitxer en el disc de la màquina que feu servir per aquest laboratori. Per exemple, podeu fer-ho sobre un fitxer en el directori /tmp. Si teniu dues versions del codi, una desenvolupada per cada membre del grup:

   $ write-to-disk1 /tmp/file1.dat $ write-to-disk2 /tmp/file2.dat

4. Modifiqueu el(s) programa(es) per poder canviar la mida de les dades escrites al disc. Un cop executat(s) amb les diferents mides, per exemple (0.5, 1, 2, 4 Gbytes, si teniu espai al disc), feu una gràfica amb els resultats que obteniu de bandwidth. A l'eix de les X situeu el tamany de les dades transmeses i a l'eix de les Y, el bandwidth. Podeu usar LibreOffice scalc, el GNU plot, el jgraph… i comparar els resultats que obteniu amb el vostre company de laboratori.
5. <opcional> Si executeu el programa com a administradors (root), obteniu alguna diferència en els resultats?
6. <opcional> Escriviu i executeu una aplicació similar a la del punt 2, però que llegeixi del disc. Feu una gràfica similar a la del punt 2, que mostri ara el bandwidth obtingut per les lectures. Compareu els resultats.
7. <opcional> Useu les eines del sistema **vmstat** i **iostat** per veure el bandwidth obtingut en lectura i escriptura de fitxers quan executeu els programes write-to-disk i read-from-disk, per separat i alhora. Per una mida concreta (p. ex. 500 Mb), coincideixen amb els vostres números?

L'entrega, a través del Racó, abans del 16/04/20

Entregareu la taula de l'apartat 1, la resposta a la pregunta 2, i els programes i les gràfiques obtingudes a l'apartat 3.