COMPUTACIÓ D'ALTES PRESTACIONS – PRÀCTICA 2

Semestre Setembre 2013

- Presentació i objectius
- Enunciat de la pràctica
- Criteris d'avaluació
- Format de lliurament
- Data de lliurament

Presentació i objectius

Objectius

Aquesta pràctica té com a objectius introduir els conceptes bàsics dels models de programació MPI i OpenMP. Per a la realització es posaran en pràctica diversos dels conceptes presentats en aquesta assignatura.

Presentació de la pràctica

Cal lliurar els fitxers amb el codi font realitzat i un document amb les respostes a les preguntes formulades, l'avaluació de rendiment i els comentaris que considereu. Tota la codificació es realitza exclusivament en llenguatge C amb les extensions associades a MPI i OpenMP.

Restriccions de l'entorn

Tot i que el desenvolupament de la pràctica és molt més interessant utilitzant dotzenes de computadors, s'assumeix que inicialment tindreu accés a un nombre força reduït de computadors amb diversos nuclis per node.

Material per a la realització de la pràctica

En els servidors de la UOC teniu el programari necessari per executar MPI i OpenMP. De totes maneres, si voleu fer el vostre desenvolupament i proves al vostre sistema local haureu de tenir instal·lat algun software que implementi MPI. Podeu utilitzar el que millor us sembli però us aconsellem OpenMPI o MPICH2, que és una distribució lliure, portable i molt popular. Podeu descarregar-la i trobar més informació sobre la instal·lació / utilització en la següent adreça:

http://www.mcs.anl.gov/research/projects/mpich2/

En qualsevol cas s'espera que realitzeu la vostra investigació i si cal que es debateixi al fòrum de l'assignatura.

S'espera que es produeixi debat al fòrum de l'assignatura per aprofundir en l'ús dels models de programació i els sistemes paral·lels proporcionats.

Enunciat de la pràctica

Paso 1 : Implementació i avaluació de ping-pong utilitzant MPI

La primera part d'aquesta pràctica consisteix a familiaritzar-se amb l'entorn i ser capaç de compilar i executar programes MPI. A continuació es presenten els passos necessaris per compilar i executar un simple "hello world" que realitza pas de missatges entre dos processos MPI:

- Compilar el programa `hello.c` amb mpicc (més detalls en la documentació).
- Executar el programa MPI Hello Word mitjançant "`mpirun -np NUM_PROCS ./BINARI`" .

Una qüestió clau en l'anàlisi de rendiment de programes en sistemes paral·lels amb memòria distribuïda és el cost de la comunicació en relació al de la computació.

Es demana que escriviu un programa MPI per estimar experimentalment els costos de computació i comunicació. El cost de la comunicació entre dos processadors MPI es pot dividir (aproximadament) en dues parts:

(i) Temps de d'inicialització (startup) `Tstartup` i

(ii) Temps de transmissió `Tt = (Data size)/Bandwidth`

El temps d'inicialització és el temps que necessita MPI per establir els enllaços de comunicació. El temps de transmissió és el temps necessari per enviar el missatge al seu destí. El model més comú utilitzat per quantificar el cost de la comunicació és linial:

```
Tcommunication = Tstartup + (Data size)/Bandwidth
```

La comunicació tipus "ping-pong" sovint s'utilitza per mesurar la durada de les comunicacions. En aquesta mesura, els missatges s'envien repetidament des del procés 0 al procés 1 i després de tornada al procés 0, tal com mostra en el següent codi:

```
/* Ping-Pong message */
if (myRank == 0)  {
    /* send to process 1 */
    MPI_Send(message, count, MPI_CHAR, 1, tag, MPI_COMM_WORLD);

    /* receive from process 1 */
    MPI_Recv(message, count, MPI_CHAR, 1, tag, MPI_COMM_WORLD, &status);
}  else /* myRank == 1 */
{
    /* receive from process 0 */
    MPI_Recv(message, count, MPI_CHAR, 0, tag, MPI_COMM_WORLD, &status);
    /* send to process 0 */
    MPI_Send(message, count, MPI_CHAR, 0, tag, MPI_COMM_WORLD);
 }
```

Es pot estimar `Tstartup` i l'ample de banda mitjançant l'enviament de missatges de diverses grandàries i utilitzant els mínims quadrats per ajustar una recta a les dades. El temps computacional es pot mesurar de la manera següent:

```
float x, y, z;
double start, elapsed;
start = MPI_Wtime();
z = x * y;                  // DO YOUR OWN COMPUTATION HERE
elapsed = MPI_Wtime() - start;
```

La resolució de `MPI_Wtime()` es proporciona a través de la funció MPI

```
double MPI_Wtick(void);    /* for MPI time resolution */
```

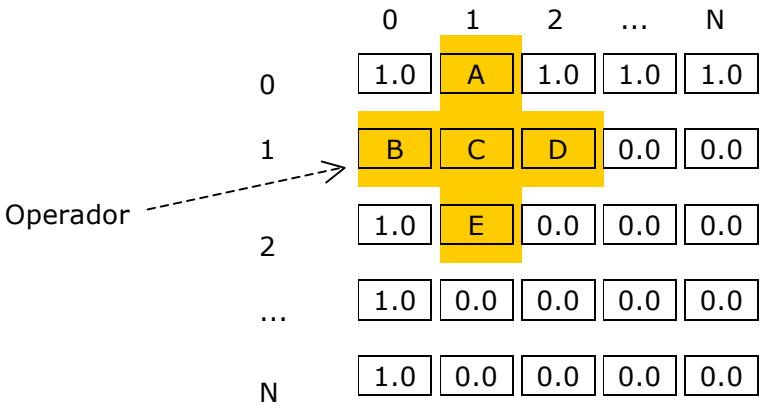
Cal tenir en compte que si el càlcul porta menys temps que la resolució, MPI_Wtime () retornarà zero. Com a resultat d'això, per obtenir els temps haureu d'executar les operacions repetides vegades i calcular el temps mitjà. Per als temps de càlcul, s'han d'utilitzar vectors grans en lloc d'escalars (variables individuals).

Sortida: El programa ha d'imprimir el cost mitjà d'un càlcul, i una sèrie de costos mitjans de comunicació: el cost d'un missatge de 0 bytes, 1 byte, 2 bytes, 4 bytes, ..., 2^{20} bytes. No hi haurà cap tipus d'entrada en el programa. Utilitzeu aquestes mesures per calcular l'ample de banda i `Tstartup`. És possible que vulgueu fer un gràfic de dispersió dels resultats. Incloeu una discussió dels resultats la qual ha d'incloure el següent:

a. Com es mesuren els temps de càlcul? Vas tenir en compte la sobrecàrrega de bucle? Per què?
b. Quin creus que és l'efecte de la crida a MPI_Wtime ()?, Per què?, Cal esbiaixar el resultat? - És a dir, què passa amb el temps dedicat a cridar MPI_Wtime ()?
c. Hi va haver gran variació en els temps de ping-pong per als missatges de mida fixa? Si és així podeu suggerir per què passa això?

Paso 2 : Implementació d'un solver mitjançant MPI

La segona part d'aquesta pràctica consisteix en implementar la versió paral·lela d'un programa seqüencial que implementa un solver. En primer lloc, el programa inicialitza una matriu i després aplica un operador sobre tota la matriu durant un determinat nombre de vegades a la recerca de convergència (vegeu la figura a continuació i els detalls de convergència en el codi proporcionat). Nota: per a cada element, l'operador aplica la fórmula:
C '= 0,2 (C + A + B + D + E).



Nota : Un programa seqüencial que implementa el solver es troba disponible en el calendari de l'assignatura .

Els passos sol·licitats en aquesta part de la pràctica són els següents:

a) Implementar el programa utilitzant MPI. Expliqueu les vostres decisions / observacions.
b) Avaluar el rendiment de l'aplicació en termes de:
   i. La seva escalabilitat respecte el nombre de nuclis utilitzats.
   ii. Desbalanceig i els overheads (per exemple , a causa del pas de missatges) .
c) Feu una comparació de diferents estratègies de partició de dades i de mides del problema (mida de la matriu). (Nota: En utilitzar un nombre major de nodes , a més de més nuclis també es disposa de més memòria. Com podeu explotar aquesta qüestió en aquest problema?)
d) Exploreu diferents tipus de crides MPI (per exemple, asíncrona o buffered) i compareu amb la seva implementació inicial (per exemple, la seva escalabilitat i overheads)

Si us plau, expliqueu les vostres observacions i proporcioneu gràfics per a l'avaluació de rendiment (almenys speedup) .

Paso 3 : Implementació del solver mitjançant MPI+OpenMP

En aquesta última part es demana que feu la implementació del mateix solver però aquesta vegada mitjançant la combinació de MPI i OpenMP. A més es demana que feu un estudi de rendiment perquè pugueu determinar i quantificar si MPI+OpenMP ofereix algun avantatge respecte la versió original en MPI.

Criteris d'avaluació

Es valorarà l'ús correcte dels models de programació MPI i OpenMP. També es tindrà en compte la correcta programació, estructura i funcionament de l'aplicació i el correcte ús dels recursos del sistema. Com a referència, respondre els tres quatre passos **correctament** serà avaluat amb una A, respondre dues amb una B, i un amb una C + o C-en funció de la els materials lliurats.

Format de lliurament

Es crearà un fitxer `tar` amb tots els fitxers font de la pràctica.

Amb la comanda següent:

```
$ tar cvf tot.tar fitxer1 fitxer2 ...
```

es crearà el fitxer "tot.tar" on s'hauran emmagatzemat els fitxers "fitxer1", fitxer2" i . . .

Per llistar la informació d'un fitxer `tar` es pot utilitzar la comanda següent:

```
$ tar tvf tot.tar
```

Per extraure la informació d'un fitxer `tar` es pot utilitzar:

```
$ tar xvf tot.tar
```

El nom del fitxer tindrà el format següent: "Cognom1Cognom2PAC2.tar". Els cognoms s'escriuran sense accents. Per exemple, l'estudiant Marta Vallès i Marfany utilitzarà el nom de fitxer següent: `VallesMarfanyPAC2.tar`

Format de lliurament

Dissabte 28 de Desembre de 2013.