Esercitazione del 01/12/04 per il corso di Calcolo Parallelo e Distribuito Anno Accademico 2004/2005 Prof. Luisa D'Amore

Punti trattati nella lezione:

1. Calcolo dello *Speed up* e dell'*efficienza* degli algoritmi che eseguono il prodotto della matrice A per vettore x in due delle strategie discusse:
   - distribuzione per righe della matrice A;
   - distribuzione per colonne della matrice A;
2. Grafici dello speed up e dell'efficienza;
3. Descizione delle routine MPI necessarie per il calcolo dei tempi delle routine elaborate durante il corso.

1) Calcolo dello *Speed up* e dell'*efficienza* degli algoritmi che eseguono il prodotto della matrice A per vettore x.

In ambiente parallelo si è elaborato un algoritmo che calcoli:

$$A \cdot x = y \;\; con \;\; A \in \Re^{n \times n}, \;\; x, y \in \Re^n$$

con A distribuita per blocchi di righe tra p processi. Calcoliamo di tale algoritmo Speed up ed efficiena definite rispettivamente da:

$$S_p = \frac{T_1}{T_p} \;\; e \;\; E_p = \frac{S_p}{p}$$

Premesse fatte sul modello utilizzato per il calcolo di S_p ed E_p:

- Indichiamo con t_{calc} il tempo ci calcolo di 1 FLOPS coincidente con 1 operazione di somma più 1 operazione di prodotto [1];
- Indichiamo con t_{calc} il tempo ci calcolo di 1 operazione f.p.;

[1] Per semplicità di espressione non è stato considerato il tempo di latenza.

- Indichiamo con $nloc = n/p$;
- Poniamo $t_{com} \approx 2 \cdot t_{calc}$.

Se la matrice A è distribuita tra i p processi per blocchi di righe di dimensione $nloc \times n$. Abbiamo che :

$$T_1 = (n \cdot n)\, t_{calc} = n^2\, t_{calc}$$

mentre per il calcolo di T_p dobbiamo sommare i tempi di calcolo $(T_p)_{calc}$ ai tempi di comunicazione $(T_p)_{com}$. Nella fase di calcolo dei sotto problemi ogni processo esegue:

```
⋮
for (i=0; i<nloc; i++)
{ yloc(i) = 0;
  for (j=0; j<n; j++)
  {
    yloc[i] += aloc[i][j] * x[j];
  }
}
⋮
```

Abbiamo quindi:

$$(T_p)_{calc} = [nloc \cdot n]\, t_{calc} = \left[\frac{n}{p} \cdot n\right] t_{calc} = \frac{n^2}{p}\, t_{calc}$$

Per ottenere il risultato finale tutti i processi spediscono a P_0 le loro $nloc$ componeti del vettore risultante. Abbiamo cosí:

$$(T_p)_{com} = [nloc \cdot (p-1)]\, t_{com} = \left[\frac{n}{p} \cdot (p-1)\right] t_{com} = \left[2 \cdot \frac{n}{p} \cdot (p-1)\right] t_{calc}$$

Si ha quindi che :

$$T_p = (T_p)_{calc} + (T_p)_{com} = \frac{n^2}{p}\, t_{calc} + \frac{2n(p-1)}{p}\, t_{calc} = \frac{n^2 + 2n(p-1)}{p}\, t_{calc}$$

Segue che:

$$S_p = \frac{T_1}{T_p} = \frac{n^2}{\frac{n^2+2n(p-1)}{p}} = \frac{p \cdot n}{n + 2p - 2}$$

ed

$$E_p = \frac{T_p}{p} = \frac{n}{n + 2p - 2}$$

Se la matrice A è distribuita tra i p processi per blocchi di colonne di dimensione $n \times nloc$ ed il vettore x per blocchi di righe di lunghezza $nloc$, abbiamo che:

$$T_1 = (n \cdot n)\, t_{calc} = n^2\, t_{calc}$$

mentre per il calcolo di T_p dobbiamo sommare i tempi di calcolo $(T_p)_{calc}$ ai tempi di comunicazione $(T_p)_{com}$. Nella fase di calcolo dei sotto problemi ogni processo esegue:

```
⋮
for (i=0; i<n; i++)
{ yloc(i) = 0;
  for (j=0; j<nloc; j++)
  {
    yloc[i] += aloc[i][j] * x[j];
  }
}
⋮
```

Abbiamo quindi:

$$(T_p)_{calc} = [n \cdot nloc]\, t_{calc} = \left[n \cdot \frac{n}{p}\right] t_{calc} = \frac{n^2}{p}\, t_{calc}$$

Per ottenere il risultato finale tutti i processi spediscono a P_0 le loro n componeti parziali del vettore risultante. Abbiamo cosí:

$$(T_p)_{com} = [n \cdot (p-1)]\, t_{com} = [2 \cdot n \cdot (p-1)]\, t_{calc}$$

Il processore P_0 deve ora calcolare le componenti del vettore risultante, sommando termine a termine le compontenti dei vettori ricevuti. Dobbiamo

quindi aggiornare $(T_p)_{calc}$ con queste nuove operazioni:

$$(T_p)_{calc} = \frac{n^2}{p}\, t_{calc} + [n(p-1)]\, t_{calc}$$

Si ha quindi che :

$$T_p = (T_p)_{calc} + (T_p)_{com} = \left[\frac{n^2}{p} + n(p-1) + 2n \cdot (p-1)\right] t_{calc} = \left[\frac{n^2 + 3n(p-1)p}{p}\right] t_{calc}$$

Segue che:

$$S_p = \frac{T_1}{T_p} = \frac{n^2}{\frac{n^2+3n(p-1)p}{p}} = \frac{p \cdot n}{n + 3p(p-1)}$$

ed

$$E_p = \frac{T_p}{p} = \frac{n}{n + 3p(p-1)}$$

2) Grafici dello *Speed up* e dell'*efficienza*
Sono stati tracciati i grafici ideali dello *Speed up* e dell'*efficienza*. Si sono poi considerati degli ipotetici grafici relativi ai software realizzati con conseguente estrapolazione delle informazioni che tali grafici forniscono.

3) Descizione delle routine MPI necessarie per il calcolo dei tempi delle routine elaborate durante il corso.
Si è descritta le routine:

$$double\ MPI\_Wtine()$$

che permette di calcolare il tempo di esecuzione di una routine, restituendo in double il tempo in sencondi. Nel discutere il problema della sincronizzazione dei processi si è arrivati ad estrapolare il seguente schema:

⋮
MPI_Barrier(MPI_Comm comm)
t_1=MPI_Wtine()
Chiamata alla routine di cui prendere i tempi
MPI_Barrier(MPI_Comm comm)
t_2=MPI_Wtine()
t=$t_2 - t_1$
calcolo del massimo tra i tempi t degli nproc processi (= T_p)
⋮