Université Pierre et Marie Curie
MASTER d'informatique 2^e année
Novembre 2011

Examen du module Programmation Parallèle

Documents autorisés : polycopiés de cours et de TD

Tous les systèmes nécessitant de l'électricité sont interdits.

Le barême est donné à titre indicatif
Durée 2 heures

Calcul de la plus grande valeur propre d'une matrice par la méthode de la puissance itérée

Considérons une matrice carrée A de dimension n à coefficients réels.

$$A = (a_{ij}), \quad a_{ij} \in \mathbb{R},\ i = 1, ..., n;\ \ j = 1, ..., n.$$

On appelle λ_1, λ_2,..., λ_n les valeurs propres de cette matrice et on suppose que:

$$| \lambda_1 | > | \lambda_2 | > ... > | \lambda_n |$$

L'algorithme suivant permet alors de calculer $\rho = | \lambda_1 |$.

Algorithme:
a- Choisir un vecteur $X^{(0)}$ arbitraire de norme 1.
b- Calculer les vecteurs itérés:

$$X^{(k+1)} = \frac{AX^{(k)}}{\|AX^{(k)}\|}$$

c- Arrêt des itérations lorsque: $\|X^{(k)} - X^{(k-1)}\| \leq \epsilon$, ϵ étant une quantité positive fixée à l'avance.
NB:
- La norme d'un vecteur X est définie par: $\|X\| = Sup_{i=1}^{n} \mid x_i \mid$.
- L'exposant entre parenthèses est le numéro d'iteration.
On peut alors montrer que lorsque k tend vers l'infini, la quantité $\rho_k = \|AX^{(k)}\|$ tend vers $\rho = | \lambda_1 |$.

Implémentation en OpenMP (5 points)

La fonction `puissance` qui implémente la méthode de la puissance itérée est donnée en annexe. Lors de l'appel de fonction, le vecteur initial est déjà initialisé.

Question 1

Ecrivez une version parallèle utilisant OpenMP de la fonction `puissance`. On considérera que le nombre de threads est précisé par la variable d'environnement `OMP_NUM_THREADS`.

Question 2

Proposez une autre solution pour paralléliser avec OpenMP les lignes de programme 18 à 21. Expliquer les différences de comportement avec votre première proposition.

Implémentation MPI (10 points)

Pour implanter cet algorithme de façon parallèle on dispose d'une machine parallèle à p processeurs dont la mémoire est entièrement distribuée. On suppose aussi que p est plus petit que la dimension n de la matrice.
On dispose d'une fonction `liretab` dont le prototype est :
`void liretab(char *s, double *a, int *n )`.
`s` : nom du fichier qui contient la matrice, `a` pointeur vers les valeurs de la matrice, `*n` : dimension de la matrice.
La fonction alloue le tableau `a` en fonction du nombre de valeurs contenu dans le fichier.

Question 3

Expliquez comment les données vont être réparties sur les processeurs.

Question 4

Ecrivez soit un pseudo-code, soit une explication détaillée de l'implémentation que vous proposez.

Implémentation hybride (5 points)

Question 5

Quels sont les avantages et les inconvénients d'une implémentation hybride (MPI+OpenMP) de cet algorithme.

Annexe

Dans la fonction puissance, `EPS` correspond à la valeur de ϵ donnée dans l'algorithme et `MAX`) la dimension maximale des vecteurs.

```
1  double puissance(double *a, double *v, int n )
2  {
3    int i,j,k;
4
5    double vn[MAX],s,max=0;
6    double norme=1;
7
8    while  (norme > EPS){
9
10     for(i=0;i<n;i++){
11       s=0.;
12       for (j=0;j<n;j++)
13         s+=a[i*n+j]*v[j];
14       vn[i]=s;
15     }
16
17     // recherche du max
18     max=0;
19     for(i=0;i<n;i++)
20       if (max < fabs(vn[i]))
21         max=fabs(vn[i]);
22
23     // division par le max
24     for(i=0;i<n;i++)
25       vn[i]=vn[i]/max;
26
27
28     // calcul de la norme |Xk -X(k-1)|
29     norme=0.;
30     for(i=0;i<n;i++){
31       double diff=fabs(vn[i]-v[i]);
32       if (norme < diff)
33         norme=diff;
34       v[i]=vn[i];
35     }
36   }
37   return max;
38 }
```