Maîtrise d'informatique
Module de Programmation parallèle

Examen

Philippe MARQUET

Juin 2001

Tous documents autorisés
Durée : 3h

Important Plusieurs questions vous demandent d'expliquer/décrire/motiver l'énoncé d'un algorithme, la parallélisation d'un problème... Ne les négligez pas ; ces questions sont notées ! Bien entendu, des réponses concises et précises sont appréciées.

1 Résolution de systèmes triangulaires supérieurs en MPI

Nous allons proposer une parallélisation à l'aide de la bibliothèque de passage de messages MPI de la phase de remontée, dernière étape de la résolution d'un système linéaire, après une triangularisation de Gauss ou une décomposition de Cholesky.

Nous allons résoudre le système triangulaire supérieur (STS)

$$Ux = b$$

avec

$$U = \begin{pmatrix} U_{1,1} & U_{1,2} & \dots & U_{1,N} \\ 0 & U_{2,2} & \dots & U_{2,N} \\ 0 & \dots & \dots & \dots \\ 0 & 0 & \dots & U_{N,N} \end{pmatrix} \qquad \text{et} \qquad U_{i,i} \neq 0 \quad \text{pour tout} \quad 1 \leq i \leq N.$$

On a

$$b_i = \sum_{j=i}^{N} U_{i,j}.x_j \qquad \text{pour} \quad i = N, N-1, N-2, ..., 1.$$

On procède donc par remontée en calculant successivement

$$x_N = \frac{b_N}{U_{N,N}} \qquad \text{puis} \qquad x_i = \frac{b_i - \displaystyle\sum_{j=i+1}^{N} U_{i,j}.x_j}{U_{i,i}} \qquad \text{pour} \quad i = N-1, N-2, ..., 1.$$

Un code séquentiel pour ce calcul est le suivant :

```
x[N] = b[N] / U[N,N];
for (i=N-1 ; i >= 1 ; i--) {
   acc = 0;
   for (j=i+1 ; j <= N ; j++) {
      acc += U[i,j] * x[j];
   }
   x[i] = (b[i] - acc) / U[i,i] ;
}
```

Id : xam01-juin.tex,v 1.3 2001/06/14 13 :25 :54 marquet Exp

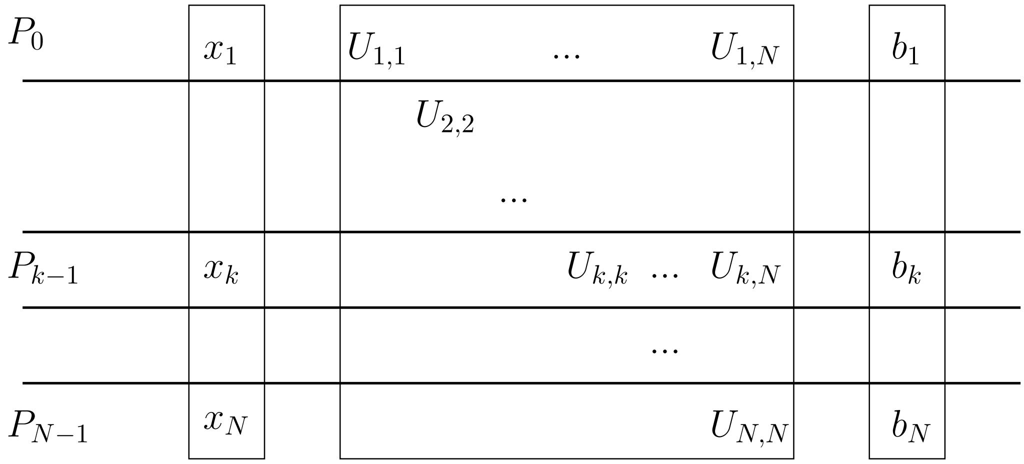

FIG. 1 – Distribution des tableaux pour la résolution du système linéaire triangulaire supérieur

Nous travaillons avec N processus : P_0 à P_{N-1}. Les tableaux sont distribués par lignes sur ces processus selon le schéma de la figure 1.

Initialement,le processus P_0 détient U et b (qu'il initialise par un appel à `sts_lire (U, b)`) ; il les distribue aux autres processus. À l'issue du calcul, P_0 centralisera les résultats (qu'il sauvegardera par un appel à `sts_ecrire (x)`).

Exercice 1 (Initialisation et terminaison)
Donner les codes C/MPI ou Fortran/MPI des phases de diffusion des valeurs initiales aux processus et de collecte du résultat par le processus P_0.

Il sera nécessaire de préciser les tableaux devant être alloués localement par chacun des processus.

Exercice 2 (Schéma de communications)
Identifier ce que doit recevoir le processus P_p pour mener à bien son calcul local à l'étape t ; de quel(s) processus il doit recevoir ces données. En conséquence, identifier ce que doit envoyer le processus P_p, et à quel(s) processus.

Exercice 3 (Résolution de STS)
Donner le code C/MPI ou Fortran/MPI de la résolution d'un STS.

Exercice 4 (Nouvelle(s) distribution(s) des données)
De part la distribution des données proposée, il appert un déséquilibre de travail entre les processus. Proposer des alternatives pour palier à ce déséquilibre. Identifier, si cela est possible, un recouvrement des communications par des calculs.

2 Différences finies en mémoire partagée

Nous allons implanter le problème dit des différences finies sur une machine multiprocesseurs à mémoire partagée à l'aide de processus légers. Nous travaillons au choix avec la bibliothèque `pthread` ou avec OpenMP.

On considère un domaine à deux dimensions de taille $N \times N$. La valeur au point i, j au temps $t + 1$ est calculée de la manière suivante :

$$X_{i,j}^{(t+1)} = \frac{4X_{i,j}^{(t)} + X_{i-1,j}^{(t)} + X_{i,j-1}^{(t)} + X_{i+1,j}^{(t)} + X_{i,j+1}^{(t)}}{8}$$

La résolution du problème se termine quand en tout point la différence entre la valeur de $X^{(t)}$ et la valeur de $X^{(t-1)}$ au temps précédent est inférieure, en valeur absolue, à une valeur ε donnée.

Exercice 5 (Organisation pour les différences finies parallèles)
Donner une organisation de processus légers qui peut être mis en œuvre pour la résolution du problème des différences finies 2-D sur une machine à mémoire partagée.

Exercice 6 (Programme des différences finies parallèles)
Donner le code d'un programme implantant votre proposition énoncée à l'exercice précédent.

3 Annexe

Pourvu que vous en définissiez précisément le comportement, vous pouvez introduire nombre de fonctions utiles qui ne sont pas en rapport direct avec les aspects centraux des développements à réaliser ici. Par exemple :

`lire_X0(double **X)` initialise la valeur du domaine 2-D X.

`double max_erreur(double **d1, double **d2)` retourne la valeur maximale de la différence élément à élément entre les domaines 2-D $d1$ et $d2$.

Fin du sujet