Algorithmique et Programmation Parallèle
TD1 OpenMP

romain.pereira@inria.fr, hugo.taboada@cea.fr, marc.perache@cea.fr

Février 2023

Rappel

Compilation et exécution de programme OpenMP

Pour compiler un programme C utilisant OpenMP avec le compilateur gcc:

```
gcc -fopenmp source.c -o monprog
```

La variable d'environnement `OMP_NUM_THREADS` permet de spécifier le nombre de threads OpenMP

```
# pour définir une variable d'environnement au lancement:
OMP_NUM_THREADS=4 ./monprog

# pour définir une variable d'environnement entre plusieurs lancement:
export OMP_NUM_THREDS=4
./monprog
```

Utilisation du cluser de l'école

Se connecter

```
ssh prenom.nom@hpc.pedago.ensiie.fr
```

Lancer un job sur 1 noeud

```
srun -p calcul -N 1 ./program
```

Envoyer un fichier via FTP dans son home cluster

```
scp MON-FICHIER prenom.nom@hpc.pedago.ensiie.fr:~
```

Récupérer un fichier via FTP vers son home local

```
scp prenom.nom@hpc.pedago.ensiie.fr:~ ~/MON-FICHIER
```

Exercice 1 : Passage de OpenMP à pthread

Question 1

Étudier, compiler et exécuter avec 4 threads le programme `omp_omp2pth/prog_omp.c`. Quelle doit être la valeur de la variable `sum` à la fin de l'exécution ?

Question 2

Réécrire ce programme avec les threads POSIX.

Question 3

Revenir sur le programme OpenMP. La boucle `for` est-elle parallélisée ? Modifier le programme pour que la boucle `for` soit distribuée sur tous les threads OpenMP. Quelle est la valeur attendue de la variable `sum` ?

Question 4

OpenMP propose t'il un autre moyen plus simple et efficace pour calculer `sum` ? Si oui, effectuer la modification.

Exercice 2 : Produit matrice-creuse vecteur

Une matrice creuse est une matrice dont « la plupart » des éléments sont nuls. Dans notre cas, la matrice a au plus 5 élément non nuls sur chaque ligne. Pour compresser au maximum les données, on ne stocke que ces éléments via la structure `sparse_matrix_t`. Pour une ligne `i` on a :

- le nombre d'éléments non nuls: `ncol[i]`
- le numéro des colonnes non-nuls: `col[i][k]` avec $0 \leq k < ncol[i]$
- les éléments non nuls: `elt[i][k]` avec $0 \leq k < ncol[i]$

$$\underbrace{\begin{bmatrix} 0 & 1.2 & 0 & 0 & 0 & 0 \\ 0.3 & 0 & 0 & -6. & 0 & 0 \\ 0 & 0 & 2.7 & 0 & 0 & 0 \\ 0 & -3. & 0 & 0 & 9.1 & 5.2 \\ 0 & 0 & 0 & 4. & 0 & 0 \\ 0 & 0 & -1. & 0 & 0.8 & 0 \end{bmatrix}}_{\text{Matrice}} \quad \underbrace{\begin{matrix} 1 \\ 2 \\ 1 \\ 3 \\ 1 \\ 2 \end{matrix}}_{\text{ncol}} \quad \underbrace{\begin{matrix} 1.2 & & \\ 0.3 & -6. & \\ 2.7 & & \\ -3. & 9.1 & 5.2 \\ 4. & & \\ -1. & 0.8 & \end{matrix}}_{\text{elt}} \quad \underbrace{\begin{matrix} 1 & \\ 0 & 3 \\ 2 & \\ 1 & 4 \quad 5 \\ 3 & \\ 2 & 4 \end{matrix}}_{\text{col}}$$

Question 1

Paralléliser la fonction `prod_mat_vec` en veillant à l'équilibrage de charge.

Question 2

Paralléliser la fonction `is_equal`

Question 3

Relever les temps elapsed pour 1, 2, 4 et 8 threads OpenMP pour les paramètres $N = 1000000$ et $niter = 100$. Commenter.