Algorithmique et Programmation Parallèle

TD 1 OMP

Exercice 1 : Passage de OpenMP à Pthread

- 1. Étudier, compiler et exécuter avec 1, 2 puis 4 threads le programme omp_omp2pth/prog_omp.c. Quelle doit être la valeur de la variable sum à la fin de l'exécution ?
- 2. Réécrire ce programme avec les threads POSIX.
- 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 ?
- 4. OpenMP propose t'il un autre moyen plus simple et plus 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 éléments non nuls sur chaque ligne.

Pour compresser au maximum les données, on ne stocke que ces éléments (structure sparse matrix t). Pour une ligne i, on a :

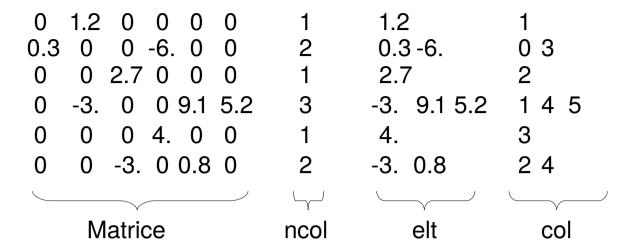
- le nombre d'éléments non nuls (i.e. le nombre de colonnes « utiles ») : ncol[i]
- le numéro des colonnes « utiles » : col[i][k] où $0 \le k \le ncol[i]$
- les éléments non nuls : elt[i][k] où 0 <= k < ncol[i]

On veut paralléliser le produit d'une matrice creuse par un vecteur. Pour mesurer le coût on effectue ce produit plusieurs fois :

./monprog <N> <niter>

où N est la dimension de la matrice NxN (ou la taille du vecteur)

niter : nombre de répétitions du produits



- 1. Paralléliser la fonction prod_mat_vec en veillant à l'équilibrage des charges.
- 2. Paralléliser la fonction is_equal .
- 3. Relever les temps elapsed pour 1, 2, 4 et 8 threads OpenMP pour les paramètres suivants : N = 1000000 et niter = 100. Commenter.