8INF856
Programmation sur architectures parallèles
Devoir 2

(À remettre au plus tard lundi le 28 octobre 2024)

- **Vous pouvez travailler en équipe de 2 ou 3.**
- **Vous serez évalué pour la rectitude et la qualité de vos réponses.**
- **Vous devez respecter scrupuleusement les instructions.**

1. Dans cet exercice vous devez implémenter l'algorithme récursif parallèle de tri par fusion que nous avons vu en classe ainsi que l'algorithme séquentiel standard. Vos programmes devront être écrits en C et vous devrez utiliser la librairie pthread pour la version parallèle.

   L'entrée sera donnée dans un fichier ayant la forme suivante: un entier n suivit de n autres entiers que vous placerez dans un tableau T. Une fois trié, vous devez afficher le tableau trié. Lorsque $n > 1000$, n'affichez que les 100 premiers et les 100 derniers éléments.

   Testez et analysez vos programmes. Le programme parallèle doit être testé et analysé avec 1, 2, 4, 8, 16, 24 et 48 threads. Dans chaque cas, indiquez à partir de quelle valeur de n votre solution parallèle est plus efficace que l'algorithme séquentiel (ne comptez pas le temps nécessaire pour initialiser T). Comparez aussi votre programme parallèle en fonction du nombre de threads utilisés. Quelle est le nombre de threads optimal? Expliquez vos résultats.

   Je vous demande de procéder de la façon suivante:

- Dans votre répertoire maison sur `dim-openmpi0.uqac.ca` , vous devez créer un répertoire dont le nom est Devoir2
- Vos deux programmes devront être dans des fichiers d2s.c (séquentiel) et d2p.c (parallèle).
- Vos programmes doivent pouvoir être compilés sans faute à l'aide de la commande gcc. Pour votre programme parallèle utilisez la commande suivante: `gcc d2p.c -lpthread`
- Vous devez déposer sur Moodle un document PDF contenant les informations suivantes:
  (a) Le nom des coéquipiers
  (b) Le nom d'usager et le mot de passe du compte où se trouve le programme
  (c) Le résultat des tests ainsi que l'analyse.

Vos programmes seront testés à l'aide de la commande

```
.\a.out < fichier_test > resultat
```

où `a.out` est l'exécutable et `fichier_test` est un fichier contenant un entier n suivi de n autres entiers et `resultat` est le résultat du tri. La valeur maximale de n que j'utiliserai sera $2\,000\,000\,000$.

2. Refaites l'exercice précédent en utilisant OpenMP plutôt que la librairie pthread. Le nom de votre programme parallèle doit être `d2omp.c` et il doit être placé dans le même répertoire que celui indiqué à la question précédente. Votre programme doit pouvoir être compilé à l'aide de la commande `gcc -fopenmp d2omp.c`

   Remarque: Je vous suggère aussi d'utiliser la fonction `omp_get_wtime()` pour déterminer le temps d'exécution d'un thread. Il pourrait être aussi utile d'utiliser openMP avec votre programme séquentiel afin de chronométrer son temps d'exécution sur la même base que l'algorithme parallèle.