

TDTP5 : OpenMP

Version du 28 février 2020

Exercice 1 – Calcul de π

1. Ecrire un programme C qui calcule une valeur approchée de π à l'aide de la formule suivante (pour une valeur de N fixée, par exemple $N = 1000000000$) :

$$\pi \approx \frac{1}{N} \sum_{i=0}^N \frac{4}{1 + (\frac{i}{N})^2}$$

2. Paralléliser ce code en OpenMP de trois façons différentes. Quelle version vous semble la plus efficace ?

Exercice 2 – Stockage contigu

1. On dispose d'un grand nombre de threads $NBT = 1024$ qui effectuent tous un calcul donné. A l'issue de ce calcul, seuls certains threads devront indiquer qu'ils ont pu mener à bien ce calcul et inscrire pour cela leur rang dans un tableau (supposé suffisamment grand). On ne connaît pas à l'avance les threads qui vont devoir faire cette écriture mais on souhaite que toutes les écritures soient contigues (dans un ordre quelconque).

Ecrire de deux façons différentes un programme OpenMP qui réalise cette écriture.

Exercice 3 – Produit matriciel

On considère un code de produit matriciel du type $C = A \times B$ pour des matrices carrées d'ordre N .

1. Ecrivez le code parallèle OpenMP correspondant en justifiant vos choix (type d'équilibrage de charge retenu, choix de la (ou des) boucle(s) parallèle(s) traitée(s) par OpenMP ...).

Exercice 4 – Calcul de fractales

On considère le calcul de fractales basé sur la construction de l'ensemble de Mandelbrot du TP 2–3.

1. Ecrivez le code parallèle OpenMP correspondant en justifiant vos choix (type d'équilibrage de charge retenu, choix de la (ou des) boucle(s) parallèle(s) traitée(s) par OpenMP ...).

Exercice 5 – Diffusion de la chaleur

On considère la simulation de diffusion de la chaleur du TP précédent.

1. Ecrivez le code parallèle OpenMP correspondant en justifiant vos choix (type d'équilibrage de charge retenu, choix de la (ou des) boucle(s) parallèle(s) traitée(s) par OpenMP ...).

Exercice 6 – MPI et OpenMP

On s'intéresse ici à une programmation parallèle hybride MPI-OpenMP.

1. (**Support des threads avec MPI (en TP)**) A l'aide du manuel MPI-2.2, déterminez le niveau du support des threads de l'implémentation MPI dont vous disposez.
2. (**Diffusion de la chaleur**) En reprenant la version parallèle MPI de `heatsink.c` (**sans** recouvrement des communications par le calcul), écrivez une version parallèle hybride MPI-thread qui respecte le niveau de support des threads offert.

Pour un nombre de processeurs fixé, chaque processeur disposant d'au moins 2 cœurs, quel est l'avantage en théorie de cette version MPI-thread par rapport à la version précédente dite "MPI pur" ?

Vérifiez-le avec des tests de performance en utilisant par exemple deux PC multicœurs.

Aide : pour lancer avec `mpirun` (de l'implémentation OpenMPI) un processus MPI en positionnant certaines variables d'environnement, vous pouvez utiliser par exemple :

```
mpirun -x OMP_NUM_THREADS=2 -x OMP_SCHEDULE="static,64" -n 2 ./a.out ...
```

Exercice 7 – En TP

1. (**Nombre de cœurs disponibles**) Vérifiez à l'aide de la commande `cat /proc/cpuinfo`, le nombre de cœurs disponibles sur votre machine.
2. (**Premier programme**) Utilisez un des programmes du cours pour tester votre environnement. N'oubliez pas de positionner la variable `OMP_NUM_THREADS` !
Rappel pour la compilation : `gcc -fopenmp toto.c -o toto`
3. (**Efficacités parallèles**) Après avoir récupéré les codes sources `matmul.c` (à utiliser avec des matrices carrées d'ordre $N = 512$), `mandel.c` et `heatsink.c`, calculez l'efficacité parallèle de ces programmes en utilisant tous les cœurs physiques de votre machine. On pourra tester différents équilibrages de charge (via la clause `schedule(runtime)` et la variable d'environnement `OMP_SCHEDULE`), ainsi que différentes tailles de blocs (*chunks*) pour chaque type d'équilibrage de charge, afin de vérifier la pertinence de choix faits précédemment.