Projet Informatique
S1825
2020 - 2021

claude.tadonki@mines-paristech.fr

Objectif: Ecrire un programme « multi-threadé » et « vectoriel » en C, puis comprendre les divers aspects liés à son fonctionnement correct et à ses performances.

Remarque: Tous les codes doivent être écrits en langage C. L'écriture du programme se fera par binôme. A remettre au plus tard le lundi 15 décembre 2020 par mail.

Exercice 1: Ecrire la fonction
float norm(float *U, int n)
qui calcule

$$d(u) = \sum_{i=0}^{N-1} \sqrt{|u_i|}$$

pour un vecteur U de n réels.

Exercice 2: Ecrire une version vectorielle (AVX) de la fonction précédente en supposant que l'adresse de U est alignée (*on supposera que n est un multiple de 8*).
float vect_norm(float *U, int n);

Exercice 3: Proposer une version multithreadée de la fonction d(u)
void normPar(float *U, int n, int mode, int nb_threads);
qui considère une exécution avec nb_threads threads. On peut par exemple supposer que chaque thread traitera un block du vecteur U (partitionnement statique). Le paramètre *mode* indique le type de calcul (0 : scalaire et 1 : vectoriel). Pensez à créer une structure pour les données du thread.

Exercice 4: Ecrire la fonction principale de votre projet
Le main() doit

a) créer et initialiser le vecteur U avec des valeurs aléatoires comprises en 0 et 1
b) calculer d(U) (version de base et version multithreadée + vectorielle)
c) afficher l'accélération par rapport au temps d'exécution
d) permettre une execution de votre programme sous la forme
./program n nbthreads
où *n* est la taille du vecteur et *nbthreads* le nombre de threads.

Exercice 5: Indiquez (dans votre fichier sous forme de commentaire) comment on pourrait adapter la fonction vect_norm() pour qu'elle puisse fonctionner

- même avec une adresse U non alignée
- avec n quelconque (non nécessairement multiple de 8)

Remarque: N'hésitez pas à vous inspirer des codes de TP
www.hemisphere-education.net/cours/s1825