IFT630 - Projet 3

Violette PAULIN – PAUM1202
Violette.Paulin@USherbrooke.ca

Build et test

Une fois le dossier `build` créé, on peut build directement en exécutant `make all`. Pour tester MPI, `make runmpi`. Pour tester OpenMP, `make runomp`. Ces commandes créent des fichiers `OutX.txt`. Ceux-ci montrent la clef testée à gauche, et le temps pour la trouver à droite, séparée par un ':'.

Performance

Je ne comprends pas la question de mesure de performance. Dans mon cas, mesurer les performances revient à mesurer la totalité du temps écoulé, ce qui va comprendre une part non négligeable d'allocation, ainsi que l'overhead des différentes librairies. Alors que si je mesure le temps moyen pour trouver une clef, je mesurerais la même chose dans les deux cas. Il m'est donc impossible de pouvoir tirer une conclusion satisfaisante sur les performances, pour mon implémentation. J'ai quand même fait le choix de mesurer le temps pour trouver une clef, sans prendre en compte l'allocation.

Cependant, OpenMP peut marcher sur tous les cœurs logiques (thread) de ma machine, alors que MPI ne fonctionne qu'avec les cœurs physiques (il me semble). Ainsi, on gagne théoriquement en temps global avec OpenMP.

De plus, je n'ai pas utilisé OpenMP de la manière la plus optimale, faute de temps. A la place de diviser sur une boucle, je l'ai divisé comme je l'ai fait pour MPI. Ceci coute plus de temps en allocation, et l'on est contraint à utiliser plusieurs fichiers

Dans les deux cas, on s'aperçoit que plus une clef est loin, plus elle est difficile à charger. C'est le comportement attendu. De plus, les temps semblent être linéaire, ce qui est encore une fois attendu.