IN203 Projet - Simulation d'une colonisation d'une galaxie

Qingchen YANG

-Des éventuels conflits mémoires et pourquoi ils ne posent pas de problème.

Dans certains cas, lors de l'utilisation de la mémoire partagée, différents threads voudront modifier les mêmes données en même temps, provoquant des conflits, par exemple quand on utilise <#pragma omp parallel for> dans les fichiers galaxie.cpp. À ce stade, chaque fois que nous modifions des données, nous utilisons d'abord < #pragma omp critical > pour nous assurer que l'opération de modification est atomique.

Dans d'autres cas, comme dans colonisation.cpp, nous créons deux threads, un pour l'affichage, un pour la mise à jour, et nous utilisons <#pragma omp parallel num\_threads(2)>. Dans ce cas, nous utilisons omp\_get\_thread\_num() pour déterminer le numéro de chaque processus et les laisser effectuer leurs tâches pour éviter les conflits.

-Le calcul des accélérations (speedup) obtenus

En mémoire partagée :

Speedup = 50.919/47.176=1.079

En mémoire distribuée avec 5 processus :

Speedup = 50.919/56.359=0.903 (ici ce n’est pas plus optimal, peut-être c’est à cause de la complexité de notre structure de mémoire distribuée)