PROJET IN203

Note :

Les choses rajoutées dans le code sont dans ces balises :

//&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

Code rajouté […]

//&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

Mesure du temps

Temps passé dans la simulation par pas de temps avec affichage :

Temps moyen = 0,078 secondes

Temps passé dans la simulation par pas de temps sans affichage :

Temps moyen = 0,021 secondes

En moyenne on a un rapport 3,7 entre la simulation avec affichage et la simulation sans affichage

On trouve en mesurant avec un :

start = std::chrono::system_clock::now();

et un end = std::chrono::system_clock::now();

autour de if (affiche) afficheSimulation(écran, grille, jours_écoulés);

correspondant à la partie affichage un temps moyen passé à l'affichage (cf données dans les fichiers textes puis convertis en Excel pour faire les moyennes) de :

Temps moyen = 0,059 secondes

Ce temps est très proche de la différence des deux temps précédents correspondant finalement au temps d'affichage également

Temps à l'affichage = Temps avec affichage – Temps sans affichage

= 0,078 - 0,021 = 0,057 secondes

On constate donc des résultats proches dans ces deux résultats : écart d'environ 3% qui est donc très faible.

C'est donc bien cohérent.

Parallélisation affichage contre simulation

Temps moyen pour la simulation par pas de temps :

Temps = 0,031 secondes

On a donc un speedup de $\frac{0,078}{0,031}$ = 2,52

On a ici :

Temps moyen par pas de temps à l'affichage : 0.0366844 secondes

Temps moyen par pas de temps en simulation : 0.0364071 secondes

C'est deux temps sont proches et c'est logique puisque le processus simulation doit attendre l'acquittement du processus affichage avec l'utilisation de l'envoie par MPI_Send.

Parallélisation affichage asynchrone contre simulation

Temps moyen par pas de temps à l'affichage : 0.0120739 secondes

Temps moyen par pas de temps en simulation : 0.0115636 secondes

L'écart par rapport à la question précédente est dû aux échanges de messages avec MPI qui rallonges les délais (attente d'acquittement).

Parallélisation OpenMP

Temps moyen par pas de temps en simulation : 0.00693211 secondes

Temps moyen par pas de temps à l'affichage : 0.00742756 secondes

En prenant un nombre d'individus global constant :

Nombre de threads	Temps	Speedup
1	0,0120	1,0000
2	0,0077	0,6382
3	0,0056	0,4648
4	0,0055	0,4570
5	0,0036	0,2974
8	0,0036	0,3032

En prenant un nombre d'individus constant par thread, fait avec des tailles de paquets arbitraire de 20 000 individus (puis essais avec d'autres tailles de paquet) :

Nombre de threads	Temps	Speedup
1	0,0121	1,0000
2	0,0071	1,7006
3	0,0050	2,3958
4	0,0043	2,8274
5	0,0036	3,3070
8	0,0038	3,1741

On peut donc voir que bien que les speedups soient proches, il vaut mieux garder un nombre d'individu global constant pour que les tailles des paquets soient déterminé de façon plus optimale.

Parallélisation MPI de la simulation

Nombre de threads	Temps	Speedup
1	0,0120	1,0000
2	0,0117	1,0209
3	0,0076	1,5824
4	0,0065	1,8487
5	0,0065	1,8432
6	0,0061	1,9576
7	0,0065	1,8348
8	0,0065	1,8587

On peut voir ici qu'avec du parallélisme distribué, le speedup est moins bon qu'avec OpenMP (en parallélisation en mémoire partagée).

Néanmoins, on peut combiner les deux pour augmenter le speedup, ce qui sera l'objet de la question suivante.

Parallélisation finale

Il ne m'a pas été possible de réaliser le test avec plusieurs pcs sur un réseau (un seul à ma disposition et à la fin des vacances). Néanmoins c'est avec cette configuration (en utilisant conjointement OpenMP et MPI) qu'on s'attend à obtenir les meilleurs résultats.

Bilan :

Ce projet m'a permis de contextualiser toutes les choses vues en cours et de le mettre en application. Cela m'a également replongé dans les nuances du code C++.

Je pense avoir mieux compris les enjeux d'OpenMP et de MPI et notamment de leurs utilisations conjointes avec ce projet.

Annexe parallélisation avec OpenMP :

boucle est sectionnée avec nombre d'individus global constant

Temps moyen par pas de temps à l'affichage : 0.0126324 secondes avec 1 threads

Temps moyen par pas de temps en simulation : 0.0119949 secondes avec 1 threads

Temps moyen par pas de temps à l'affichage : 0.00776747 secondes avec 2 threads

Temps moyen par pas de temps en simulation : 0.00765518 secondes avec 2 threads

Temps moyen par pas de temps en simulation : 0.00557473 secondes avec 3 threads

Temps moyen par pas de temps à l'affichage : 0.00579663 secondes avec 3 threads

Temps moyen par pas de temps à l'affichage : 0.00586508 secondes avec 4 threads

Temps moyen par pas de temps en simulation : 0.00548156 secondes avec 4 threads

Temps moyen par pas de temps à l'affichage : 0.00390331 secondes avec 5 threads

Temps moyen par pas de temps en simulation : 0.00356693 secondes avec 5 threads

Temps moyen par pas de temps à l'affichage : 0.00382557 secondes avec 6 threads

Temps moyen par pas de temps en simulation : 0.00366196 secondes avec 6 threads

Temps moyen par pas de temps en simulation : 0.00363553 secondes avec 7 threads

Temps moyen par pas de temps à l'affichage : 0.00374661 secondes avec 7 threads

Temps moyen par pas de temps à l'affichage : 0.00385907 secondes avec 8 threads

Temps moyen par pas de temps en simulation : 0.00363627 secondes avec 8 threads

boucle est sectionnée en itérations de tailles fixes 20000 individus

Temps moyen par pas de temps en simulation : 0.0120541 secondes avec 1 threads

Temps moyen par pas de temps à l'affichage : 0.0126032 secondes avec 1 threads

Temps moyen par pas de temps en simulation : 0.00708822 secondes avec 2 threads

Temps moyen par pas de temps à l'affichage : 0.00777802 secondes avec 2 threads

Temps moyen par pas de temps à l'affichage : 0.0050911 secondes avec 3 threads

Temps moyen par pas de temps en simulation : 0.00503134 secondes avec 3 threads

Temps moyen par pas de temps à l'affichage : 0.00458647 secondes avec 4 threads

Temps moyen par pas de temps en simulation : 0.00426331 secondes avec 4 threads

Temps moyen par pas de temps à l'affichage : 0.00387918 secondes avec 5 threads

Temps moyen par pas de temps en simulation : 0.00364504 secondes avec 5 threads

Temps moyen par pas de temps en simulation : 0.00375059 secondes avec 6 threads

Temps moyen par pas de temps à l'affichage : 0.00395874 secondes avec 6 threads

Temps moyen par pas de temps à l'affichage : 0.00344692 secondes avec 7 threads

Temps moyen par pas de temps en simulation : 0.00337485 secondes avec 7 threads

Temps moyen par pas de temps à l'affichage : 0.0038305 secondes avec 8 threads

Temps moyen par pas de temps en simulation : 0.00379767 secondes avec 8 threads

Temps moyen par pas de temps à l'affichage : 0.00384574 secondes avec 9 threads

Temps moyen par pas de temps en simulation : 0.00378098 secondes avec 9 threads

Temps moyen par pas de temps à l'affichage : 0.00390129 secondes avec 10 threads

Temps moyen par pas de temps en simulation : 0.00352565 secondes avec 10 threads

Temps moyen par pas de temps à l'affichage : 0.0038697 secondes avec 20 threads

Temps moyen par pas de temps en simulation : 0.00337556 secondes avec 20 threads

Temps moyen par pas de temps à l'affichage : 0.00324938 secondes avec 30 threads

Temps moyen par pas de temps en simulation : 0.00312743 secondes avec 30 threads