Kien Do 300163370

Spécifications de mon laptop

Processor	Intel(R) Core(TM) i5-1035G7 CPU @ 1.20GHz 1.50 GHz
Installed RAM	8.00 GB (7.78 GB usable)
System type	64-bit operating system, x64-based processor
Pen and touch	Touch support with 10 touch points

Base speed	1.50 GHz
Sockets	1
Cores	4
Processors	8

Résultats - Temps d'exécution et paramètres

Paramètres requis

N = 2 and 4 consumer threads.	Execution time: 24.3288017s of 198194 points
N = 4 and 4 consumer threads.	Execution time: 9.3677512s of 206102 points
N = 4 and 10 consumer threads.	Execution time: 9.0538286s of 206102 points
N = 10 and 4 consumer threads.	Execution time: 2.4269867s of 222488 points
N = 10 and 10 consumer threads.	Execution time: 2.5953833s of 222488 points
N = 10 and 50 consumer threads.	Execution time: 2.4298802s of 222488 points
N = 20 and 10 consumer threads	Execution time: 2.127045s of 255300 points
N = 20 and 50 consumer threads.	Execution time: 2.1130431s of 255300 points
N = 20 and 200 consumer threads.	Execution time: 1.9857874s of 255300 points

Paramètres supplémentaires

N = 20 and 5000 consumer threads.	Execution time: 2.1837548s of 255300 points
N = 20 and 10000 consumer threads	Execution time: 2.0401858s of 255300 points
N = 50 and 50 consumer threads.	Execution time: 4.5412269s of 361218 points
N = 50 and 10000 consumer threads.	Execution time: 5.1496218s of 361218 points
N = 100 and 100 consumer threads.	Execution time: 15.54369s of 575776 points
N = 100 and 10000 consumer threads.	Execution time: 15.8310993s of 575776 points

Analyse et explication des résultats

On peut voir que si le nombre de partitions, N, est trop petit, le programme est très lent. La raison pour laquelle N = 2 est plus lent que N = 4 (étant donné que les deux ont le même nombre de consommateurs de 4), c'est parce que il y a plus de points GPS pour chaque partition à traiter dans une grille de 2x2 partitions qu'une grille de 4x4 partitions. En augmentant le nombre de partitions, on réduit le nombre de travaux que chaque partition doit effectuer.

Maintenant on va analyser les fils consommateurs. Dans le modèle producteur-consommateur, le producteur est comme la personne qui met du travail dans une boîte, et le consommateur est comme la personne qui livre le travail de cette boîte aux travailleurs. Dans notre programme, on a seulement 1 producteur, mais on peut avoir C consommateurs. Si on augmente le nombre de consommateurs, les travailleurs recevront leur travail plus rapidement.

Cependant, si les travailleurs travaillent encore et n'ont pas fini, le consommateur ne peut pas donner plus de travail au travailleur. Les consommateurs doivent faire la queue jusqu'à ce qu'un travailleur soit prêt à recevoir du travail. Pour cette raison, dans les exécutions supplémentaires, le temps d'exécutions de N = 100 et C = 100 est plus ou moins égale à N = 100 et C = 10000.

Avoir une valeur de N trop grande peut également ralentir le programme parce que ayant trop de partitions est comme ayant seulement 1 partition.

Supposons qu'il n'y ait pas de fils consommateurs et que les workers/partitions doivent prendre leur travail de manière procédurale (l'un après l'autre). Supposons qu'on a 100 tâches à compléter et que chaque tâche doit être effectuée l'une après l'autre. Maintenant, supposons qu'il n'y ait qu'un seul travailleur (1 partition). Ce travailleur prendrait les 100 emplois, puis essaierait de tous les terminer. On sait que c'est très lent dans ce cas-ci. Ensuite, supposons qu'on a 100 travailleurs mais que chaque travailleur ne gère qu'un seul job. Tous les 100 travailleurs devraient faire la queue, 1 travailleur prendrait 1 job, puis le travailleur suivant prendrait 1 job, etc. jusqu'à ce que les 100 travailleurs aient chacun pris un job. Vu que chaque travailleur ne prend qu'un seul job et que les job ne sont attribués qu'à 1 personne à la fois, la vitesse des deux cas sera à peu près la même.

Par conséquent, il est beaucoup plus efficace d'avoir, par exemple, 10 travailleurs, chacun occupant 10 jobs à la fois. Maintenant, la queue n'a que 10 travailleurs, et chaque travailleur n'a à gérer que 10 jobs. Il doit y avoir un équilibre entre le nombre de travailleurs (partitions) et la quantité de travail à effectuer.

Notez que c'est aussi la différence entre la concurrence et le parallélisme. En concurrence, un certain nombre de travaux sont distribués un par un, alors qu'en parallélisme, on peut avoir 2 files d'attente où le travail est distribué en même temps dans les deux files d'attente.