Projet PPAR : Parallélisation Heatsdink

Romain CAPRON, Yannick ZHANG - MAIN4

1 Parallélisation 1D

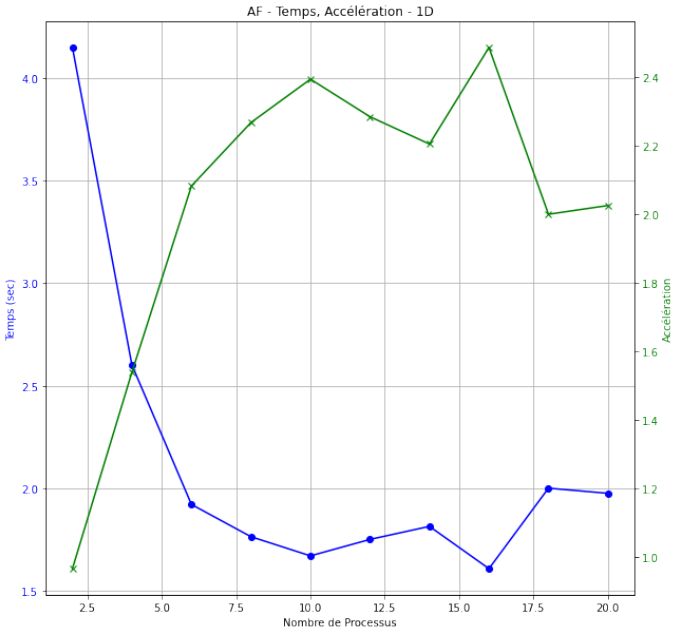
1.1 Description de la démarche

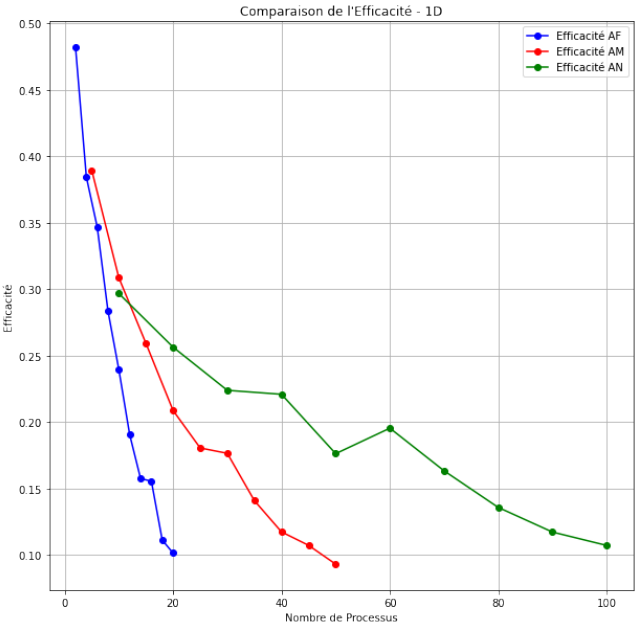
Nous avons réalisé cette première parallélisation en divisant le volume 3D (x, y, z) du dissipateur thermique en tranches le long de l'axe z pour une répartition 1D, attribuant chaque tranche à différents processus.

Puis pour gérer les tranches restantes dues à une division non entière du volume par le nombre de processus, nous avons attribué une tranche supplémentaire aux processus ayant un rang inférieur au reste de la division (rank $<$ reste). Ainsi, certains processus peuvent avoir une charge de travail légèrement plus élevée pour garantir que toutes les tranches soient traitées sans laisser de données inutilisées. Et cela assure une charge de travail équilibrée entre les processus.

En fin de simulation, nous avons rassemblé les résultats par le processus maître (rang $=$ 0) en utilisant MPI Gatherv, reconstruisant ainsi l'état thermique complet du dissipateur pour l'analyse du rendu visuel.

1.2 Mesures des performances





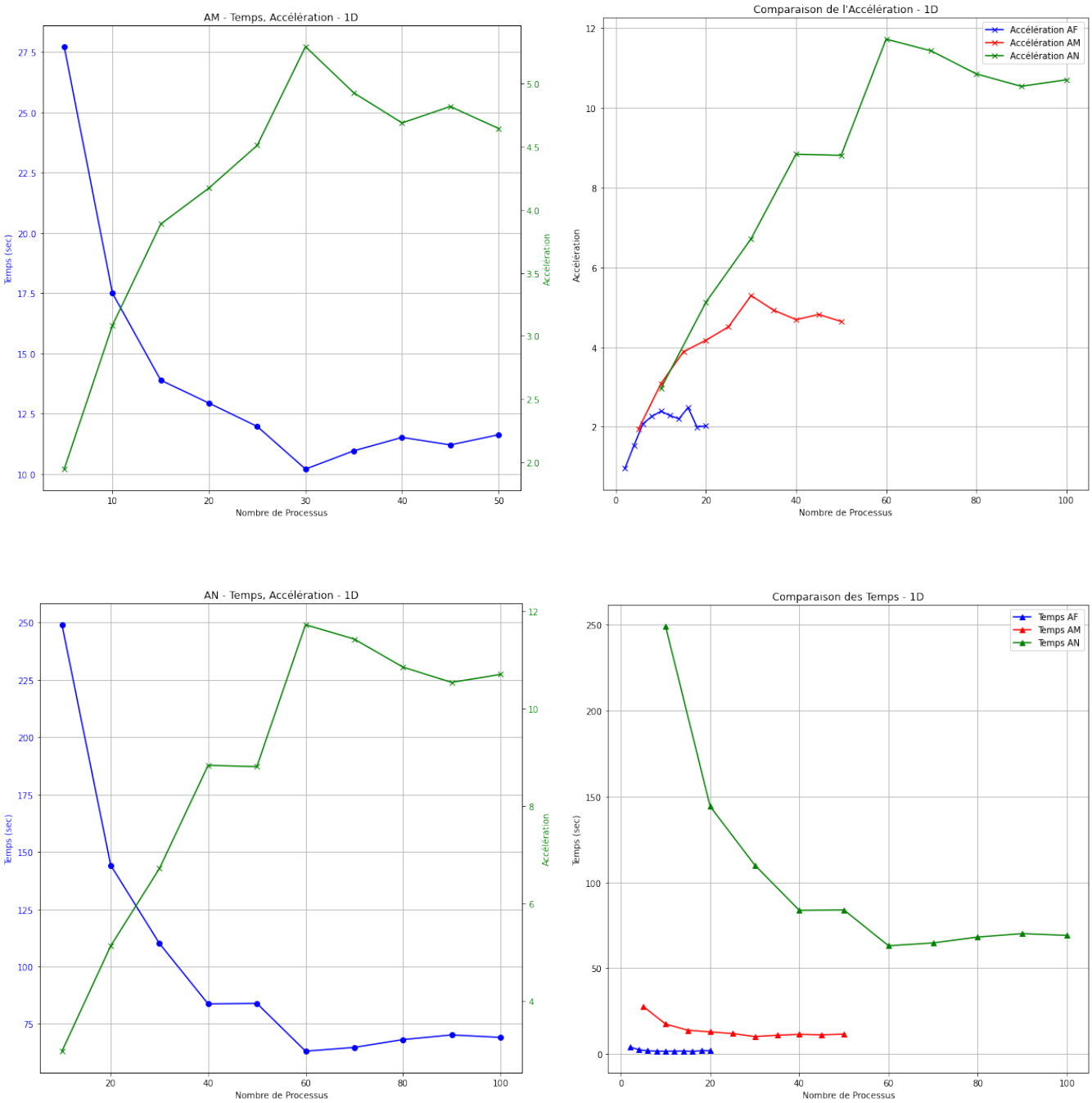


FIGURE 3 – Analyse de la parallélisation 1D

1.3 Commentaire sur les résultats

On observe des courbes de temps (et donc d'accélaration) ayant la même tendance, quelle que soit le test. En effet, l'accélération augmente, jusqu'à un certain point, puis stagne. On en déduit un nombre de processus optimaux d'environ **15, 30 et 60**, respectivement, ce qui correspond à **o/2 processus** environs. L'efficacité chute bien plus rapidement lors du test en FAST(AF), puis en MEDIUM(AM), et enfin en NORMAL(AN). Ce résultat était attendu, le temps passé à échanger des données étant plus important lors de l'utilisation de beaucoup de processus.

2 Parallélisation 2D

2.1 Description de la démarche

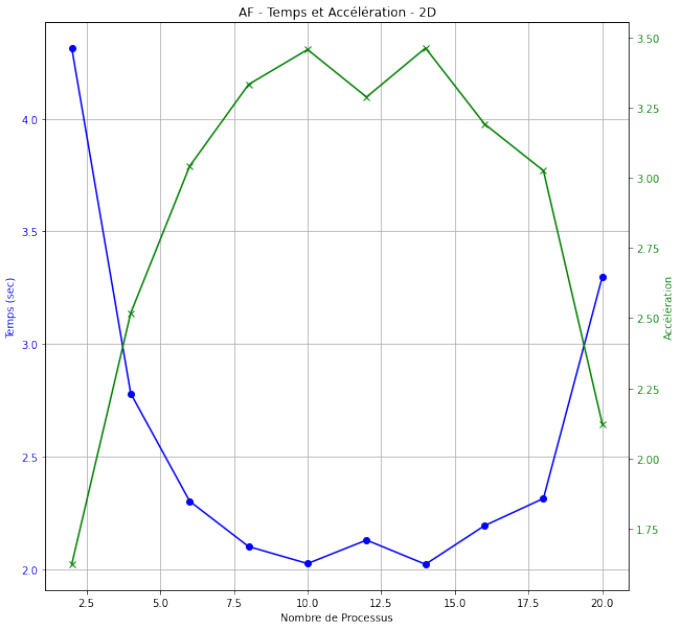
Après avoir mis en œuvre avec succès la parallélisation 1D, nous avons aussi tenté une approche de parallélisation 2D, dans le but de potentiellement améliorer les performances de notre simulation de dissipation thermique.

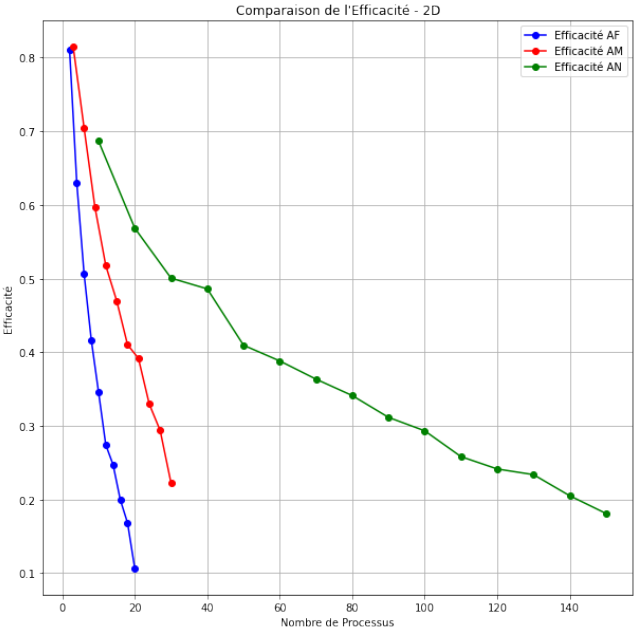
Cette nouvelle méthode divise le volume 3D du dissipateur thermique en grille 2D le long des axes x et z. Chaque processus gère une section spécifique de cette grille. Les sections sont réparties équitablement entre les processus, en commençant par les premiers pour chaque axe. Ensuite pour les restes sur les axes x ou z, chaque section supplémentaire est attribuée aux premiers processus le long de l'axe perpendiculaire. Cela garantit une répartition équitable des charges de travail supplémentaires parmi les processus.

Cependant, contrairement à la parallélisation 1D où les données étaient contiguës et ne posaient pas de problème majeur lors des communications entre les processus, la parallélisation 2D a introduit un défi supplémentaire. En effet, en découpant le volume 3D du dissipateur thermique selon les axes x et z, il était possible que des données non contiguës doivent être échangées entre les processus. Cela signifiait que les informations à envoyer ou à recevoir n'étaient pas nécessairement adjacentes dans la mémoire, ce qui nécessitait une gestion plus complexe pour garantir la cohérence des données entre les processus. Pour répondre à ce problème nous avons fait usage de MPI vector. Elle nous a permis d'envoyer une quantité déterminée de données à chaque envoi, puis de sauter une distance spécifiée pour atteindre la prochaine portion de données.

Et enfin pour recueillir les données calculées dans chaque processus, nous avons utilisé les opérations "Isend" et "Irecv" pour envoyés les données dans le processus principal. Puis une fois ces données reçues, nous les avons assemblés et placées dans la représentation globale du dissipateur thermique.

2.2 Mesures des performances





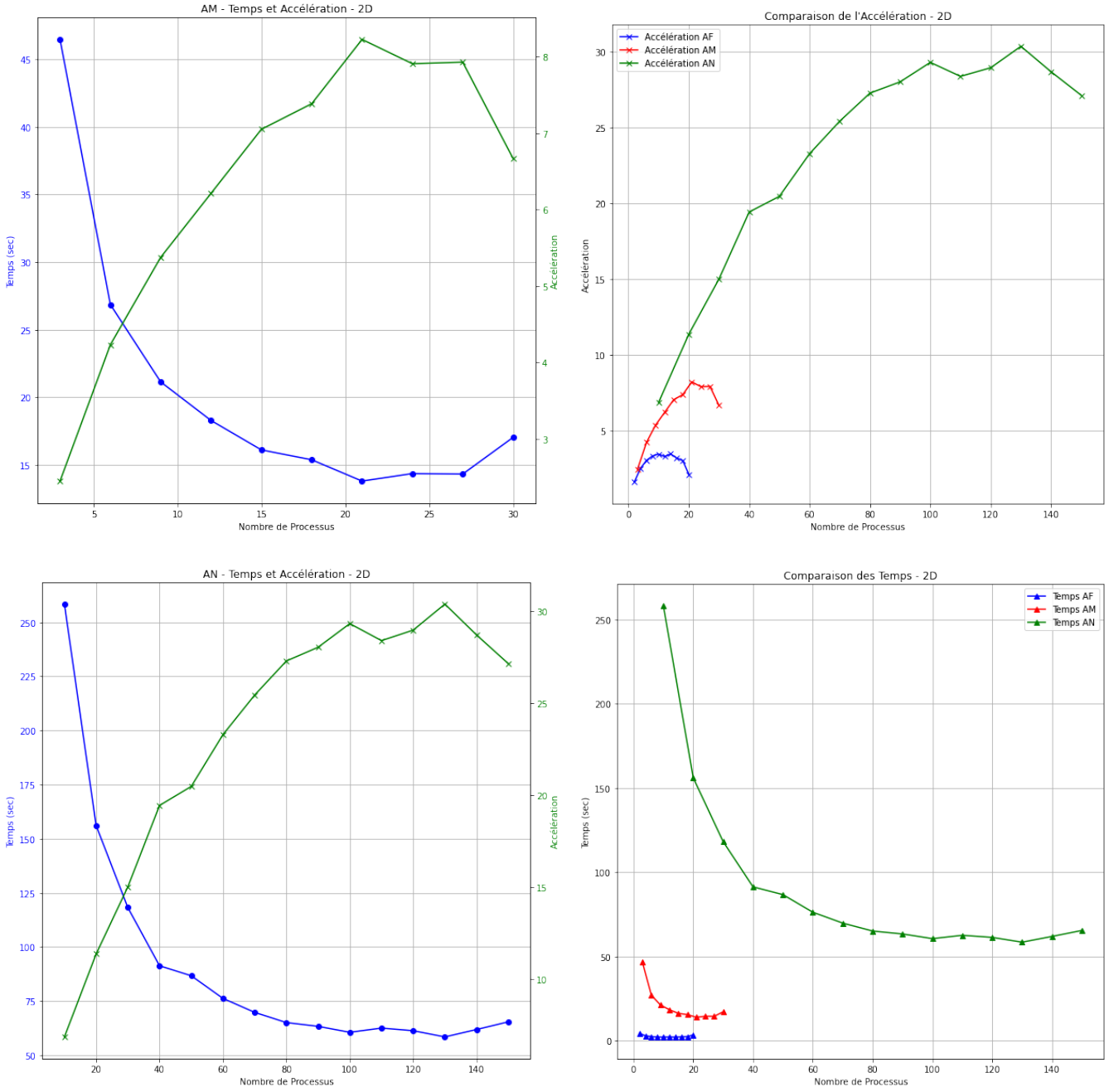


FIGURE 6 – Analyse de la parallélisation 1D

2.3 Commentaire sur les résultats

Les tests en 2D donnent des tendances similaires à ceux en 1D. Cependant, contrairement à ce à quoi on aurait pu s'attendre, la parallélisation 2D est moins efficace que la 1D. On a encore une fois un seuil atteint quelle que soit le test, à environs **2o/3 processus**. Arpès ce chiffre, l'overhead fait que les preformances diminuent. Encore une fois, l'efficacité chute bien plus rapidement pour le test FAST que pour MEDIUM et NORMAL.

3 Test Challenge avec la 1D

Test pour FAST - 15 cores
- Grid (x, y, z) = 38 x 2 x 30 (0.0Mo)
- $t = 0.0000s$; $T_{max} = 20.1C$; convergence = 125.849, time = 0.00103498 sec
- $t = 1.0000s$; $T_{max} = 27.8C$; convergence = 48.1053, time = 0.011132 sec
- $t = 255.0000s$; $T_{max} = 52.2C$; convergence = 0.098687, time = 1.84382 sec

Test pour MEDIUM - 30 cores
- Grid (x, y, z) = 75 x 4 x 60 (0.1Mo)
- $t = 0.0000s$; $T_{max} = 20.1C$; convergence = 592.331, time = 0.00135398 sec
- $t = 1.0000s$; $T_{max} = 30.3C$; convergence = 185.97, time = 0.0352111 sec
- $t = 368.0000s$; $T_{max} = 72.9C$; convergence = 0.0993016, time = 11.3524 sec

Test pour NORMAL - 60 cores
- Grid (x, y, z) = 150 x 8 x 120 (1.1Mo)
- $t = 0.0000s$; $T_{max} = 20.1C$; convergence = 2660.31, time = 0.0020268 sec
- $t = 1.0000s$; $T_{max} = 30.7C$; convergence = 523.931, time = 0.184087 sec
- $t = 470.0000s$; $T_{max} = 77.0C$; convergence = 0.0992155, time = 69.7937 sec

Test pour CHALLENGE - 600 cores
- Grid (x, y, z) = 1500 x 80 x 1200 (1098.6Mo)
- $t = 0.0000s$; $T_{max} = 20.0C$; convergence = 257038, time = 0.03909 sec
- $t = 1.0001s$; $T_{max} = 31.9C$; convergence = 17941.7, time = 1371 sec

4 Conclusion

Les résultats de nos tests de parallélisation 1D et 2D montrent des tendances similaires. Globalement, une amélioration des performances est observée avec l'augmentation du nombre de processus, atteignant un pic à un certain point, puis stagnation voire diminution.

Cependant, les performances de la 2D sont moins bonnes que celles de la 1D, alors qu'on aurait pu s'attendre au contraire. Cela est dû aux performances de communication moins importantes , ou à un équilibrage des charges moins bon. La gestion de la mémoire est également plus complexe lors de notre décomposition 2D. On préferera donc utiliser la décomposition 1D.

Pour la parallélisation 1D, nous avons identifié des nombres optimaux aux alentours de 15, 30 et 60 processus pour les tests FAST, MEDIUM et NORMAL respectivement, correspondant souvent à $(o/2)$.

Dans le contexte du test "Challenge", nous avons maintenu la tendance en choisissant $o/2$ processus, soit 600 processus.

Cependant, il est important de noter que le test "Challenge" a été effectué sur une durée plus courte (1 seconde) en raison de contraintes de temps et de ressources, limitant la capacité à observer la convergence complète du système.

En conclusion, bien que la parallélisation 1D ait démontré son efficacité dans des tests à grande échelle, le choix du nombre de processus doit être minutieusement évalué en fonction des caractéristiques spécifiques du problème, en particulier le nombre de plans sur l'axe z, et des ressources disponibles.