Support avancé AMON-GPU (septembre-octobre 2025)

Nom du code : AMON-GPU

Partenaire : Alexandre Poyé
Laboratoire de Physique des Interactions Ioniques et Moléculaires, Marseille

Personnel IDRIS : Xuezhou Lu, Juan Jose Silva Cuevas,
Isabelle Dupays

Description du code : le code AMON_GPU est une version « toy » du code MHD qui simule les plasmas de tokamak, il intègre des géométries relativement simples mais permet une très bonne analyse de la physique en se focalisant sur les instabilités qui apparaissent dans les tokamaks afin d'en connaître les causes et leurs effets.

La version dite « toy » a été mise en place par l'utilisateur avec les algorithmes de la version en production ; il s'agit d'une version simplifiée avec moins de fonctionnalités, mais avec les étapes principales de la méthode MHD. Cette version simplifiée a été parallélisée avec la bibliothèque d'échange de messages MPI sur CPU, et sur mono-GPU via des directives OpenACC. De plus, le code possède deux versions différentes appelées XYZ et YZX, dans lesquelles la manière dont les structures de données sont implémentées est modifiée.

Travail effectué et résultats obtenus :

- Validation de la version portée sous OpenACC+cuFFT développé par l'utilisateur comparée avec la version MPI+ FFTW3, les résultats sont identiques.

- Comparaison et validation des performances entre la version CPU (MPI+FFTW3) et la version GPU (OpenACC+cuFFT). La version MPI a été testée sur 1/4 de nœuds soit 10 processus MPI, la version OpenACC sur 1 GPU NVIDIA V100. Nous avons lancé les tests sur quatre résolutions et sur les deux versions du code, XYZ et YZX, fournies par l'utilisateur. L'outil Nsight Systems a été utilisé pour le profilage des performances.
  Un rapport de 3,3 est atteint sur la résolution HR pour la version XYZ entre la version OpenACC et la version MPI, et un rapport de 12 pour la version YZX.

Selon les résultats de Nsight Systems, le temps passé dans cuFFT représente plus de 73 % du temps d'exécution total. Nous avons donc trouvé particulièrement pertinente l'optimisation intégrée par l'utilisateur dans la version YZX, visant à améliorer la contiguïté des données en mémoire selon la position X dans la routine cuFFT.

- Nous avons vérifié que l'intégration des directives OpenACC est correcte. Le temps passé à copier les données reste inférieur à 1 %, ce qui est très raisonnable. Nous avons discuté de la possibilité d'exécuter cuFFT sur plusieurs GPU ou d'utiliser cuFFTMp pour obtenir une version multi-GPU. Ces bibliothèques présentent l'avantage de ne pas nécessiter d'implémentation MPI avancé. Selon la configuration matérielle, la bibliothèque peut exploiter automatiquement le transfert direct entre GPUs via NVLink pour les échanges intra-nœud ou GPUDirect RDMA pour les échanges inter-nœud, assurant ainsi des communications optimisées entre GPU. Les performances dépendent de la bande passante entre les GPU, de la puissance de calcul de chaque GPU, ainsi que du type et du nombre de FFT à effectuer. Il est à noter que l'exécution sur plusieurs GPU ne garantit pas nécessairement un temps d'exécution plus court qu'une exécution sur un seul GPU.

Les résultats des performances montrent un gain d'un facteur de 12 sur le temps d'exécution sur la version mono-GPU comparée à la version MPI. Ces résultats sont pour la version YZX et dans le cas d'une plus haute résolution. Le gain de performance atteint est sur la norme et permet de valider le travail d'optimisation fait pour le portage GPU.