Stage de Master 2/PFE :
Tolérance aux pannes dans un code de simulation pour la fusion nucléaire

Sujet Mise en œuvre et évaluation de diverses approches pour la tolérance aux pannes dans le code de simulation pour la fusion nucléaire Gysela5D

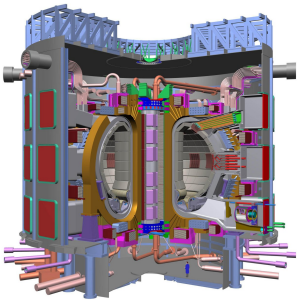
Compétences requises Bonne connaissance du langage Fortran et/ou C, maîtrise de l'environnement Unix/Linux, intérêt pour la programmation parallèle, Anglais

Durée Stage de Master 2 / Projet de Fin d'Études, 5 à 6 mois

Lieu Maison de la Simulation – CEA Saclay

Responsables Julien Bigot (julien.bigot@cea.fr – 01 69 08 01 75)
et Guillaume Latu (guillaume.latu@cea.fr)

Contexte



Coupe d'un Tokamak : réacteur pour la fusion

Un besoin majeur pour maîtriser la fusion nucléaire dans des réacteurs comme ITER est de comprendre les mécanismes qui influent sur le confinement de la chaleur au sein du plasma où a lieu la fusion. **Gysela5D**[1] (*a 5D GYrokinetic SEmi-LAgrangian code for fusion plasma simulation*) est un code utilisé pour simuler les instabilités qui se développent dans le plasma et ont un grand impact sur ce confinement. L'exécution de telles simulations nécessite des ressources de calcul particulièrement importantes qui ne peuvent être obtenues qu'en tirant parti du parallélisme massif des plus grands super-calculateurs disponibles.

Une exécution récente sur la machine Juqueen[2] a utilisé jusqu'à 458 000 cœurs de calcul en parallèle, une simulation sur la machine Jade[3] a nécessité un mois de calcul. L'utilisation d'un tel nombre de cœurs pendant des durées de cet ordre pose des problèmes liés aux pannes. En effet, même si chaque cœur est relativement fiable, la multiplication des cœurs multiplie les risques de panne. Aujourd'hui, les plus gros calculateurs subissent une panne nécessitant le remplacement de composants jusqu'à plusieurs fois par semaine. Dans ces conditions, l'exécution de grosses simulations (plusieurs jours sur plusieurs dizaines de milliers de cœurs) est quasiment assurée d'être interrompue par une panne et de ne jamais pouvoir terminer.

Une solution pour tolérer ces pannes consiste à sauvegarder l'état de la simulation sur disque régulièrement pour pouvoir la reprendre en cas de panne, on parle de checkpoint/restart. Cette sauvegarde pose des problèmes de performances liés à la grande taille des données à écrire (plusieurs téra-octets). L'augmentation de la taille des machines va rendre ce problème encore plus critique et le risque est de passer plus de temps à sauvegarder qu'à calculer. La bibliothèque **FTI**[4] propose une approche intéressante pour accélérer cette opération en tirant parti des multiples disques SSD disponibles sur certains super-calculateurs.

[1] `http://gyseladoc.gforge.inria.fr/`
[2] 5ème machine la plus puissante au monde en 2012, `http://www.top500.org/system/177722`
[3] 18ème machine en 2010, `http://www.top500.org/system/176897`
[4] `http://sourceforge.net/projects/hpc-fti/`

Description détaillée du stage

Au cours de ce stage vous serez amené à étudier les mécanismes actuellement implémenté dans Gysela5D pour la sauvegarde de son état sur disque. Gysela5D est développé en Fortran à 95% et parallélisé avec OpenMP [2] et la bibliothèque MPI [4]. Deux implémentations de la sauvegarde sur disque existent : la première, codée en Fortran, utilise des entrés-sorties synchrones ; la seconde, codée en C, utilise un thread dédié pour rendre l'écriture asynchrone [5]. Ces deux implémentations utilisent la bibliothèque d'entrés-sorties HDF5.

Vous devrez mettre en œuvre une troisième approche pour cette sauvegarde en utilisant la bibliothèque FTI développée en C. Vous ferez par ailleurs en sorte de rendre transparent l'utilisation des différentes approches en concevant une interface aussi unifiée que possible.

Votre implémentation devra être utilisable en production par les physiciens lors de l'exécutions de grande simulations. Pour cela vous devrez notamment valider que votre mise en œuvre n'introduit pas de régression en assurant que les résultats obtenus lors de l'exécution de simulations ne sont pas impactés. Pour cette phase de mise au point, vous aurez accès à un cluster de calcul comportant plus de mille cœurs.

Finalement, vous devrez analyser et modéliser le comportement de chacune des approches sur une variété de super-calculateurs incluant par exemple la machine Curie[5] comportant des SSD et donc bien adaptée à l'utilisation de la bibliothèque FTI. Vous déterminerez les critères pertinents pour le choix de l'approche à utiliser sur chaque machine disponible ainsi que sur celles à venir.

Logiciels

Gysela5D [3] est un code développé principalement au sein de l'Institut de Recherche en Fusion Magnétique (IRFM) du CEA Cadarache avec de nombreuses collaborations (Maison de la Simulation, Inria, CPT Marseille, IRMA Strasbourg, IPP Garching, LPP Paris, Montreal University, ...)

FTI [1] est une bibliothèque développée par l'équipe de Franck Cappello à l'Argonne National Laboratory (ANL) notamment suite aux travaux de thèse de Leonardo A. Bautista Gomez.

Références

[1] L. Bautista-Gomez, S. Tsuboi, D. Komatitsch, F. Cappello, N. Maruyama, and S. Matsuoka. FTI : high performance fault tolerance interface for hybrid systems. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '11, New York, NY, USA, 2011. ACM.

[2] B. Chapman, G. Jost, and R. van der Pas. *Using OpenMP : Portable Shared Memory Parallel Programming (Scientific and Engineering Computation).* The MIT Press, 2007.

[3] G. Latu, N. Crouseilles, V. Grandgirard, and E. Sonnendrücker. Gyrokinetic Semi-lagrangian Parallel Simulation Using a Hybrid OpenMP/MPI Programming. In Franck Cappello, Thomas Herault, and Jack Dongarra, editors, *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, volume 4757 of *Lecture Notes in Computer Science*, pages 356–364. Springer Berlin Heidelberg, 2007.

[4] The MPI Forum. MPI : A Message-Passing Interface Standard Version 3.0, Sep. 2012.

[5] O. Thomine, J. Bigot, V. Grandgirard, G. Latu, C. Passeron, and F. Rozar. An asynchronous writing method for restart files in the GYSELA code in prevision of Exascale systems. In *ESAIM Proceeding*, Nov 2012. To appear.

[5] `http://www.top500.org/system/177818`