# Amélioration de la parallélisation de la méthode des multipôles rapide en mémoire distribuée à l'aide d'un moteur d'exécution

Encadrants : E. Agullo, O. Coulaud
Emails : emmanuel.agullo@inria.fr, olivier.coulaud@inria.fr
Lieu : équipe HiePACS - Inria Bordeaux Sud-Ouest

## Présentation du sujet :

Les logiciels HPC ont longtemps été parallélisés avec des paradigmes « fork-join » (tels qu' « OpenMP for ») ou avec des échanges de messages (MPI). Avec l'arrivée des machines manycores composées d'unités de calcul hétérogènes (CPU et accélérateurs (GPU, MIC)) ces paradigmes ne sont pas adaptés pour utiliser pleinement et efficacement toutes les ressources. La programmation à base de tâches essaie de palier à ces limites. Ce paradigme est une façon d'abstraire la couche matérielle pour déléguer la gestion bas-niveau de la parallélisation à un moteur d'exécution qui dispatchera les tâches sur les unités de calcul disponibles sur la machine. Il existe de nombreux outils qui implémentent cette approche (StarPU, ParSEC, ...) La bibliothèque ScalFMM [1] implémentant une méthode des multipôles rapide (FMM) haute-performance a été portée sur StarPU pour exploiter efficacement dans un premier temps un nœud de calcul multicœur [2] éventuellement accéléré par des GPUs [3] et étendu à un cluster de multicœurs [4].

**Travail :**

Le code actuel a été testé jusqu'à 60 nœuds soit environ 1440 cœurs en utilisant le « local essentiel tree » (LET = octree local + Ghost). **L'objectif du stage est de passer à l'échelle sur des dizaines de milliers de cœurs**. Pour cela, il faudra améliorer la construction du LET en parallèle notamment en gérant plus finement les échanges de message. D'autre part, pour améliorer les performances, deux modifications algorithmiques seront étudiées. Dans la **première**, on dupliquera le travail en haut de l'octree sur chaque processus pour **diminuer le nombre de communications.** La **deuxième**, concerne l'introduction de deux distributions différentes des particules, une pour le champ proche et une pour le champ lointain afin d'assurer un **équilibre de charge** global pour les distributions de particules non uniformes. Pour effectuer cette deuxième tâche, dans un premier temps, un arbre entièrement dupliqué pourra être utilisé pour démontrer la faisabilité de l'approche et étudier différents couples de distributions, puis, un nouveau LET sera défini pour le couple de distributions sélectionné.

Enfin, si le temps le permet, on ré-introduira le calcul du champ proche sur GPU, pour utiliser un cluster hétérogène.

Une étude de performance à grande échelle sera effectuée pour valider les développements effectués sur la plate-forme PlaFRIM (2000 cœurs) et la machine Occigen du Cines (50544 cœurs).

**Mots-clés :** FMM, programmation en tâche, MPI, C++11, moteur d'exécution
Compétences requises : parallélisme, algorithmique, MPI, OpenMP, C++

**Commentaires :** Le stage se déroulera dans l'équipe projet HiePACS au sein du centre Inria Bordeaux – Sud-ouest.

**Références :**

[1] ScalFMM, http://scalfmm-public.gforge.inria.fr
[2] Task-based Fmm for Multicore Architectures, Emmanuel Agullo, Berenger Bramas, Olivier Coulaud, Eric Darve, Matthias Messner, Toru Takahashi, SIAM Journal on Scientific Computing. pdf
[3] Emmanuel Agullo, Berenger Bramas, Olivier Coulaud, Eric Darve, Matthias Messner, Toru Takahashi ; Task-based FMM for heterogeneous architectures. Concurrency and Computation: Practice and Experience, 2016, 28 (9),
[4] Emmanuel Agullo, Bérenger Bramas, Olivier Coulaud, Martin Khannouz, Luka Stanisic. Task-based fast multipole method for clusters of multicore processors. RR-8970, Inria Bordeaux Sud-Ouest. 2016, pp.15. pdf