Institut Farman FR 3311 : appel à projets AAP 2016

Proposition de projet Farman – Volet scientifique

NFE Miniapp : Nonlinear Finite Element Mini-application

CMLA-LMT

Intitulé du projet (acronyme ou autre) : NFEMiniapp

Titre explicite : Nonlinear Finite Element Miniapp

Version : Standard

Responsables scientifiques :

BOUCARD Pierre-Alain

pierre-alain.boucard@lmt.ens-cachan.fr

+ 33 1 47 40 53 30 / +33 6 80 61 37 77

GHIDAGLIA Jean-Michel

jmg@cmla.ens-cachan.fr

+33 1 47 40 74 29 / +336 13 79 15 12

HAFID Fikri

fikri.hafid@cmla.ens-cachan.fr

+33 6 25 97 66 41

Durée du projet : 24 mois

Membres pressentis de l'équipe-projet :

Amine MRABET, Doctorant 3ième année CMLA

Amine.mrabet@cmla.ens-cachan.fr

Résumé du projet :

Une mini-application (ci-après miniapp) est un code autonome, indépendant et de petite taille reproduisant dans les grandes lignes le comportement informatique, critique du point de vue des

performances, d'un logiciel complet. Le rôle d'une miniapp est de donner des informations de performance pour l'application dont elle est l'approximation sur de nouvelles architectures informatiques. Elles peuvent également être utilisées pour évaluer de nouveaux paradigmes de programmation.

L'objectif de ce projet est de développer une miniapp, non disponible dans la suite déjà développée, reproduisant le comportement d'un logiciel de simulation numérique par éléments finis sur un problème de mécanique non linéaire. Le problème envisagé est la plaque trouée avec un matériau dont le comportement est non linéaire.

Description scientifique du projet (3 à 8 pages)

Pendant plusieurs décennies, l'architecture des systèmes informatiques a été stable. Les améliorations (fréquence des microprocesseurs, vitesse des interconnexions, capacité mémoire) suivaient la loi de Moore sans perturbation de la structure de l'architecture et permettaient de facto un gain en performance des logiciels notamment de simulation numérique à un coût moindre : il suffisait de porter le code sur la nouvelle architecture est le gain était assuré. Depuis quelques années, la fréquence des microprocesseurs stagnant, la tendance d'évolution des architectures informatiques va vers plus de complexité (hiérarchies de mémoire complexe, des microprocesseurs plus nombreux mais moins performants,…). Cette tendance pousse les développeurs de logiciel à repenser le développement de leurs applications, à reconsidérer de nouveaux algorithmes qui, bien qu'ayant une complexité supérieure aux algorithmes optimaux, présentent une meilleure scalabilité et donc une meilleure performance lorsque l'architecture est fortement parallèle. Dans ce cadre, la portabilité d'une ancienne architecture à une nouvelle a un coût certain avec un certain nombre d'incertitudes quant à la montée en performance. Cette évolution a vu également le développement de nouveaux paradigmes de programmation (programmation parallèle, GPGPU, hybride MPI/OpenMP, hybride CPU/GPU). L'adaptation d'une application à ces paradigmes nécessite un travail complexe et profond sur le code de cette dernière.

Par ailleurs, les applications de simulation numérique présentent bien souvent deux traits caractéristiques de performance ([1]):

- Bien que le code comporte plusieurs millions de lignes, la performance globale est souvent assujettie à quelques noyaux de calcul ne représentant qu'une portion congrue du code.
- Le reste du code, bien que modélisant des physiques différentes selon les applications, possède les mêmes caractéristiques de performances.

Dans ce contexte, il est apparu nécessaire de développer des outils pour évaluer la performance des nouvelles plateformes de calculs pour des applications en amont de leur mise en production. Dans le monde du calcul scientifique, il existe plusieurs benchmarks dont l'un des plus connus est le High Performance Linpack du site Top 500. Ce benchmark évalue la performance d'une plateforme par la résolution d'un système dense n×n, Ax=b, mais il ne rend pas compte du comportement global d'une application. Le logiciel dans son ensemble peut être également utilisé pour l'évaluation non loin de la mise en production de la plateforme.

Entre ces deux extrêmes, il existe un champ pour le développement d'applications capables de reproduire le comportement d'un logiciel dans son ensemble dont le développement ou l'adaptation ne nécessite pas de grands efforts : il s'agit du champ des miniapps dont nous donnerons plus loin quelques spécifications. Le projet le plus récent à s'inscrire dans cette logique est le projet Mantevo des Sandia National Laboratories (voir [1]). L'équipe en charge du projet a développé plusieurs miniapps parmi lesquelles :

- miniFE : il s'agit de la résolution par éléments finis linéaires de l'équation de conduction de chaleur en 3D dans un parallélépipède. Elle reprend toutes les phases d'un calcul éléments finis : génération

des éléments finis, leur assemblage et leur analyse. Le système linéaire engendré est résolu par gradient conjugué sans préconditionnement. La parallélisation est réalisée par décomposition de domaine, l'algorithme utilisé étant le RCB (Recursive Coordinate Bissection). Les articles [2], [3] et [4] traitent de minFE et de son utilisation.

- miniGhost : elle implémente un schéma aux différences finis (avec différents stencils) pour un problème de diffusion de chaleur avec condition de Dirichlet sur un domaine 3D homogène. Plusieurs versions existent : monoprocesseur, parallèle (MPI uniquement), parallèle hybride (OpenMP intra-noeud/MPI inter-nœud).
- miniMD : il s'agit d'une miniapp de dynamique moléculaire qui reproduit le code LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator).
- miniXyce : il s'agit encore d'une miniapp qui reproduit un logiciel spécifique Xyce qui est un code de simulation de circuit électrique. Elle possède entre autres un noyau qui résout implicitement une équation différentielle algébrique par une méthode de Newton.

Cette même équipe a par ailleurs quelque peu formalisé la notion de « miniapp » (voir [1], [5], [6]):

- O(1k) lignes de codes
- Priorité est donnée à la reproduction des partie-clefs de la performance de l'application et non à la reproductibilité de la physique
- Distribution open source
- Objectif d'aide au co-design de la partie hardware jusqu'au développement de l'application dont elle est l'«approximation ».

Le projet que nous soumettons s'inscrit dans la droite lignée de ce dernier. Il a pour objectif de développer une miniapp qui remplira au mieux ces spécifications pour reproduire le comportement d'un logiciel de simulation numérique de mécanique des structures non linéaire.

Les codes commerciaux de simulation mécanique en non linéaire (ABAQUS, SAMCEF, ANSYS…) mais aussi les codes accessibles aux académiques librement (Code_Aster, CAST3M…) se basent tous sur un algorithme de Newton-Raphson. C'est une méthode itérative qui permet de se ramener à une suite de problèmes linéaires et qui est implémentée (avec des variantes) dans tous les codes. Cette méthode a une convergence quadratique au voisinage de la solution.

De par son déploiement dans ces logiciels, il est naturel d'implémenter cette méthode dans la miniapp visée. Afin de reproduire le comportement de ces grands codes tout en restant dans un cadre raisonnable, on limitera la miniapp au critère suivant : non-linéarité matérielle (pas de grandes déformations ou de contact), problème 2D type. Le cas considéré à ce jour est une plaque trouée élasto-plastique en traction qui permet de prendre en compte la redistribution des contraintes, dans le but de solliciter le solveur non-linéaire. Pour la résolution des problèmes non-linéaires (qui nécessitent le calcul d'un opérateur de comportement tangent), on peut soit s'orienter vers une dérivation numérique soit utiliser lorsque c'est possible une forme explicite de l'opérateur tangent.

Pour la résolution des problèmes linéaires associés aux itérations, on utilisera dans un premier temps un solveur direct, sachant que des solveurs de type Krylov sont utilisées pour des problèmes de grande taille dans les codes.

Ces différents éléments permettront de reproduire les bases d'un solveur non-linéaire en calcul de structures, et en particulier les caractéristiques principales d'un problème de mécanique des structures traités par

éléments-finis.

Originalité du projet (1/2p)

L'originalité du projet porte trois points :

- Le but visé par la miniapp
- Le domaine d'application du logiciel
- Un champ de recherche relativement libre

Le premier point concerne le but de la miniapp qui est le co-design notamment de la partie hardware. L'un des buts annoncés est donc d'influencer les constructeurs d'hardware pour les orienter vers des architectures qui épousent au mieux les contraintes de performances de certaines applications de calcul scientifique même si ce segment de marché ne représente que la part congrue de la vente de hardware. Cependant plus il y aura de miniapps particulières plus le fabricant de hardware aura une image précise de ce que son architecture pourra permettre de gagner en terme de performance. Les documents [7] et [8] comportent des benchmarks réalisés respectivement par Intel et Nvidia avec miniFE.

Le second point concerne le champ d'application du logiciel. Le but visé est ici de développer une application qui reproduit le comportement d'une application industrielle. Les problèmes de mécaniques rencontrés dans l'industrie sont souvent des problèmes non linéaires. Dans le bestiaire de miniapps que nous connaissons jusqu'à présent, aucune ne traite de problème non linéaire.

Le troisième point concerne le fait que ce champ d'investigation est relativement nouveau et qu'il est donc intéressant pour les laboratoires de l'ENS Cachan de se placer sur ces thématiques. L'une des préconisations du rapport de projet Mantevo ([1]) est de mettre à contribution pour le développement de miniapps à la fois des gens issus des développeurs et des gens issues du métier ce qui se traduit par cette collaboration CMLA/LMT.

Références

1] M. Heroux et al, «Improving performance via mini-applications,» 2009.

2] P. Lin, M. Heroux, R. F. Barett et A. B. Williams, «Assessing a mini-application as a perfromance proxy for a finite element method engineering application,» *Concurrency and Computation: Practice and Experience*, vol. 27, pp. 5374-5389, 2015.

3] M. Wu, C. Yang, T. Xiang et D. Cheng, «The research and optimization of parallel finite element algorithm based on MiniFE,» *arxiv*, 2015.

I. Karlin and al, "Exploring traditional and emerging parallel programming models using
4] a proxy app," in *Parallel & Distributed Processing (IPDPS)*, Boston, MA, 2013.

M. Heroux and al, "Summary of work for ASC L2 Milestone 4465: Characterize the role
5] of mini-applicationin predicting key performance characteristics of real applications," 2012.

M. H. Alice Koniges, *Using application proxies for co-design of future HPC computer*
6] *systems and applications*, Dagstuhl, 2012.

A. Supalov, A. Semin, M. Klemm et C. Dahnken, Optimizing HPC applications with
7] Intel cluster tools, Appress Media, 2014.

L. Justin, W. Alan et H. Mike, *Optimizing miniFE an implicit finite element application*
8] *on GPUs.*

Valeur ajoutée des différents partenaires à la réalisation du projet (1/2p)

LMT : connaissance mécanique non linéaire / logiciel de simulation numérique en mécanique non linéaire /mise à disposition d'un cluster de calcul pour l'évaluation de performance

CMLA : connaissance en schéma numérique / paradigme de programmation

Publication du projet scientifique sur site web Farman

Acceptez-vous la publication de ce projet scientifique sur le site web Farman ? Oui