Reproduction des résultats de parallélisation du code DynELA : Une étude de cas sur la pérennité des résultats de recherche

Olivier PANTALÉ, Université de Toulouse – UTTOP - LGP - Tarbes

Cette étude présente la reproduction des résultats d'un article publié en 2005 concernant la parallélisation du code de calcul par éléments finis DynELA [1]. Ce code, développé entre 1997 et 2005 au Laboratoire Génie de Production de l'ENIT, permet la simulation numérique de structures mécaniques soumises à des impacts en grandes déformations thermomécaniques. L'objectif était d'évaluer la reproductibilité des résultats et la robustesse des stratégies de parallélisation proposées, tout en identifiant les défis liés à la pérennisation des codes de recherche.

La première difficulté rencontrée a été la récupération du code source original, dispersé sur différents supports numériques obsolètes. Une version fonctionnelle a été reconstruite en fusionnant les fichiers sources de la version parallèle avec ceux de la version standard 1.0. La méthodologie a nécessité une réorganisation de l'arborescence des sources et l'utilisation de CMake pour générer les Makefiles. Le code a été compilé avec le compilateur C++ 7.4.0 sur Ubuntu 18.04.4 LTS, en utilisant les bibliothèques de parallélisation OpenMP. Cette adaptation a permis de surmonter les différences entre les environnements de développement de 2005 et actuels. Les tests ont été effectués sur un serveur Dell R730 équipé de 2 processeurs Intel Xeon E5-2650 v4 (12 cœurs chacun), configuration significativement différente de la machine Compaq Proliant 8000 SMP d'origine avec 8 processeurs Intel Xeon PIII 550/2Mb 32 bits.

Malgré ces différences matérielles importantes, les résultats numériques obtenus sont remarquablement cohérents avec ceux de 2005, avec seulement quelques écarts minimes potentiellement liés à l'accumulation d'erreurs de virgule flottante entre architectures 32 bits et 64 bits. Les quatre méthodes de parallélisation originales ont été testées, confirmant la hiérarchie d'efficacité établie en 2005 : la méthode 4, intégrant un équilibrage de charge dynamique, offre le meilleur speedup en fonction du nombre de processeurs, tandis que la méthode 1 (parallélisation naïve) montre rapidement ses limites. L'algorithme d'équilibrage de charge a également été vérifié, démontrant l'évolution du nombre d'éléments attribués à chaque processeur au cours du temps pour optimiser les performances de calcul.

Cette reproduction réussie démontre la robustesse des stratégies de parallélisation proposées dans l'article original, qui restent pertinentes malgré l'évolution significative des architectures informatiques en deux décennies. Elle souligne également l'importance cruciale d'une meilleure procédure d'archivage du code source pour garantir la pérennité et la reproductibilité des résultats de recherche, un enjeu majeur pour la communauté scientifique contemporaine. L'ensemble des résultats et méthodes a été publié en 2020 lors de la réalisation de ce travail [2].

[1] O. Pantalé. *Parallelization of an object-oriented FEM dynamics code : influence of the strategies on the Speedup.* Advances in Engineering Software, **36(6)**, 361–373, 2005. doi : 10.1016/j.advengsoft.2005.01.003.

[2] O. Pantalé. *[Rp] Parallelization of an object-oriented FEM dynamics code.* ReScience C, **6(1)**, #8, 2020. doi :10.5281/zenodo.3901241.

Contact : olivier.pantale@uttop.fr