2019-01286 - PhD Position F/M Automatic task-based parallelization and scheduling for OO languages by source-to-source transformations [S]

Type de contrat: CDI de la fonction publique
Niveau de diplôme exigé: Bac + 5 ou équivalent
Autre diplôme apprécié: M.S. in Computer Science
Fonction: Doctorant

Contexte et atouts du poste

The Inria CAMUS/Cube ICPS research team focus on automatic parallelization and optimization, profiling, modeling, and compilation. The team has increased interests for the approaches used and enhanced in the high-performance community such as the task-based method.

Mission confiée

Scientific Context

The task-based method consists in dividing an application into interdependent sections, called tasks, and providing the dependencies between them. The dependencies allow obtaining valid parallel executions, i.e. with a correct execution order of the tasks and without any race conditions. The method has gained popularity in the high-performance computing community because it allows parallelizing with an abstraction of the hardware by delegating the task distribution and the load balancing to the dynamic schedulers. See [2] for an illustrative application that relies on the method. While all aspects related to the TB parallelization are active research topics, the method is mainly used by HPC experts because parallelizing a code must be done manually. With this in mind, the position aims in solving the challenges to bridge the gap between automatic parallelization and the task-based method.

Missions

The project aims in having an automatic task-based parallelization system for C++ code-based applications using the sequential task-flow (STF) model. In this model, a sequential code (i.e. a code that is originally not designed to run in parallel) is annotated to decide what sections to transform into tasks and how these ones access the data. From this description, a runtime system can build a graph of tasks and execute them in parallel, while respecting the data dependencies.

The transformation from an STF code to a graph of tasks is done by a runtime system. The current project will focus on the transformation from a sequential code to an STF code. To do so, a source-to-source transformation of sequential C++ source code will insert tasks and data accesses. The first step will be to consider one function/method as one task and will manage the data accesses at "object" level (one object = one data handle). Then, the research work will focus on fighting the successive pitfalls that will arise and increase the performance/degree of parallelism.

Principales activités

We provide here a non-exhaustive list of possible key challenges:

- How to manage the granularity to enable/disable tasking at runtime
- How/when to split a function/loop into tasks
- How to detect patterns for commutative data accesses to increase the degree of parallelism, such as in [2], or to enable speculative execution, as done manually in [3]
- How/when to remove/delay/regroup synchronizations
- How to schedule an automatically parallelized application, including on heterogeneous architectures using our own LAHeteroprio scheduler
- How to move to automatic distributed parallelization using a common task declaration strategy, as the one used in [3]

The significant results from past and ongoing researches on automatic parallelization will give few hints and will help to parallelize computational loops, see [6,7] for example. The validation of the proposed solutions will be done on large open-source C++ projects that are currently sequential.

Technology

The source-to-source transformation system will be based on clang-LLVM. It will be developed on a modern Gitlab and proposed to the community with an open-source license. We will rely on our in-house runtime system SPETABARU to manage and execute the tasks.

References

Polyhedral parallelization of binary code, Benoit Pradelle, Alain Ketterlin, Philippe Clauss, 2012, ACM Transactions on Architecture and Code Optimization (TACO), Volume 8, Number 4

Compétences
- C++ (17)
- Knowledge of compilation would be really appreciated
- Knowledge of parallel programming/computing is an asset
- Proactive, high interest in solving problems, interest in learning clang-LLVM

Avantages
- Subsidized meals
- Partial reimbursement of public transport costs
- Leave: 7 weeks of annual leave + 10 extra days off due to RTT (statutory reduction in working hours) + possibility of exceptional leave (sick children, moving home, etc.)
- Possibility of teleworking (after 6 months of employment) and flexible organization of working hours
- Professional equipment available (videoconferencing, loan of computer equipment, etc.)
- Access to vocational training
- Social security coverage

Rémunération
Monthly salary after taxes : around 1596,05€ for 1st and 2nd year. 1678,99€ for 3rd year. (medical insurance included).

Ce poste est susceptible d'être affecté dans une zone à régime restrictif (ZRR), telle que définie dans le décret n°2011-1425 relatif à la protection du potentiel scientifique et technique de la nation (PPST). L'autorisation d'accès à une zone est délivrée par le chef d'établissement, après avis ministériel favorable, tel que défini dans l'arrêté du 03 juillet 2012, relatif à la PPST. Un avis ministériel défavorable pour un poste affecté dans une ZRR aurait pour conséquence l'annulation du recrutement.

Politique de recrutement :
Dans le cadre de sa politique diversité, tous les postes Inria sont accessibles aux personnes en situation de handicap.

Attention: Les candidatures doivent être déposées en ligne sur le site Inria. Le traitement des candidatures adressées par d'autres canaux n'est pas garanti.