Master 2 internship proposal
Compiler/runtime support for dynamic DAG optimizations

Advisors: Christophe Alias (Inria & ENS de Lyon) and Samuel Thibault (Inria & University of Bordeaux)
Contact: `Christophe.Alias@ inria.fr`

Duration: 6 months (stip-end ≈ 500 euros/month)

Place: ENS de Lyon or University of Bordeaux. Teleworking might be negociated depending on the rules of your institution.

Context

Since the early days of parallel computing, industry is pushing towards programming models, languages and compilers to help the programmer in the tedious task to parallelize a program. *Task-based programming models* [1, 5, 3] view the program as a composition of coarse-grain *tasks* to be executed in a dataflow fashion, expressed as a task graph (DAG). A runtime is generally in charge of orchestrating the computation, and mapping the tasks to processing units so load balancing and data transfers are optimized. Usually, the task partitioning is obtained by partitioning the computation into *tiles*. The selection of the *tile size* is a central problem. It should expose enough parallelism to fit the resources of the processing elements while exploiting properly the cache memory. Unfortunately, with heterogeneous systems, each Processing Element (PE) has its own requirements and exposes different optimal tile sizes. For instance, GPU kernels reach their optimal efficiency for larger tile sizes, as they need to dispatch computation on many individual units to keep the occupancy high. On the other hand, CPU cores often reach good efficiency for moderate or small tile sizes [4].

Goals

In this internship, we investigate the idea of *adapting dynamically the tile size*, depending on the PE where the computation is mapped. We focus on *hierarchical DAGs* [4], which adapt dynamically the granularity of tasks with a multi-level approach, by allowing each task to be dynamically subdivided into a finer granularity inner DAG, operating on smaller tiles. New *compiler* and *runtime* algorithms are required to support such hierarchical DAGs from any HPC kernel. The internship will focus on the *compiler support*. More specifically, the intern will:

- Analyse the runtime algorithm to rewrite a task implemented in StarPU [1].
- Investigate how a task might be expressed as a composition of sub-tasks. Some insights might be borrowed from *semantic tiling* [2]. In case several compositions are possible, metrics could be proposed to select the best one given the DAG context.
- Investigate how to *update runtime metrics* after rewriting a task (e.g. bottom-level, parallelism per level). Again, a compiler algorithm will be designed to solve that issue.

Skills expected. Notions in compilers, parallelism and experience with C++.

References

[1] Cédric Augonnet, Samuel Thibault, Raymond Namyst, and Pierre-André Wacrenier. Starpu: a unified platform for task scheduling on heterogeneous multicore architectures. *Concurrency and Computation: Practice and Experience*, 23(2):187–198, 2011.

[2] Guillame Iooss, Sanjay Rajopadhye, and Christophe Alias. Semantic tiling. In *Workshop on Leveraging Abstractions and Semantics in High-performance Computing (LASH-C'13)*, Shenzhen, China, February 2013.

[3] Josep M Perez, Rosa M Badia, and Jesus Labarta. A dependency-aware task-based programming environment for multi-core architectures. In *Cluster Computing, 2008 IEEE International Conference on*, pages 142–151. IEEE, 2008.

[4] Wei Wu, Aurelien Bouteiller, George Bosilca, Mathieu Faverge, and Jack Dongarra. Hierarchical dag scheduling for hybrid distributed systems. In *2015 IEEE International Parallel and Distributed Processing Symposium*, pages 156–165. IEEE, 2015.

[5] Asim Yarkhan, Jakub Kurzak, and Jack Dongarra. Quark users' guide. *Electrical Engineering and Computer Science, Innovative Computing Laboratory, University of Tennessee*, 2011.