Batch Scheduling for Identical Multi-Tasks Jobs on Heterogeneous Platforms

Speaker: Laurent Philippe
Co-authors: Sékou Diakité and Jean-Marc Nicod*

We consider the scheduling of one batch of identical jobs on a grid of heterogeneous computers. Each job of the batch is an application described by a Directed Acyclic Graph (DAG) of tasks without forks (intree) and executed independently from the others jobs. Each job is an instance of the same DAG and the tasks of the DAG are typed, ie. there may be two identical tasks in one job and a task of one type can only be executed on the computers where the corresponding program is installed. The execution resources are distributed and each resource can carry out a set of task types, the installed programs, so several hosts may be able to carry out the same task. The objective function of our problem is to minimize the makespan of the batch execution. An example that matches these definitions, is a workflow to process a set of data in several steps on a grid where the softwares installed on the hosts differs: each step could be a filter applied to one or more images and only some filters are available on each host.

In this context, we consider the optimization of the schedule depending on the size, from small to large sized batches, on the characteristics of the batches, etc. Different scheduling techniques could be used to schedule such applications: off-line or online, flow oriented, etc. Classical off-line scheduling techniques for DAGs are heuristics, as there is no direct optimal solution to this problem. They just take one job into account so that multiple jobs will be scheduled as if they were just one application, without benefiting from the knowledge that there are several identical jobs nor of the structure of the jobs. Moreover, if the size of the batch is large, some heuristics may become very costly. On-line techniques are usually simple but they do not either take benefit of the identical structure of the jobs. Steady state oriented scheduling techniques provides an optimal solution in this case but for an infinite number of jobs.

We compare three scheduling techniques: a simple on-line scheduler, a steady-state scheduling technique and a genetic heuristic based algorithm. The on-line algorithm just schedule the tasks as soon as they are free of their dependencies. The genetic based algorithm start from a list-scheduling solution and improve it by selecting the best individuals from the makespan of their schedule. The steady-state technique generates a periodic schedule from the result of a linear program, built with the constraints that characterize the problem.

By using an experimental evaluation based on simulation, we show that the platform architecture and the DAG structure affect the performances of the schedule differently depending on the used algorithm. Then, we propose some adaptations to these scheduling algorithms to improve their performances on the execution of the batch.

*LIFC/ INRIA GRAAL, 16 route de Gray 25000 Besançon, France, [philippe,diakite,nicod]@lifc.univ-fcomte.fr