Malleable Cube&Conquer SAT Solving

Master Thesis

Satisfiability (SAT) Solving tackles one of the most prominent NP-complete problems and is of surprisingly high practical relevance for many interesting applications (e.g. software verification, automated planning, theorem proving).

Cube&Conquer is a technique especially useful for parallelizing SAT Solving. The given problem is broken down into large amounts of sub-problems ("cubes") using expensive lookahead heuristics; then the sub-problems are randomly and evenly distributed among the processing elements and each is solved by conventional means, i.e. with fast and inexpensive heuristics.

In parallel algorithms, we call a computation malleable if the amount of processing elements working on it can be adjusted while the computation is ongoing. Malleability of parallel jobs enables a scheduler to grow and shrink jobs based on the overall system load and the jobs' demands and priorities (see above figure).

For our current research, we want to resolve many SAT instances at once in a massively parallel manner, therefore requiring malleable solving algorithms. So far, simple portfolio-based parallelization is employed for malleable SAT solving; to our knowledge, there are no notable malleable realizations of Cube&Conquer SAT solving yet. Still, we believe that such an approach could lead to higher efficiency, better speedups, and overall improved performance of the system.

Your tasks:

**Design** techniques for malleable C&C SAT Solving. In particular, outline how cube generation, cube distribution and cube solving can be adapted in an efficient manner, and consider possible clause sharing strategies. For best results, incremental SAT solving (multiple SAT solving iterations with a growing set of clauses and iteration-specific assumptions) should be considered as a use case as well.

**Implement** an interface inside mallob, an existing framework for massively parallel malleable jobs, which solves SAT instances with your C&C-based approach. Make use of existing SAT / C&C solvers and build upon them in order to reach good performance.

**Evaluate** your system's performance and parallel speedups and compare them both against non-malleable C&C SAT solvers and against our portfolio-based malleable SAT solver.

**Prerequisites:**

Good programming skills in C/C++ are required. Previous experience with distributed programming via MPI will be useful and is recommended.

**Organization:**

The thesis can be supervised in German or English and should be written in English. The contact person is Dominik Schreiber, Institute of Theoretical Informatics, Room 209.