Frontiers of Distributed SAT Solving

Praxis der Forschung 2025-2026

The **Scalable Automated Reasoning** group (KASTEL-VADS, YIG Schreiber) researches scalable methods for propositional satisfiability (SAT solving), which is a highly practical problem for a wide range of applications (verification, circuit design, security, scheduling, XAI, …). In this context, we are offering a number of potential *Praxis der Forschung* one-year research projects, by arrangement.

Regardless of the topic, the project should include a prior review of directly relevant topics from SAT literature and subsequently offers a large design space for own research, including algorithmic design, theoretical considerations, and practical engineering. As part of the hands-on implementation and experimental evaluation, we offer the opportunity to get involved in a world-leading SAT solving system and to run the resulting software on a supercomputer. Programming skills in C++ are recommended. We offer comprehensive support for the used software framework.

Contact: Dominik Schreiber - https://satres.kikit.kit.edu/schreiber - dominik.schreiber@kit.edu

1. Information Exchange Beyond Conflict Clauses

For parallel and distributed SAT solving in high-performance computing environments, e.g., on supercomputers or in clouds, a well-established paradigm is to run many sequential solvers, which search the space of possible variable assignments in different ways, and let these solvers occasionally exchange selected found *conflict clauses*. Recent results show that this technique accelerates solving unsatisfiable instances in particular, while satisfiable instances benefit comparatively little.

Analogous to conflict clauses, we would like to investigate to let the individual solvers not only exchange conflict clauses but also promising (partial) *variable assignments*, which the solvers find during their search and that already fulfill a large part of the formula. Each solver should then occasionally replace its own current variable assignment with a shared, external variable assignment and thus intensify the search within this subspace. This technique could improve performance especially for satisfiable instances. The design space includes possible quality and diversity metrics for variable assignments, space-efficient representation of assignments for distributed communication, and the basis for deciding when a solver selects an external assignment (and which one).

2. Integrating GPUs

Large-scale distributed SAT solvers only utilize CPUs as of yet. A recent trend in HPC is that an increasing amount of CPU resources is being replaced with graphical processing units (GPUs) and/or related “accelerator” hardware. While numerical applications (such as Machine Learning) profit tremendously from this modern hardware, exploiting them for combinatorial search applications (such as SAT solving) is significantly more challenging. Some recent works in SAT literature showed that GPUs can support and accelerate SAT solving by performing certain special tasks, such as formula simplification, improving learned conflict clauses, or stochastic local search. However, it is unclear to which degree these improvements will translate to distributed scales and how the achieved acceleration relates to the increase in energy consumption.

The advertised project aims to explore the exploitation of GPUs in *distributed* SAT solving systems. Using previous work on GPU-accelerated SAT solving, the main task is to integrate selected existing "GPU offloading" into the world-leading distributed SAT solving platform Mallob, investigate its impact relative to the scale of solving in terms of solving performance and energy efficiency, and propose changes and improvements to the setup(s) in order to increase scalability and/or efficiency.

For this topic in particular, prior experience with GPU programming is required.

3. Extracting Unsatisfiability Cores

A frequently used feature of SAT solvers is the ability to produce *unsatisfiability cores* (UNSAT cores). An UNSAT core, or *unsatisfiable subset*, is a subset of an unsatisfiable formula's logical constraints (clauses) that is unsatisfiable in and of itself. Non-trivial UNSAT cores $C \subset F$, or even minimal ones ("*Minimally Unsatisfiable Subsets*", MUS), are important for iterative SAT-based procedures that need to know *why* a certain formula is unsatisfiable, and react accordingly. As of yet, large-scale distributed SAT solvers in high-performance computing environments, e.g., on supercomputers or in clouds, do not support reporting UNSAT cores, which limits their use for certain applications.

The advertised project aims to explore how to generate and report non-trivial UNSAT cores in parallel and distributed SAT solving. As a point of departure, prior work on (small-scale) parallel MUS extraction should be considered and revisited. The main task is to design, analyze, and implement a practical US/MUS approach in the cutting-edge distributed SAT solver MallobSat and to evaluate it in terms of running time overhead and obtained quality. Possible avenues are the use of *incremental SAT solving* and also the analysis of produced *proofs of unsatisfiability*.