Using Hints to Speed-Up SAT

Jonathan Kalechstain,[1,2] Vadim Ryvchin,[1] and Nachum Dershowitz[2]

[1] Design Technology Solutions Group, Intel Corporation, Haifa, Israel
[2] School of Computer Science, Tel Aviv University, Tel Aviv, Israel

We propose and investigate a novel method for cutting the search space explored by the SAT solver so as to help it reach a solution faster. The idea is to add *hints*, clauses that are not necessarily "correct", in the sense that they are not necessarily implied by the original input formula. We call our hint-addition platform HSat (Hint Sat), and present two variants implemented in HaifaMUC (`https://www.dropbox.com/s/uhxeps7atrac82d/Haifa-MUC.7z`), an adaptation of MiniSat 2.2, the latter referred to it as Base.

The addition of hints H to the original formula F creates an extended formula F'. Hints can, of course, affect the satisfiability of the formula. As long as H is implied by F, the extended formula F' will be equi-satisfiable with the original F (either both are satisfiable or neither is). This means that if F is satisfiable but F' is not, then there must be a contradiction between the added hints and the original formula.

In HSat, we try to solve only the extended formula F'. In case it is satisfiable, we are done, and the solver declares that the original formula was likewise satisfiable. Otherwise, the extended formula is unsatisfiable, in which case we need to understand whether the hints are the cause of unsatisfiability, that is, whether any hint is a necessary part of the proof of the empty clause. This is accomplished by an examination of the resolution graph that is built during the run of the solver on F'. In [3], the authors presented an efficient way (their "optimization **A**") of saving a partial resolution with respect to a given subset of input clauses. We use this ability to restrict tracking so that only the effects of hints are recorded in the partial graph. When the extended formula is unsatisfiable, we check the cone of the empty clause. If it includes a hint, then the status of the original formula remains unknown and additional operations are required (like deletion of the hints). Otherwise, the original formula is unsatisfiable, and we are done. Handling of inconsistent clauses was done in several other applications, like parallel solving; our solution differs, having the ability to track the full effect of the partial resolution tree.

In case the result is unknown and the UNSAT core contains only one hint, an additional optimization can be made by using the UNSAT core of the partial resolution graph. Suppose the UNSAT core contains only hint h, then h must contradict F, and $\neg h$ is implied by F. As $\neg h$ is, in this circumstance, a set (conjunction) of unit clauses, each literal in h can be negated and added as a fact to F, which will increase the number of facts and reduce the search space to be explored. This optimization can be generalized to include all graph dominators in the partial resolution graph.

We describe two heuristics for hint generation. The first, "Avoiding Failing Branches" (AFB), is a purely deterministic hint-addition method. The main idea behind it is the same idea that drives restarts in modern SAT solvers, namely, the possibility that the solver is spending too much time on "bad branches", branches that do not contain the satisfying assignment to the problem. Our motivation is to prevent the solver from entering branches that have already been explored. In our algorithm, we describe an *explored branch* that is a subset of decision variables. We pick the most conflict-active decisions and add a hint that explicitly precludes choosing that set again. In this approach, we keep a score for each literal. The score is boosted every time a clause containing it participates in a conflict. The literals with the highest scores are added to a hint in their negated form. The hint is then added right after a restart, and the same set of active decision variables will never be chosen unless the hint is removed. This approach leads to significantly improved solver times for satisfiable instances.

A second heuristic, "Randomize Hints", draws a given number of random assignments, and tries to create a set of hints that will contradict the instance. When the solver concludes unsatisfiability, all dominators of the partial resolution graph are extracted, and all literals in all dominators are added as facts in their negated form.

This work makes several contributions. An efficient generic mechanism is introduced to add hints, the goal of which is to speed up the solver. It is based on the ability to remove clauses and all the facts derived from them. In HSAT, we use the partial resolution graph of BASE to remove the hints and their effect in case of an unsatisfiable conclusion. In [1,2,3], it was shown that using selector variables for clause removal is inferior to the use of the resolution graph. We extend the path-strengthening technique [2]. Instead of using only immediate children of the removed clauses, we use all dominators in the partial resolution graph provided in BASE. We introduce two algorithms for hint generation, one of which (AFB) improves performance for satisfiable instances by 19–30%.

References

1. Nadel, A.: Boosting minimal unsatisfiable core extraction. In: Bloem, R., Sharygina, N. (eds.) Proceedings of 10th International Conference on Formal Methods in Computer-Aided Design (FMCAD), Lugano, Switzerland. pp. 221–229. IEEE (Oct 2010), `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5770953`
2. Nadel, A., Ryvchin, V., Strichman, O.: Efficient MUS extraction with resolution. In: Formal Methods in Computer-Aided Design (FMCAD), Portland, OR. pp. 197–200. IEEE (Oct 2013), `http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6679410`
3. Ryvchin, V., Strichman, O.: Faster extraction of high-level minimal unsatisfiable cores. In: Proceedings of the 14th International Conference on Theory and Application of Satisfiability Testing (SAT), Ann Arbor, MI. pp. 174–187. No. 6695 in Lecture Notes in Computer Science, Springer, Berlin (2011)