Satisfiability: where Theory meets Practice*

Inês Lynce

INESC-ID/IST,
Technical University of Lisbon, Portugal
ines@sat.inesc-id.pt

Abstract

Propositional Satisfiability (SAT) is a keystone in the history of computer science. SAT was the first problem shown to be NP-complete in 1971 by Stephen Cook [4]. Having passed more than 40 years from then, SAT is now a lively research field where theory and practice have a natural intermixing. In this talk, we overview the use of SAT in practical domains, where SAT is thought in a broad sense, i.e. including SAT extensions such as Maximum Satisfiability (MaxSAT), Pseudo-Boolean Optimization (PBO) and Quantified Boolean Formulas (QBF).

1998 ACM Subject Classification I.2.8 Problem Solving, Control Methods, and Search

Keywords and phrases Propositional Satisfiability, SAT solvers, Applications

Digital Object Identifier 10.4230/LIPIcs.CSL.2012.12

Category Invited Talk

1 Overview

Given a Boolean formula, the SAT problem is to find an assignment to the Boolean variables such that the formula is satisfied or prove that no such assignment exists. Whereas SAT is used for solving decision problems, extensions of SAT are used for solving optimization problems. Interestingly, many implementations of SAT extensions make iterative calls to a SAT solver. This means that SAT research has a direct impact on SAT extensions. Also interesting is the fact that SAT solvers are also used by or have been an inspiration to other paradigms. Well-known examples are Satisfiability Modulo Theories (SMT) [15], Answer Set Programming (ASP) [11], model checking [3] and planning [10].

Currently, SAT is well known not only for its theoretical interest but also for the effectiveness of modern SAT solvers. SAT solvers are reliable and easy to use as the result of more than one decade accessing the status of SAT solvers in international competitions [9]. Modern SAT solvers find their roots in the seminal contributions made in the 60s with the resolution based DP procedure [6] and the backtrack search based DPLL procedure [5]. DPLL has later been enhanced with a few techniques: clause learning [17] to reduce the search space, search restarts [7] to diversify the search and lazy data structures [14] to reduce the cost of propagation, among others.

SAT and SAT extensions are clearly application driven research fields. Advances are most often motivated by real problems requiring a solution. Successful examples are haplotype inference [12] and biological networks [8] in the field of biology, as well as upgrades in software packages [1] in the field of software engineering, among many others. SAT is now used not only by SAT researchers but also by other researchers who use a SAT solver as a black box.

* This work was partially supported by national funds through FCT research project ASPEN (PTDC/EIA-CCO/110921/2009) and INESC-ID multiannual funding from the PIDDAC program funds.

Computer Science Logic 2012 (CSL'12).
Editors: Patrick Cégielski, Arnaud Durand; pp. 12–13

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

SAT solvers are not limited to providing a solution or proving that no such solution exists. Another interesting topic in SAT is to reason over formulas. The motivation is to better understand the *structure* of the formula either to improve the SAT solvers performance or to better perceive the problem being encoded into SAT. For example, we may want to eliminate redundant clauses, i.e. to get a minimal equivalent subformula (MES) [2]. In terms of variables, one may be interested in identifying the backbones [13], i.e. the value assignments that are common to all solutions, and the backdoors, i.e. a set of value assignments such that the simplified formula can be solved by a poly-time algorithm [18]. As for unsatisfiable formulas, it may be relevant to identify a minimal unsatisfiable subformula (MUS) [16].

References

1 Josep Argelich, Inês Lynce, and João P. Marques Silva. On solving boolean multilevel optimization problems. In *IJCAI*, pages 393–398, 2009.

2 Anton Belov, Mikoláš Janota, Inês Lynce, and João Marques-Silva. On computing minimal equivalent subformulas. In *CP*, 2012. To appear.

3 Armin Biere, Alessandro Cimatti, Edmund M. Clarke, and Yunshan Zhu. Symbolic model checking without BDDs. In *TACAS*, pages 193–207, 1999.

4 Stephen A. Cook. The complexity of theorem-proving procedures. In *ACM Symposium on Theory of Computing*, pages 151–158, 1971.

5 Martin Davis, George Logemann, and Donald W. Loveland. A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397, 1962.

6 Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215, 1960.

7 Carla P. Gomes, Bart Selman, and Henry A. Kautz. Boosting combinatorial search through randomization. In *AAAI*, pages 431–437, 1998.

8 João Guerra and Inês Lynce. Reasoning over biological networks using maximum satisfiability. In *CP*, 2012. To appear.

9 Matti Järvisalo, Daniel Le Berre, Olivier Roussel, and Laurent Simon. The international SAT solver competitions. *AI Magazine*, 33(1), 2012.

10 Henry A. Kautz and Bart Selman. Planning as satisfiability. In *ECAI*, pages 359–363, 1992.

11 Fangzhen Lin and Yuting Zhao. ASSAT: computing answer sets of a logic program by SAT solvers. *Artif. Intell.*, 157(1-2):115–137, 2004.

12 Inês Lynce and João Marques-Silva. Efficient haplotype inference with boolean satisfiability. In *AAAI*, pages 104–109, 2006.

13 João Marques-Silva, Mikoláš Janota, and Inês Lynce. On computing backbones of propositional theories. In *ECAI*, pages 15–20, 2010.

14 Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. In *DAC*, pages 530–535, 2001.

15 Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli. Solving SAT and SAT modulo theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL(t). *J. ACM*, 53(6):937–977, 2006.

16 João P. Marques Silva and Inês Lynce. On improving MUS extraction algorithms. In *SAT*, pages 159–173, 2011.

17 João P. Marques Silva and Karem A. Sakallah. GRASP - a new search algorithm for satisfiability. In *ICCAD*, pages 220–227, 1996.

18 Ryan Williams, Carla P. Gomes, and Bart Selman. Backdoors to typical case complexity. In *IJCAI*, pages 1173–1178, 2003.