

Satisfiability Modulo Theories Competition (SMT-COMP) 2006: Rules and Procedures

Clark Barrett
Computer Science
New York University

Leonardo de Moura
Computer Science Laboratory
SRI International

Aaron Stump
Computer Science and Engineering
Washington University in St. Louis

1 Introduction

The annual Satisfiability Modulo Theories Competition (SMT-COMP) is held to spur advances in SMT solver implementations on benchmark formulas of practical interest. Public competitions are a well-known means of stimulating advancement in software tools. For example, in automated reasoning, the CASC competition for first-order reasoning tools has seen steady improvement in the systems entered since the competition began in 1996 [3]. The SAT competition for propositional SAT solvers has also seen similar, sometimes dramatic, improvements from year to year [2]. Anecdotal evidence suggests that these competitions act as a significant catalyst for tool implementors. For more information on the history and motivation for SMT-COMP, as well as the results for 2005, please see the forthcoming journal article [1], available from the SMT-COMP website: <http://www.csl.sri.com/users/demoura/smt-comp/2005/>. Please also see the website for other information on SMT-COMP 2006. The rest of this document describes the rules and competition procedures for SMT-COMP 2006. SMT-COMP 2006 is affiliated with the Computer-Aided Verification (CAV) conference, as part of the Federated Logic Conference 2006, held in Seattle, Washington.

2 Entrants

Solver format. An entrant to SMT-COMP is an SMT solver submitted in either source code or binary format by email to the organizers. Binary format submissions must be compatible with the competition hardware and operating system, which will be x86 Linux. For solvers submitted in source code form, the organizers will make reasonable efforts to ensure that the source code is not made public. Binaries for all solvers, however, will be made public after the competition. No

accomodation can be made at this time for parties interested in participating in SMT-COMP who cannot provide, either directly or via source code, a publicly distributable x86 Linux binary.

Solvers provided in source format must include a README file explaining how to compile and install the solver. Solvers must compile using publicly available (or specially provided) compilation tools on the competition machines. Installation and execution of solvers must not require root access. The organizers will make reasonable efforts to install each system, including communication with the submitters of the system in case of difficulties. Nevertheless, the organizers reserve the right to reject an entrant if its compilation or installation process proves overly difficult.

System description. SMT-COMP entrants must come with a short (1-2 pages) description of the system. This should include a list of all authors of the system and their present institutional affiliations. It should list the Problem Divisions (see Section 4 below) in which the entrant wishes to compete. The programming language(s) and software architecture of the system should also be described, as well as the basic SMT solving approach employed (e.g., lazy integration of a Nelson-Oppen combination with SAT, translation to SAT, etc.). The system description should also include some speculation as to likely performance in SMT-COMP. System descriptions are encouraged to include a URL for a web site for the submitted tool, but this is optional. Collected system descriptions will be posted on the SMT-COMP website, and should be submitted via the EasyChair electronic submission page for SMT-COMP (linked from the SMT-COMP web page). System descriptions should also include an arbitrary positive integer, called a seed number, which will be used as part of the pseudo-random selection of benchmarks.

Other systems. Due to limitations on computational resources, the organizers reserve the right not to accept more than one version of the same solver. The organizers reserve the right to submit their own systems, or other systems of interest, to the competition.

Attendance. New for SMT-COMP 2006, there will be a public session the evening of August 21st, 2006, in which submitters will have an opportunity to give a short presentation about their SMT-COMP entrant. While highly encouraged, presentation at this session is not required. So as for 2005, submitters of an SMT-COMP entrant need not be physically present at the competition to participate or win.

Deadlines. SMT-COMP entrants must be submitted by August 8, 2006. See Section 6 below for a full timeline.

3 Execution of Solvers

Dates of competition. Solvers will be executed for SMT-COMP 2006 during the course of CAV 2006 (August 16-21). Results will be announced in a special session of CAV, on CAV's last day, as well as on the SMT-COMP web site. Depending on computational resources and the number of entrants, the competition may begin some days earlier than CAV, to allow sufficient execution time. Intermediate results will be regularly posted to the SMT-COMP website as the competition runs.

Input. Each SMT-COMP entrant, when executed, must read a single input formula presented on its standard input channel. All formulas will be given in the concrete syntax of the SMT-LIB

format, version 1.2¹. The SMT-LIB format specification is publicly available from the “Documents” section of the SMT-LIB website [4]. Solvers will be given formulas just from the Problem Divisions indicated in their system descriptions. Solvers must run without any command-line arguments or special environment variable settings.

Output. For its given input formula, each SMT-COMP entrant is expected to attempt to report on its standard output channel whether the formula is satisfiable (“sat”, without the quotation marks) or unsatisfiable (“unsat”). An entrant may also report “unknown” to indicate that it cannot determine satisfiability of the formula.

Timeouts. Each SMT-COMP solver will be executed on an unloaded competition machine for each given formula, up to a fixed time limit. The time limit is yet to be determined, but it is anticipated to be around 10 minutes. Solvers that take more CPU time than the time limit or more wall clock time than this time limit plus 30 seconds will first be sent the signal SIGXCPU and then, 3 seconds later, killed. This protocol is chosen to allow solvers a last chance to emit some output before being terminated. Solvers are allowed to spawn other processes, but these will not be sent SIGXCPU. They will be killed at approximately the same time as the first started process, in some unspecified order. Solvers which fail due to exhausting memory or crashing will be considered to have timed out.

Environment. Solvers will be started in a clean directory without any other files or subdirectories. Solvers are allowed to create and write to files and directories during the course of an execution, but they are not allowed to read such files or directories back during later executions. Any files written should be put in the directory in which the tool is started, or in a subdirectory. Solvers are not responsible for cleaning up files or subdirectories they generate. The restriction on not reading in files or directories created in previous runs will not be rigorously enforced, but entrants found to have willfully violated it will be considered to have cheated.

4 Benchmarks and Problem Divisions

The Problem Divisions for SMT-COMP 2006 are the following SMT-LIB *logics*. These logics are specified in SMT-LIB format on the SMT-LIB web page. Note that the “QF_” prefix means the division’s formulas are quantifier-free, and that QF_BVUF and AUFLIA are new divisions this year.

- QF_UF: uninterpreted functions.
- QF_RDL: real difference logic.
- QF_IDL: integer difference logic.
- QF_UFIDL: uninterpreted functions and integer difference logic.
- QF_LRA: linear real arithmetic.

¹As of February 17th, the anticipated release date for this standard is the end of March. This version is expected to differ only minorly from version 1.1, which is currently available.

- QF_LIA: linear integer arithmetic.
- QF_AUFLIA: arrays, uninterpreted functions and linear integer arithmetic.
- QF_BVUF: fixed-width bitvectors and uninterpreted functions.
- AUFLIA: arrays, uninterpreted functions, and linear integer arithmetic, where formulas may involve arbitrary (first-order) quantifiers.

Benchmark sources. Benchmark formulas for these divisions will be drawn from the existing SMT-LIB library of benchmarks, populated in 2005 as part of the previous SMT-COMP effort. Additional benchmarks will be solicited from the SMT community, Spring 2006. SMT-COMP will give preference to benchmarks that are “real-world”, in the sense of coming from or having some intended application outside SMT.

Benchmark availability. The pool of benchmarks from which the competition benchmarks will be drawn will be made available in a first version on June 1, 2006. A second version incorporating any repairs to problems detected by the SMT community will be made available July 1, 2006. A final version will then be posted August 1, 2006. Each benchmark will be initially marked by the organizers as either satisfiable or unsatisfiable, according to their best information. The feedback process and iteration through multiple versions will allow the community to catch any (*a priori* conceivable) errors in classification.

Benchmark rating. Benchmarks will be assigned a difficulty rating by running the solvers entered in SMT-COMP 2005 on them. Benchmarks solvable in shorter time by more solvers will be assigned a lower difficulty rating. Benchmarks solvable only in longer time by fewer solvers will be assigned a higher rating.

Benchmark selection. Benchmarks will be chosen from the pool of all benchmarks collected according to the following scheme. Benchmarks from each division will be initially divided by the organizers into four groups, according as they are satisfiable or unsatisfiable, easier or harder (as determined by the difficulty rating). Large families of very similar benchmarks will be handled by selecting from a smaller representative subset. For SMT-COMP 2005, it proved difficult to find satisfiable benchmarks, although the organizers will make every effort to have a balance for SMT-COMP 2006. Nevertheless, an even split of each division may not be possible. To select N benchmarks for the division, where N is determined by the number of entrants and the available computational resources, the first $N/4$ pseudo-randomly selected benchmarks from each group will be chosen. Pseudo-random numbers will be generated using the standard C library function `random()`, seeded (using `srandom()`) with the sum, modulo 2^{30} , of the seed numbers provided (as part of their system descriptions; see Section 2 above) by all SMT-COMP entrants other than the organizers. A benchmark is selected from a group by a certain pseudo-random number if that number, modulo the number of benchmarks in the group, is equal to the position of the benchmark’s (uniquely identifying) name in the list of benchmarks for the group, sorted in increasing alphabetical order (and starting with position 0). Code implementing this selection mechanism will be made available in advance of the competition.

Benchmark scrambling. Simple scrambling operations which do not grossly change the structure of the benchmarks will be performed for the competition. Examples include renaming of

Reported	Correct?	Point/penalty
unsat	yes	+1
unsat	no	-8
sat	yes	+1
sat	no	-8
unknown	n.a.	0
<i>timeout</i>	n.a.	0

Figure 1: Points and Penalties

uninterpreted symbols and exchanging the orders of arguments to commutative operations. The scrambler will work in a pseudo-random fashion, seeded with the number computed from competitors' seed numbers. The scrambler will be made available along with the first release of the benchmarks, so that competitors can ensure their tools correctly handle the output of the scrambler.

5 Judging and Scoring

Winners in each Problem Division for which there are at least three entrants from distinct research groups competing will be taken to be those with the highest score, according to the system of points and penalties in Figure 1. As indicated, correct answers are awarded a positive number of points, while incorrect answers are penalized by assigning a negative number of points. Timeouts and reports of unknown are awarded zero points; as mentioned above, failures like exhausting memory or crashing are also treated as timeouts. Unlike for 2005, SMT-COMP 2006 will allow at most three wrong answers per division. Four wrong answers within a single division will cause the tool to be disqualified from all divisions. The increased strictness of this penalty system is partly offset by the timeline for benchmark availability: benchmarks will be available much earlier for SMT-COMP 2006 than they were for SMT-COMP 2005 (indeed, the benchmarks collected for SMT-COMP 2005 are available now). Hence, implementors have longer lead time to test their tools. In the event of a tie in total number of points, the solver with the lower average CPU time on formulas for which it did not timeout or report unknown will be considered the winner. For Problem Divisions with fewer than three entrants, the results will be reported but no winner officially declared.

6 Timeline (2006)

June 1 First version of the benchmark library posted for comment. Pseudo-random benchmark scrambler available.

July 1 Revised version of the benchmark library posted. Pseudo-random benchmark selector becomes available.

August 1 Final version of the benchmark library posted, and system submission opened.

August 8 Final system descriptions due, with magic numbers for pseudo-random selection of benchmarks.

August 9 Selected benchmarks posted.

August 16 SMT-COMP regular competition begins, coinciding with the start of CAV.

7 Acknowledgements

The organizers are grateful to the SMT community for feedback on these rules in several fora, including an open discussion at PDPAR 2005, for which thanks are due to the PDPAR 2005 organizers Alessandro Armando and Alessandro Cimatti; as well as email discussions. For their feedback in response to an electronic request for comment on these rules, thanks go to the following people: Madan Musuvathi, Albert Oliveras, Kenneth Roe, Hossein Sheini, and Cesare Tinelli.

References

- [1] C. Barrett, L. de Moura, and A. Stump. Design and Results of the 1st Satisfiability Modulo Theories Competition (SMT-COMP 2005). to appear in the Journal of Automated Reasoning.
- [2] D. Le Berre and L. Simon. The essentials of the SAT 2003 competition. In *Sixth International Conference on Theory and Applications of Satisfiability Testing*, volume 2919 of *LNCS*, pages 452–467. Springer-Verlag, 2003.
- [3] F.J. Pelletier, G. Sutcliffe, and C.B. Suttner. The Development of CASC. *AI Communications*, 15(2-3):79–90, 2002.
- [4] Silvio Ranise and Cesare Tinelli. The SMT-LIB web site, 2004. <http://combination.cs.uiowa.edu/smtlib>.