Selected Methods of Model Checking using SAT and SMT-solvers *

(Doctoral Consortium)

Agnieszka M. Zbrzezny
Supervisor: Bożena Woźna-Szcześniak
IMCS, Jan Długosz University. Al. Armii Krajowej 13/15, 42-200 Częstochowa, Poland.
agnieszka.zbrzezny@ajd.czest.pl

ABSTRACT

The objectives of this research are to further investigate the foundations for novel SMT and SAT-based bounded model checking (BMC) algorithms for real-time and multi-agent systems. A major part of the research will involve the development of SMT-based BMC methods for standard Kripke structures, extended Kripke structures, and for different kinds of interpreted systems for different kinds of temporal languages, each of which will be augmented to include the standard epistemic and deontic operators. The algorithms will be implemented into several modules of the model checker VerICS (`http://verics.ipipan.waw.pl/`).

Categories and Subject Descriptors

D.2.4 [**Software/Program Verification**]: Model checking

Keywords

Bounded model checking, SMT, SAT, Temporal Logic, PhD thesis extended abstract

1. INTRODUCTION

Model checking [2] provides algorithmic means of determining whether an abstract model M - representing, for example, a hardware or software project - satisfies a formal specification expressed as a temporal logic formula F. Moreover, if the property does not hold, the method identifies a counterexample execution that shows the source of the problem.

Model checking problem has been proposed independently by Quielle and Sifakis [7], and by Clarke and Emerson [5] as a method for automatic and algorithmic verification of finite state concurrent systems, and impressive strides have been made on this problem over the past thirty years. Model checking of real-time [1] and multi-agent systems [6] is a very active field for both theoretical research and practical applications.

The practical applicability of model checking in real-time, and multi-agent settings has required the development of sophisticated means of coping with what is known as the state explosion problem. It means that the number of model states grows exponentially in the size of the system representation. To avoid this problem a number of state reduction techniques and symbolic model checking approaches have been developed.

Automated verification of real-time systems (RTS) and multi-agent systems (MAS), performed by the analysis of their models is a very important subject of research. This is highly motivated by an increasing demand to verify safety critical systems, i.e., time-dependent systems, failure of which could cause dramatic consequences for both people and hardware. MAS are open, distributed software systems, in which the individual processes, or agents are rational and autonomous entities engaged in social activities such as communication, coordination, negotiation, cooperation, etc. These systems include robotic surgery machines, nuclear reactor control systems, railway signalling, breaking systems, air traffic control systems, flight planning systems, rocket range launch safety systems, and many others. Humans already benefit a lot from a variety of RTS and MAS, being often unaware of this. Model checking of RTS and MAS is known to be a difficult problem and its practical applicability is strongly limited by the state explosion problem. There is still a lack of efficient methods for RTS and MAS. In view of this, there is an obvious need to develop efficient SMT/SAT-based verification methods which could be used in practice.

2. SAT AND SMT

SAT-based bounded model checking (BMC) [3] uses a reduction of the problem of truth of a modal formula in a model (transition system) to the problem of satisfiability of formulae of the classical propositional calculus, i.e. SAT-problem. The reduction is achieved by a translation of the transition relation and a translation of a given property to formulae of classical propositional calculus. It should be emphasised that for a given temporal logic, bounded model checking is mainly used to disprove safety properties and to prove liveness properties.

The SMT problem [4] is a generalisation of the SAT problem, where Boolean variables are replaced by predicates from various background theories, such as linear, real, and integer

*The study is co-funded by the European Union, European Social Fund. Project PO KL "Information technologies: Research and their interdisciplinary applications", Agreement UDA-POKL.04.01.01-00-051/10-00.

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

arithmetic. SMT generalises SAT by adding equality reasoning, arithmetic, fixed-size bit-vectors, arrays, quantifiers, and other useful first-order theories. Using SMT to express different problems has important advantages over SAT. If one uses SAT, then, for example, data must be encoded into a Boolean representation: a bit-vector must be represented as just its individual bits, for example. In contrast, an SMT encoding can represent the bit-vector directly, and may be able to reason more efficiently at the bit-vector level of abstraction, without resorting to bit-level reasoning (though this may sometimes be necessary).

3. BMC ALGORITHM

For a given modal logic ML the application of the BMC method requires proving the theorem which provides the basis for the verification of formulae of this logic in a given model using finite prefixes of paths. A finite prefix of the length $k \geqslant 0$ is called a k-paths. The aforementioned theorem says that a formula φ of a modal logic ML is true in a transition system $\mathcal{M}$ if and only if there exists a natural number k, such that the propositional formula (for SAT-based BMC) or the quantifier-free first-order formula (for SMT-BMC) $[\mathcal{M}, \varphi]_k$ being a conjunction of a formula encoding a finite set of k-paths of cardinality $f_k(\varphi)$ and a formula being a translation of the formula φ, is satisfiable. The function f_k, whose form depends on the particular modal logic, sets a minimum number of k-paths sufficient, regardless of the transition system, for the verification of the formula φ.

The aforementioned theorem justifies the correctness of the standard BMC algorithm. Starting with $k = 0$, the algorithm creates, for a given transition system $\mathcal{M}$ and a given formula φ, a propositional formula $[\mathcal{M}, \varphi]_k$. Then the formula $[\mathcal{M}, \varphi]_k$ is converted to a satisfiability-equivalent propositional formula in conjunctive normal form and forwarded to a SAT-solver or it is converted to a quantifier-free first-order formula and forwarded to a SMT-solver. If the tested formula is unsatisfiable, then k is increased (usually by 1) and the process is repeated.

The BMC algorithm terminates if either the formula $[\mathcal{M}, \varphi]_k$ turns out to be satisfiable for some k, or k becomes greater than a certain, $\mathcal{M}$-dependent threshold (e.g. the number of states of $\mathcal{M}$). Exceeding this threshold means that the formula φ is not true in the transition system $\mathcal{M}$. On the other hand, satisfiability of $[\mathcal{M}, \varphi]_k$, for some k means that the formula φ is true in the transition system $\mathcal{M}$. Moreover, the valuation found by the SAT-solver or SMT-solver allows to determine a set of k-paths, which is a witness for φ

Note that the BMC algorithm also terminates when, for some k, the available resources (memory or time) are either insufficient to generate the formula $[\mathcal{M}, \varphi]_k$ or are insufficient for the SATor SMT-solver. In such a case, it means that the BMC algorithm is not able to check whether the property expressed by the formula φ holds in the transition system $\mathcal{M}$ due to limited resources available.

4. METHODOLOGY

The main aim is to compare the existing SAT-based bounded model checking algorithms for standard Kripke structures, extended Kripke structures, and weighted interpreted systems with our new SMT-based bounded model checking techniques for the same models.

Implementations The implementations are written in C++ programming language.

Scenarios and benchmarks In the area of formal verification it is customary to use scalable benchmarks in order to demonstrate the effectiveness of verification tools. These benchmarks are usually theoretical devices, used to compare the efficiency of used algorithms. There exists a number of such standard, typical benchmarks, e.g. several types of Mutual Exclusion Protocol, Bit Transmission Problem, General Pipeline Paradigm, etc. On the other hand, restricting only to a few standard benchmarks makes it difficult to predict the performance on real-life systems, which prompts for use of the models originating from the practical applications. We plan to use both the types of benchmarks.

5. CONCLUSIONS

We proposed, implemented, and experimentally evaluated some SAT-based and SMT-based bounded model checking approach i.e. for WECTLK interpreted over the weighted interpreted systems, for RTECTL properties interpreted over the simply-time systems and for ECTL* properties. We have compared our SMT-based method with the corresponding SAT-based method. The experimental results show that the approaches are complementary. Also an observation of experimental results leads to the conclusion that the SAT-based BMC for uses less memory comparing to the SMT-based BMC. This is a novel and interesting result, which shows that the choice of the BMC method should depend on the considered system.

REFERENCES

[1] R. Alur, C. Courcoubetis, and D. Dill. Model checking for real-time systems. In *Proceedings of the 5th Symp. on Logic in Computer Science (LICS'90)*, pages 414–425. IEEE Computer Society, 1990.

[2] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[3] A. Biere, A. Cimatti, E. Clarke, M.Fujita, and Y. Zhu. Symbolic model checking using SAT procedures instead of BDDs. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC'99)*, pages 317–320, 1999.

[4] A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2009.

[5] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite state concurrent systems using temporal logic specifications: A practical approach. In *Conference Record of the Tenth Annual ACM Symposium on Principles of Programming Languages, Austin, Texas, USA, January 1983*, pages 117–126. ACM Press, 1983.

[6] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning About Knowledge*. MIT Press, 1995.

[7] J. P. Quielle and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *Proceedings of the 5th International Symp. on Programming*, volume 131 of *LNCS*, pages 337–351. Springer-Verlag, 1981.