NuSMV 2: An OpenSource Tool for Symbolic Model Checking

Alessandro Cimatti[1], Edmund Clarke[2], Enrico Giunchiglia[3], Fausto Giunchiglia[4], Marco Pistore[1], Marco Roveri[1], Roberto Sebastiani[4], and Armando Tacchella[3]

[1] ITC-IRST, Via Sommarive 18, 38050 Trento, Italy
{cimatti,pistore,roveri}@irst.itc.it
[2] Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh (PA), USA
emc@cs.cmu.edu
[3] DIST – Università di Genova, Viale Causa 13, 16145 Genova, Italy
{enrico,tac}@mrg.dist.unige.it
[4] Università di Trento, Via Sommarive 14, 38050 Trento, Italy
{fausto,rseba}@science.unitn.it

1 Introduction

This paper describes version 2 of the NuSMV tool. NuSMV is a symbolic model checker originated from the reengineering, reimplementation and extension of SMV, the original BDD-based model checker developed at CMU [15]. The NuSMV project aims at the development of a state-of-the-art symbolic model checker, designed to be applicable in technology transfer projects: it is a well structured, open, flexible and documented platform for model checking, and is robust and close to industrial systems standards [6].

The first version of NuSMV, referred with NuSMV1 in the following, basically implements BDD-based symbolic model checking. The new version of NuSMV (NuSMV2 in the following) inherits all the functionalities of the previous version, and extend them in several directions. The main novelty in NuSMV2 is the integration of model checking techniques based on propositional satisfiability (SAT) [4]. SAT-based model checking is currently enjoying a substantial success in several industrial fields (see, e.g., [10], but also [5]), and opens up new research directions. BDD-based and SAT-based model checking are often able to solve different classes of problems, and can therefore be seen as complementary techniques.

Starting from NuSMV2, we are also adopting a new development and license model. NuSMV2 is distributed with an OpenSource license [17], that allows anyone interested to freely use the tool and to participate in its development. The aim of the NuSMV OpenSource project is to provide to the model checking community a common platform for the research, the implementation, and the comparison of new symbolic model checking techniques. NuSMV2 has been released in November 2001. Since then, the NuSMV team has received code contributions for different parts of the system. Several research institutes and commercial companies have express interest in collaborating to the development of NuSMV.

In this paper we describe the goals of the NuSMV OpenSource project (Section 2), we give an overview of the system (Section 3), and we end with some concluding remarks (Section 4). Further information on NuSMV can be found at http://nusmv.irst.itc.it/.

2 The NuSMV Open Source project

Enormous progress has been carried out over the last decade in the applicability of symbolic model checking to practical verification problems. However, most of the state-of-the-art model checkers are proprietary. Moreover, several important techniques have been implemented only in prototype tools and have not been further maintained or developed (see e.g., the very nice results described in [8]). This is a clear disadvantage in terms of scientific progress, and is slowing down the introduction of model checking in non-traditional application domains.

With the OpenSource model [17], a whole community participates in the development of a software systems, with a distributed team and independent peer review. This may result in a rapid systems evolution, and in increased software quality and reliability. The OpenSource model has boosted the take-up of notable software systems, such as Linux and Apache. With the NuSMV OpenSource project, we would like to reach the same goals within the model checking community, providing a publicly available state-of-the-art symbolic model checker, and opening to anybody interested in the development of the tool.

NuSMV2 is distributed under GNU Lesser General Public License 2.1 (LGPL in brief; see [14]). This license grants full right to use and modify a program, for research and commercial applications, stand-alone or as part of a larger software system. On the other hand, this license is "copyleft": any improvement to NuSMV should be made freely available, under the terms of the LGPL. In this way, we achieve the goals of allowing for a free usage of NuSMV and to guarantee that the extensions become available to the whole community.

3 Overview of NuSMV2

In order to integrate SAT-based and BDD-based model checking, a major architectural redesign was carried out in NuSMV2, with the goal of making as many functionalities as possible independent of the actual model checking engine used. This allows for the effective integration of the new SAT-based engine, and opens up towards the implementation of other model checking procedures. A high level description of the internal structure of NuSMV2 is given in Figure 1.

NuSMV is able to process files written in an extension of the SMV language. In this language, it is possible to describe finite state machines by means of declaration and instantiation mechanisms for modules and processes, corresponding to synchronous and asynchronous composition, and to express a set of requirements in CTL and LTL. NuSMV can work batch or interactively, with a textual interaction shell.

An SMV file is processed in several phases. The first phases require to analyze the input file, in order to construct an internal representation of the system. NuSMV2 neatly

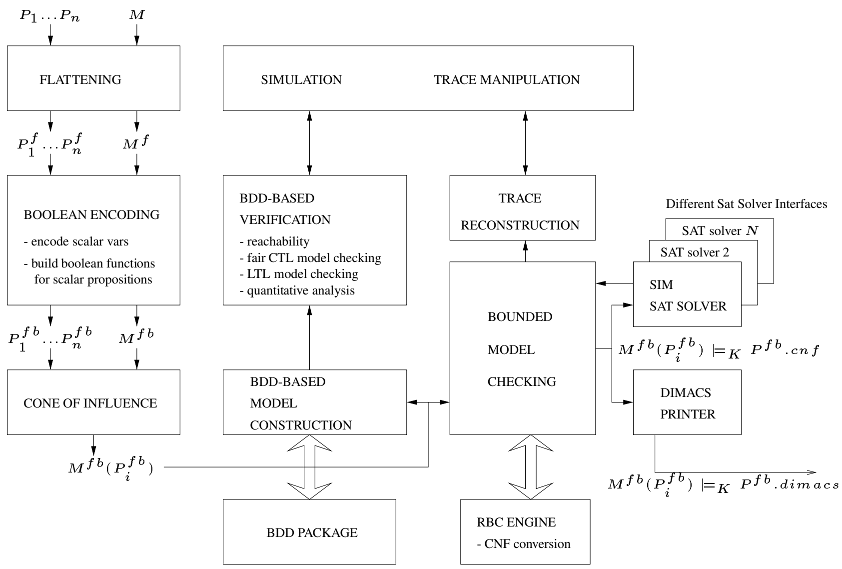

Fig. 1. The internal structure of NuSMV2.

separates the input language in different layers, of increasing complexity, that are incrementally eliminated. The construction starts from the modular description of a model M and of a set of properties $P_1, \dots, P_n$. The first step, called *flattening*, performs the instantiation of module types, thus creating modules and processes, and produces a synchronous, flat model M^f, where each variable is given an absolute name. The second step, called *boolean encoding*, maps a flat model into a boolean model M^{fb}, thus eliminating scalar variables. This second step takes into account the whole SMV language, including the encoding of bounded integers, and all the set-theoretic and arithmetic functions and predicates. It is possible to print out the different levels of the input file, thus using NuSMV2 as a flattener. The same reduction steps are applied to the properties P_i, thus obtaining the corresponding flattened boolean versions P_i^{fb}. In addition, by means of the *cone of influence* reduction [2], it is possible to restrict the analysis of each property to the relevant parts of the model $M^{fb}(P_i^{fb})$. This reduction can be extremely effective in tackling the state explosion problem.

The preprocessing is carried out independently from the model checking engine to be used for verification. After this, the user can choose whether to apply BDD-based or SAT-based model checking. In the case of BDD-based model checking, a BDD-based representation of the Finite State Machine (FSM) is constructed. In this step, different partitioning methods and strategies [18] can be used. Then, different forms of *BDD-based verification* can be applied: reachability analysis, fair CTL model checking, LTL model checking via reduction to CTL model checking, computation of quantitative characteristics of the model.

In the case of SAT-based model checking, NuSMV2 constructs an internal representation of the model based on a simplified version of *Reduced Boolean Circuit* (RBC), a representation mechanism for propositional formulae. Then, it is possible to perform SAT-based *bounded model checking* of LTL formulae [4]. Given a bound on the length of the counterexample, a LTL model checking problem is encoded into a SAT problem. If a propositional model is found, it corresponds to a counterexample of the original model checking problem. NuSMV2 represents each SAT problem as an RBC, that is then converted in CNF format and given in input to the internal SAT solver. Alternatively, the SAT problems can be printed out in the standard *DIMACS* format, thus allowing for the stand-alone use of other SAT solvers. With respect to the tableau construction in [4], enhancements have been carried out that can significantly improve the performances of the SAT solver [7]. In bounded model checking, NuSMV2 enters a loop, interleaving problem generation and solution attempt via a call to the SAT solver, and iterates until a solution is found or the specified maximum bound is reached.

NuSMV2 uses SIM [13] as the internal SAT solver. SIM is a SAT solver based on the Davis-Logemann-Loveland procedure. The features provided by SIM can produce dramatic speed-ups in the overall performances of the SAT checker, and thus of the whole system (see e.g., [19, 10] for a discussion). It is currently under development a generic interface to SAT solvers to allow for the use of new state of the art SAT solvers like e.g. CHAFF [16].

The different properties that are checked on a FSM are handled and shown to the user by a property manager, that is independent of the model checking engine used for the verification. This means that it is possible for the user to decide what solution method to adopt for each property. Furthermore, the counterexample *traces* being generated by both model checking modules are presented and stored into a unique format. Similarly, the user can *simulate* the behavior of the specified system, by generating traces either interactively or randomly. Simulation can be carried out both via BDD-based or SAT-based techniques.

4 Concluding remarks

NuSMV is a robust, well structured and flexible platform for symbolic model checking, designed to be applicable in technology transfer projects. In this paper, we have shown how BDD-based and SAT-based model checking are integrated in the new version of NuSMV, that significantly extends the previous version. In particular, we have discussed the functionalities and the architecture of NuSMV2, that integrates SAT-based state of the art verification techniques, is able to work as a problem flattener in DIMACS format, and tackles the state explosion with cone of influence reduction. In the future, we plan to investigate a tighter integration between BDD-based and SAT-based technologies. The new internal architecture also opens up the possibility to integrate different boolean encodings (e.g. [8]) and verification engines (e.g. [1], allowing for Bounded Model Checking of Timed Automata.)

NuSMV2 has been used as the starting framework for the implementation and the evaluation of new verification techniques (see, e.g., [9] in this volume). It has also been used as the verification engine for tools in different application areas, ranging from the formal validation of software requirements [12], to the verification of StateChart models [11], to automated task planning [3]. Several of these applications have required the development of new functionalities and improvements to NuSMV2. The code for these extensions is currently being included in the mainstream NuSMV2 distribution.

References

1. G. Audemard, P. Bertoli, A. Cimatti, A. Kornilowicz, and R. Sebastiani. A SAT based approach for solving formulas over boolean and linear mathematical propositions. In *Proc. of CADE'02*, 2002.
2. S. Berezin, S. Campos, and E. M. Clarke. Compositional reasoning in model checking. In *Proc. COMPOS*, 1997.
3. P. Bertoli, A. Cimatti, M. Pistore, M. Roveri, and P. Traverso. MBP: a Model Based Planner. In *Proc. of the IJCAI'01 Workshop on Planning under Uncertainty and Incomplete Information*, Seattle, August 2001.
4. A. Biere, A. Cimatti, E. .M. Clarke, and Y. Zhu. Symbolic model checking without BDDs. In *Proc. of the Fifth International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99)*, 1999.
5. A. Borälv. A Fully Automated Approach for Proving Safety Properties in Interlocking Software Using Automatic Theorem-Proving. In S. Gnesi and D. Latella, editors, *Proc. of the Second International ERCIM FMICS*, Pisa, Italy, July 1997.
6. A. Cimatti, E. M. Clarke, F. Giunchiglia, and M. Roveri. NUSMV : a new symbolic model checker. *International Journal on Software Tools for Technology Transfer (STTT)*, 2(4), March 2000.
7. A. Cimatti, M. Pistore, M. Roveri, and R. Sebastiani. Improving the Encoding of LTL Model Checking into SAT. In *Proc. WMCAI 2002*, number 2294 in LNCS, pages 182–195, 2002.
8. E. Clarke and X. Zhao. Word Level Symbolic Model Checking: A New Approach for Verifying Arithmetic Circuits. Technical Report CMU-CS-95-161, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213-3891, USA, May 1995.
9. E. M. Clarke, A. Gupta, J. Kukula, and O. Strichman. Sat based abstraction-refinement using ILP and machine learning techniques. In *Proc. of Conference on Computer-Aided Verification (CAV'02)*, LNCS, 2002. To appear in this volume.
10. F. Copty, L. Fix, E. Giunchiglia, G. Kamhi, A. Tacchella, and M. Vardi. Benefits of bounded model checking at an industrial setting. In *Proc. of CAV 2001*, LNCS, pages 436–453, 2001.
11. R. Eshuis and R. Wieringa. Verification support for workflow design with UML activity graphs. In *Proc. of ICSE*, 2002. To appear.
12. A. Fuxman, M. Pistore, J. Mylopoulos, , and P. Traverso. Model checking early requirements specifications in Tropos. In *Proc. of the Fifth IEEE International Symposium on Requirements Engineering (RE'01)*, Toronto, August 2001.
13. E. Giunchiglia, M. Maratea, A. Tacchella, and D. Zambonin. Evaluating search heuristics and optimization techniques in propositional satisfiability. In *Proc. of IJCAR 2001*, volume 2083 of *LNCS*, pages 347–363. Springer, 2001.
14. The Gnu Lesser General Public License: http://www.fsf.org/licenses/lgpl.html.
15. K.L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publ., 1993.
16. M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an Efficient SAT Solver. In *Proc. of the 39th Design Automation Conference*, June 2001.
17. The Open Source Organization. http://www.opensource.org.
18. R. K. Ranjan, A. Aziz, B. Plessier, C. Pixley, and R. K. Brayton. Efficient BDD algorithms for FSM synthesis and verification. In *Proc. IEEE/ACM International Workshop on Logic Synthesis*, Lake Tahoe (NV), May 1995.
19. O. Shtrichman. Tuning SAT checkers for bounded model-checking. In *Proc. 12th International Computer Aided Verification Conference (CAV'00)*, 2000.