`prod` 3.2
An Advanced Tool for Efficient Reachability Analysis

Kimmo Varpaaniemi, Keijo Heljanko and Johan Lilius
{Kimmo.Varpaaniemi,Keijo.Heljanko,Johan.Lilius}@hut.fi

Helsinki University of Technology, Digital Systems Laboratory

Abstract. `prod` is a reachability analyzer for Predicate/Transition Nets. The tool incorporates several advanced reduced reachability graph generation methods. The tool also includes a CTL model checker and supports on-the-fly verification of LTL formulas. `prod` is being used in industrial projects at the Digital Systems Laboratory.

1 Intro

Reachability analysis is a powerful formal way to analyze concurrent and distributed systems such as telecommunication protocols. However reachability analysis suffers from the *state-explosion problem*: the state space of the system can be far too large with respect to the time and other resources needed to inspect all states in the space.

Fortunately, errors can be detected in a variety of cases without inspecting all reachable states of the system. The *stubborn set method* [11],[10] is one of the methods that try to relieve the state space explosion. On the other hand in *on-the-fly verification of a property* the property is verified during state space generation, in contrast to the traditional approach where properties are verified after state space generation.

The Pr/T-net reachability analysis tool `prod` [13], developed at the Digital Systems Laboratory at Helsinki University of Technology, implements among other methods the two above mentioned methods for efficient reachability analysis.

The aim of this article is to give a brief overview of the basic features of the tool and shortly describe some of its applications.

2 The reachability analyzer `prod`

`prod` is a command-line driven tool, ie. the user interacts with the tool from a Unix-shell. Currently no graphical user-interface exists. The tool consists of 5 components:

1. the control program `prod`, a make-like program that is used to execute the other components of the tool,
2. the reachability graph generator generator `prpp`, a program that reads a net description given in the Net Description Language, and generates a C-program that when compiled and run generates the reachability graph,
3. the query program `probe`, a tool for navigating the reachability graph,
4. a tool for calculating strongly connected components (`strong`), and
5. an interface to ARA, `araprod`, which will not be discussed further in this article.

Given a file `net.net` with a net description the command `prod net.init` executes `prpp` on the net-file `net.net`. `prpp` generates a C-file `net.c` together with some other data-files, that are then compiled and linked by the native C-compiler to an executable `net`. The program `net` is the reachability graph generator proper. The executable `net` takes a number of options, of which we shall describe the ones relating to reduced reachability graph generation.

`prod` supports the following reduced reachability graph generation methods:

- The *stubborn sets method* [11] is based on the idea that for the verification of certain properties of the net it is unnecessary to generate all possible interleavings of independent transitions. In `prod` the calculation of the stubborn set can be done with two main algorithms:
  1. The incremental algorithm (option `-s`), selects one enabled transition and then looks for other transitions to include so that the stubbornness property is retained.
  2. The deletion algorithm (option `-d`) starts with the set of all transitions deletes one transitions from it and the looks for other transitions to delete so that the stubbornness property is retained.
- The *CFFD preserving stubborn sets method* (option `-C`) is a version of stubborn set that preserve all chaos-free traces, failures and divergences of the net [10].
- *Sleep sets* (option `-S`) are an alternative method that also utilizes the independence of transitions to reduce the reachability graph. Sometimes sleep sets used in conjunction with stubborn sets can increase the reduction.
- *Symmetries* (option `-e`) are equivalence classes of markings. The calculation of symmetries can be very time-consuming, but it is well known that symmetries together with stubborn set can reduce the reachability graph radically.

On-the-fly verification of a property means that the property is verified during state space generation, in contrary to the traditional approach where properties are verified after state space generation. On-the-fly verification of *linear time temporal properties* [2],[12] with the aid of the CFFD preserving stubborn set method has been implemented in `prod`.

`prod` 3.2 also contains a branching time temporal logic CTL model checker, which is implemented as a part of the `probe` reachability graph navigator tool. It has a new global CTL model checking algorithm [3], which contains a counterexamples and witnesses facility.

3 Applications of `prod`

`prod` is and has been used in several academic and industrial projects at the Digital system laboratory. During the last year `prod` has been downloaded to over 200 sites worldwide.

- *The analysis of the FSR:* The Frame Synchronized Ring (FSR) is a high speed parallel data bus designed for high-throughput applications like ATM-switches. The Medium Access Control Algorithm of the FSR was analyzed for deadlock freeness for arbitary number of nodes, fairness and maximal waiting times [5], [6], [7].
- *The Emma project:* The Emma (Extendible Multi-Method Analyzer) project [4] has developed a dynamic analyzer for TNSDL programs (TeleNokia SDL), which supports the detection of some errors related to parallelism and the investigation of their causes. The Emma analyzer is designed on top of Nokia's TNSDL translator and `prod`. The user of Emma is working only on the level of TNSDL.
- *Teaching:* `prod` has been used successfully as a teaching aid in a course on Parallel and Distributed systems. The students do a verification project in groups of 2 or 3. The course is taken by approximately 100 students yearly.
- *Other applications:* Several other substantial models have been built and analyzed with `prod` including the TCAP/TSL-protocol, a video on demand system [8], [9], an authentication protocol [1], and a specification of a simple telephone exchange (YXA).

4 Availability

`prod` is available over the Internet. The sources to the tool and a bibliography with relevant literature about `prod`, and other Petri net research done at the Digital systems laboratory can be found at
`http://topos.hut.fi/~petrinet/`.

References

1. Tuomas Aura. Modelling the Needham-Schröder authentication protocol with high level Petri nets. Technical Report B14, Helsinki University of Technology, Digital Systems Laboratory, Espoo, Finland, September 1995.
2. C. Courcoubetis, M.Y. Vardi, P. Wolper, and M. Yannakakis. Memory efficient algorithms for the verification of temporal properties. *Formal Methods in System Design*, 1(2/3):275–288, 1992.
3. K. Heljanko. Model Checking the Branching Time Temporal Logic CTL. Research Report A45, Helsinki University of Technology, Digital Systems Laboratory, Espoo, Finland, 1997.
4. N. Husberg. SDL Modelling with High Level Petri Nets. In *Proceedings of the Fifth International Workshop on Concurency, Specification & Programming (CSP), Berlin, Germany, September 25–27, 1996*, pages 85–96, Berlin, 1996. Humboldt-University Berlin.
5. Tino Pyssysalo. Proving Properties of a New High Speed Data Bus with Predicate/Transition Nets. *Microprocessing and Microprogramming*, 40:791–794, 1994.
6. Tino Pyssysalo. The Modeling and Analysis of the Frame Synchronized Ring Using the Predicate/Transition Net Formalism. In *Proceedings of COST 247 Meeting, Berlin, Germany, February 9–10, 1995*, page 6, Berlin, 1995. Humboldt-University Berlin.
7. Tino Pyssysalo. An Induction Theorem for Ring Protocols of Processes Described with Predicate/Transition Nets. Research Report A37, Digital Systems Laboratory, Helsinki University of Technology, 1996.
8. Tino Pyssysalo and Leo Ojala. Modelling of a "Video on Demand" System Using Pr/T-Net Formalism—a Case Study. In *Proceedings of the Third International Workshop on Concurency, Specification & Programming (CSP), Berlin, Germany, October 12–14, 1994*, page 8, Berlin, 1994. Humboldt-University Berlin.
9. Tino Pyssysalo and Leo Ojala. Causal Modeling of a Video on Demand System Using Predicate/Transition Net Formalism. In *Proceedings of the 22nd Euromicro Conference, Prague, Czech Republik, September 2–5, 1996*, page 6, Los Alamitos, California, 1996. The Institute of Electrical and Electronics Engineers (IEEE) Computer Society Press.
10. A. Valmari. Alleviating state explosion during verification of behavioral equivalence. Report A-1992-3, University of Helsinki, Department of Computer Science, 1992.
11. A. Valmari. A stubborn attack on state explosion. *Formal Methods in System Design*, 1(4):297–322, 1992.
12. M.Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proceedings of LICS'86*, pages 332–344. IEEE Computer Society Press, 1986.
13. Kimmo Varpaaniemi, Jaakko Halme, Kari Hiekkanen, and Tino Pyssysalo. PROD reference manual. Technical Report B13, Helsinki University of Technology, Digital Systems Laboratory, Espoo, Finland, August 1995.