The PRISM Benchmark Suite

Marta Kwiatkowska
Department of Computer Science, University of Oxford, UK
marta.kwiatkowska@cs.ox.ac.uk

Gethin Norman
School of Computing Science, University of Glasgow, UK
gethin.norman@glasgow.ac.uk

David Parker
School of Computer Science, University of Birmingham, UK
d.a.parker@cs.bham.ac.uk

***Abstract*—We present the PRISM benchmark suite: a collection of probabilistic models and property specifications, designed to facilitate testing, benchmarking and comparisons of probabilistic verification tools and implementations.**

***Keywords*-Benchmarks; Probabilistic models; Probabilistic model checking; Probabilistic verification tools.**

I. Introduction

Probabilistic verification techniques are increasingly being used for the formal modelling and analysis of systems that exhibit stochastic behaviour. They have been deployed in a diverse range of application areas, from wireless communication protocols, to robotic planning and control, to systems biology. As for any kind of formal verification, developing efficient and scalable techniques remains a key research challenge, and a large amount of effort has been invested in this area in recent years.

A variety of probabilistic verification tools have been produced, the most widely used of which is the PRISM model checker [1]. Many models for use with PRISM are publicly available, either distributed with the tool itself, included in the case study repository [3], or described in publications. Hence, these models are commonly used to test, evaluate and benchmark new probabilistic verification techniques and tools, as they are developed. Various other tools also use the PRISM modelling language, or slight extensions of it, as an input format for models, for example, PASS, INFAMY, PARAM, APMC and Ymer.[1]

However, despite the availability of PRISM models, there are often multiple variants of each one, distributed across various different sources. Furthermore, it is not always easy to identify a particular class of probabilistic models or property specifications for those models, as required to test a specific verification or analysis technique. In an attempt to address this situation, we present the PRISM benchmark suite [4]: a unified collection of PRISM models and properties, grouped by type, specifically intended to facilitate testing, benchmarking and comparisons of probabilistic model checking tools and implementations.

[1]For pointers to these, as well as to the various other currently available probabilistic verification tools, see [2].

II. Models

The core part of the PRISM benchmark suite is a set of probabilistic models, mostly sourced from existing probabilistic verification case studies. These are specified in the PRISM modelling language, a textual language based on the Reactive Modules formalism, used to specify models for the tool. As described below, it is straightforward to convert these high-level descriptions into their corresponding underlying models for input to other tools.

So far, there are 30 models in the benchmark suite, covering the four types of probabilistic models currently supported by PRISM:

- discrete-time Markov chains (DTMCs);
- continuous-time Markov chains (CTMCs);
- Markov decision processes (MDPs);
- probabilistic timed automata (PTAs).

Further model types will be added as the functionality of PRISM develops. For each model, we provide:

- a short *name*, for easy citation in publications;
- *references* to the original source of the PRISM model, and/or papers describing the system being modelled;
- the PRISM language *model description*;
- details of *parameters*, which can be used to generate models of varying size and complexity.

For convenience, we tabulate information about the size of each model (number of states), for a range of different parameter values. This makes it easy to locate models of a particular type and size for experiments.

The benchmark suite is also intended to be a useful source of models for tools and techniques that do not use the PRISM modelling language as an input. Using PRISM, it is straightforward to build and then export the underlying probabilistic model (Markov chain or Markov decision process) for a benchmark model, along with accompanying information needed for model checking:

- the transition matrix/function for the model;
- the set of all reachable states;
- the set of states satisfying a given label/predicate;
- the associated state/transition rewards.

This information can be exported in various different forms, including the input format for either Matlab or the probabilistic model checker MRMC, or a simple plain-text format

Table I
CLASSES OF PROPERTY SPECIFICATION INCLUDED IN THE BENCHMARK SUITE

Property class	Example PRISM notation	Applicable models
probabilistic reachability	`P=? [ F "target" ]`	DTMCs/CTMCs/MDPs/PTAs
time-bounded probabilistic reachability	`P=? [ F<=T "target" ]`	DTMCs/CTMCs/PTAs
expected cumulated reward to reach a target	`R=? [ F "target" ]`	DTMCs/CTMCs/MDPs
expected cumulated reward within a time bound	`R=? [ C<=T ]`	DTMCs/CTMCs
steady-state probability	`S=? [ "pred" ]`	DTMCs/CTMCs

that can easily be converted to other model/matrix formats. Alternatively, since PRISM is an open source tool, it can easily be adapted to generate such files directly.

III. PROPERTIES

Each model in the benchmark suite is accompanied by several properties that can be verified against it. The properties are given in the PRISM property specification language, a temporal logic-based notation that subsumes various existing logics, such as PCTL, CSL, LTL and PCTL*.

Primarily, the properties included are of a quantitative (numerical) nature, i.e., they are of the form `P=? [ F<=10 "err" ]` ("what is the probability of an error occurring within 10 time-steps?"), rather than, e.g., `P<=0.1 [ F<=10 "err" ]` ("the probability of an error occurring within 10 time-steps is at most 0.1").

Properties are grouped by target model and by type. Currently, a selection of common classes of property is included, summarised in Table I. Although the properties are all expressed in PRISM notation, they typically fit one of the templates shown in the middle column of Table I. Thus, PRISM can be used to identify and export the states satisfying required predicates (such as `"target"` and `"pred"` in the table), and these can be read in by separate tools, without the need to implement property parsing. Other commonly used data for probabilistic models, such as steady-state or transient probability distributions, can also easily be generated for the benchmark models.

IV. TESTING & BENCHMARKING

The contents of the benchmark suite are specifically designed for testing and benchmarking.

Firstly, for testing purposes, each property is annotated with the expected result of model checking. Typically, several different results are given, for various different model or property parameters. The results are embedded in a comment before each property, e.g.:

```
// RESULT (N=2): 0.02
// RESULT (N=3): 0.03
P=? [ F<=10 "err" ]
```

PRISM itself includes a test mode (enabled using the command-line switch `-test`), which extracts these values and ensures that the computed value matches.

Secondly, the benchmark suite includes a script, called `prism-auto`, which automates the process of running a large set of benchmark problems. It identifies a set of models and properties (either from a specified list or by recursively searching one or more directories), executes PRISM (or some other probabilistic model checking tool) on each model/property found and automatically generates log files for later inspection. Lists for the various classes of properties described in the previous section are provided in the benchmark suite. The `prism-auto` script can also be supplied with a set of switches/options over which the models/properties are to be compared. Additional scripts to help automatically generate tables summarising the results of benchmarking are also available.

V. FUTURE WORK

The PRISM benchmark suite is freely available now from [4]. The set of models and properties will be expanded as more become available; external contributions are also welcome. We plan to add further categories of property (for example, LTL and automata-based specifications) and also specialised classes of models (for example, systems biology models) in due course. The range of types of models and properties included will also be enlarged as the capabilities of PRISM are extended.

ACKNOWLEDGEMENTS

This work has been supported in part by the ERC Advanced Grant VERIWARE, EU-FP7 project CONNECT and EPSRC grant EP/F001096/1.

REFERENCES

[1] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of probabilistic real-time systems," in *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, ser. LNCS, G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806. Springer, 2011, pp. 585–591.

[2] http://www.prismmodelchecker.org/other-tools.php.

[3] http://www.prismmodelchecker.org/casestudies/.

[4] http://www.prismmodelchecker.org/benchmarks/.