PRINSYS – a Software Tool for the Synthesis of Probabilistic Invariants ⋆

Friedrich Gretz[1,2], Joost-Pieter Katoen[1], and Annabelle McIver[2]

[1] RWTH Aachen University, Germany
[2] Macquarie University, Sydney, Australia

Abstract

We are interested in aiding correctness proofs for probabilistic programs, i.e. WHILE programs, enriched with a probabilistic choice operator "$[p]$" that executes the left alternative with probability p and the right alternative with $1-p$. There are tools for non-probabilistic programs that generate invariants for verification purposes [2, 1]. For probabilistic programs the existing tools rely on model checking and are limited to finite-state systems or do not allow parameters [6, 4, 3]. One tool that is based on abstract interpretation is mentioned in [7] but its merits cannot be assessed[3]. Our novel tool PRINSYS[4] is based on some of the ideas in [5] and aids the verification of *probabilistic programs with loops and real valued variables*. We can derive quantitative properties of a given program by reasoning over program annotations in the style of Hoare logic. Our annotations are generalised to take account of the quantitative properties of probabilistic programs. PRINSYS computes sound annotations for probabilistic while loops.

Our approach to generate invariants is constraint-based. Given a template, i.e. the shape of the desired invariant, and a loop we generate constraints in terms of inequalities between piecewise functions. These constraints are then transformed into universally quantified first-order formulas and solved using off the shelf tools. The result is the description of all invariant instances of the template that was given by the user. Using this semi-automatical approach iteratively, a user can construct an invariant that helps him to prove a property of the program.

In the workshop we would like to report about the latest ongoing work and show some use cases of PRINSYS.

References

1. Colón, M., Sankaranarayanan, S., Sipma, H.: Linear Invariant Generation Using Non-linear Constraint Solving. In: CAV. pp. 420–432 (2003)

⋆ This work is supported by the EU FP7 project CARP, DFG research training group Algosyn, Australian Research Council DP1092464 and NWO visitor grant 040.11.302.

[3] ABSINTHE appears to be unmaintained and no documentation or download could be found as of this writing.

[4] Download the tool at: http://www-i2.informatik.rwth-aachen.de/prinsys/.

2. Gupta, A., Rybalchenko, A.: InvGen: An Efficient Invariant Generator. In: Bouajjani, A., Maler, O. (eds.) CAV. LNCS, vol. 5643, pp. 634–640. Springer (2009)
3. Hahn, E.M., Hermanns, H., Wachter, B., Zhang, L.: Pass: Abstraction refinement for infinite probabilistic models. In: Esparza, J., Majumdar, R. (eds.) TACAS. Lecture Notes in Computer Science, vol. 6015, pp. 353–357. Springer (2010)
4. Hahn, E.M., Hermanns, H., Zhang, L.: Probabilistic Reachability for Parametric Markov Models. STTT 13(1), 3–19 (2011)
5. Katoen, J.P., McIver, A., Meinicke, L., Morgan, C.: Linear-Invariant Generation for Probabilistic Programs. In: SAS, pp. 390–406 (2011)
6. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of Probabilistic Real-time Systems. In: CAV. pp. 585–591 (2011)
7. Monniaux, D.: An Abstract Analysis of the Probabilistic Termination of Programs. In: Cousot, P. (ed.) SAS. LNCS, vol. 2126, pp. 111–126. Springer (2001)