Software Modeling and Verification Chair | RWTH AACHEN UNIVERSITY

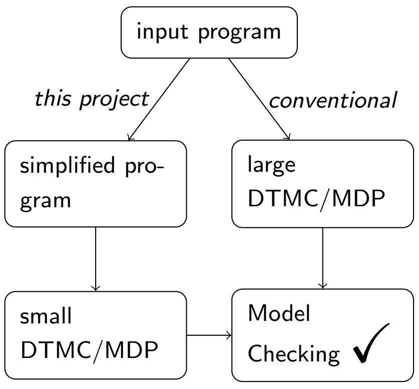
— Master's Thesis —

Static Analyses for Probabilistic Model Simplification

What is it all about?

Probabilistic models such as Discrete Time Markov Chains (DTMCs) and Markov Decision Processes (MDPs) are versatile tools for the analysis of many real-world scenarios. Examples include lossy communication systems, randomized distributed protocols, error-recovery in unrealiable hardware, physical and biological system analyses and many more. Probabilistic Model Checking is a technique concerned with the automatic formal verification of the aforementioned models. The input of probabilistic model checkers such as Storm [DJKV17] is typically specified by means of a human-readable program with finite variable domains to ensure decidability of all non-trivial properties. The state space of the DTMC or MDP to be analyzed is implicitly spanned by the Cartesian product of all variable domains and is thus exponentially large in the number of variables – this phenomenon is often referred to as the state explosion problem in Model Checking.

The main goal of this Master's thesis is to mitigate the state explosion problem to make (probabilistic) Model Checking more applicable. Existing software tools build the underlying DTMC/MDP (more or less) directly from the input program. In this project, a more refined approach based on static analysis of the program at hand should be investigated.



What is to be done?

Instead of building the (large) underlying DTMC/MDP directly from the input program, static analysis techniques should be applied to first simplify the program and thus allow for a smaller explicit state model. One promising direction is the elimination of probabilistic transitions on the program(-graph) level. To this end, single variables (not all at once) may be unfolded into the program-graph's states so that classical state elimination can be applied already on the high-level model. Research questions include:

1. Which variables can be unfolded? In which order?
2. Which properties are preserved by this kind of transformation?
3. Can we make use of traditional static analysis techniques such as data flow analysis or abstract interpretation?
4. How well do the proposed methods perform on practically relevant benchmarks?

This list is of course non-exhaustive! To answer the last question in particular, the techniques you develop should be implemented in our probabilistic Model Checker Storm [DJKV17], which is written in C++.

What we expect:

- Solid background in theoretical computer science – ideally you have already taken theoretical CS electives.
- Passion and endurance for solving difficult theoretical problems.
- Willingness to work with a larger C++ code base.

What you can expect:

- Get a chance to work on relevant problems of both theoretical and practical nature.
- Work closely together with us – you can work in our student's room at the chair whenever you like and enjoy our coffee machine!

Contact

- Tobias Winkler, tobias.winkler@cs.rwth-aachen.de

References

[DJKV17] Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk, A storm is coming: A modern probabilistic model checker, Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part II, 2017, pp. 592–600.