DD2552 Fall 2021
Homework

Karl Palmskog

Abstract

Due: January 14, 2022, at 23:59 CET. Submit your solutions as a PDF file by e-mail to `palmskog@kth.se`. State your name and e-mail address at the very top of the first PDF page. Discussions of ideas in groups of two people are allowed, but you should write down your own solutions individually. You should also acknowledge any collaboration. Some of the problems are meant to be quite challenging and you are not necessarily expected to solve all of them. A total score of 60 is enough for grade FX, 70 points for grade E, 100 points for grade D, 120 points for grade C, 140 points for grade B, and 160 points for grade A. The statistical model checking survey used in the course (`https://doi.org/10.1145/3158668`) is referenced in several exercises.

1. Path and State CTL Formulas (20p). Consider the original non-probabilistic CTL (`https://dl.acm.org/doi/pdf/10.1145/5397.5399`):

$$\varphi ::= \top \mid a \mid \neg\varphi \mid \varphi \wedge \varphi \mid AX\varphi \mid EX\varphi \mid A[\varphi\, U\, \varphi] \mid E[\varphi\, U\, \varphi]$$

- Show how to break up formulas φ into state formulas ϕ and path formulas ψ.
- Define the semantics, in the usual form of a relation $\mathcal{M}, s \models \_$, of ϕ and ψ for $\mathcal{M}$ a non-probabilistic transition system model, i.e., a *DTMC without probabilities*.
- Define a translation function that takes any formula φ and returns an equivalent formula ϕ or ψ.
- Prove that the translation of $A[a_0\, U\, a_1]$ is true for a system $\mathcal{M}$ in your semantics whenever the original formula is true for $\mathcal{M}$ in the original CTL semantics (defined in the paper referenced above).

2. Extended PCTL definition and semantics (10p). Consider the PCTL syntax fragment usually used in the course:

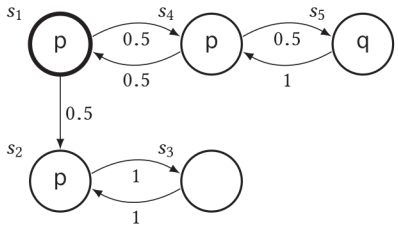
$$\phi ::= \top \mid a \mid \neg\phi \mid \phi \wedge \phi \mid P_{\geq\theta}(\psi)$$
$$\psi ::= \phi\, U^{\leq t}\, \phi$$
$$t \in Z^{\geq 0},\ \ \theta \in [0,1]$$

Define an extended PCTL syntax which also includes:

- implication ($\rightarrow$)
- disjunction ($\vee$)
- unbounded until (U)
- weak bounded until ($W^{\leq t}$)
- weak unbounded until (W)
- probability operators for less than, greater than, and less than or equal to ($P_{<\theta}$, $P_{>\theta}$, $P_{\leq\theta}$)

Then, define extensions of the semantic relations $\mathcal{M}, s \models \_$ and $\mathcal{M}, \pi \models \_$ for the new syntax, where $\mathcal{M}$ is a DTMC. Please follow the general approach used in the paper by Hansson and Jonsson (`https://link.springer.com/article/10.1007/BF01211866`) and do not transcribe definitions from other papers or other lecture notes on the web.

3. A DTMC as a program (10p). Consider the DTMC below. Define this DTMC programmatically in PRISM's textual language (see examples: `https://www.prismmodelchecker.org/manual/Appendices/ExplicitModelFiles`). Formulate a true, meaningful PCTL formula for the DTMC in PRISM specification syntax that includes a probability operator and a non-zero/non-one bound (in particular, do not use `=?`). Give an intuitive explanation why the formula is true.



M	s_1	s_2	s_3	s_4	s_5
s_1	0	0.5	0	0.5	0
s_2	0	0	1	0	0
s_3	0	1	0	0	0
s_4	0.5	0	0	0	0.5
s_5	0	0	0	1	0

4. Programmatic DTMC/CTMC for SMC (10p). What is the advantage for *statistical* verification of PCTL/CSL properties to have a programmatic definition of a DTMC/CTMC (as opposed to, for example, a matrix-based representation)?

5. Curtailed sampling plan (10p). Consider statistical verification of a stochastic system for a "good" path property ψ using a *curtailed sampling plan* (page 30–31 in `https://doi.org/10.1016/j.ic.2006.05.002`). Define the indifference interval by setting $2\delta = 10^{-5}$ and error probabilities by setting $\alpha = \beta = 10^{-8}$. Assume the underlying probability p of ψ being true is 0.9. What is the probability that we will accept the system after we have carried out the curtailed sampling plan? Motivate your answer carefully.

6. Equivalence of CTMC path definitions (10p). Let $\mathcal{M}$ be a CTMC and π be a path of $\mathcal{M}$ starting in s. We have two possible definitions of π:

- As a total function from positive real numbers to states in $\mathcal{M}$.
- As an infinite sequence of states in $\mathcal{M}$ with labels of positive real values between adjacent pairs of states.

Show that these two definitions are equivalent.

7. CTMCs and rate matrix diagonals (10p). In the usual definition of CTMCs, diagonal elements $Q(s_j, s_j)$ in the rate matrix Q are constrained to be $-(\sum_{j \neq k} Q(s_k, s_j))$. Explain why this constraint is used, and what can happen with a "CTMC" whose Q does not satisfy this constraint. Is the model checking problem for the logic CSL still well defined for such CTMCs?

8. CSL and QuaTEx (10p). Consider the QuaTEx language defined in reference 5 in the survey (`https://doi.org/10.1016/j.entcs.2005.10.040`). Make a succinct comparison between, on one hand, PCTL/CSL as defined in the course, and, on the other hand, QuaTEx. Consider in particular expressive power and supported operators. Does the comparison change if you instead of the course PCTL/CSL consider PCTL/CSL as defined by PRISM?

9. Runtime verification and SMC (10p). Runtime verification is a system analysis approach based on extracting information from a running

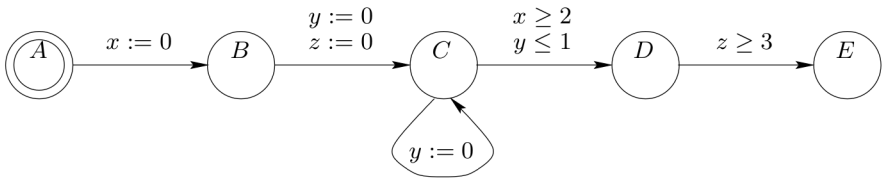
system and using it to detect observed behaviors satisfying or violating certain properties (see for example Wikipedia or reference 49 in the survey).

Explain how runtime verification can be useful when implementing statistical model checking, and give an example of a workflow, i.e., a concrete sequence of steps, applying statistical model checking and runtime verification to a (small) DTMC model and (simple) PCTL formula. You do not have to explain any details on how the statistical tester works.

10. MDPs modelling a distributed system (20p). Assume you have n (independent) DTMCs $\mathcal{M}_1, \ldots, \mathcal{M}_n$. Consider the Markov Decision Process which nondeterministically interleaves transitions from these DTMCs. For example, the scheduler can select the DTMC $\mathcal{M}_i$ and perform a probabilistic transition $(s_1, \ldots, s_i, \ldots, s_n) \to (s_1, \ldots, s'_i, \ldots, s_n)$, where s_i and s'_i are states in $\mathcal{M}_i$. Show how this MDP can be defined using *action labels* in the style that is used to define MDPs in the PRISM tool (`https://www.prismmodelchecker.org/manual/Appendices/ExplicitModelFiles`).

Note: you should give a general abstract MDP definition for any n, **not** a programmatic definition in PRISM's language or similar.

11. PRISM style PTAs (10p). Consider the timed automaton below. Define this automaton as a PTA as accurately as you can in PRISM's language for PTAs (you may have to decorate some transitions with reasonable probabilities). Define a PRISM-compatible non-probabilistic specification for the automaton that captures reachability of the right-most end state from the left-most starting state (see examples at `http://www.prismmodelchecker.org/benchmarks/props-pta.php`). Then, formulate a PRISM-compatible specification describing the *minimum probability* of reaching the end state from the start state.



12. Nested PCTL probability operators (10p). Let `send` and `end` be atomic state formulas, and let $\top$ be the usual the state formula that is always true. Consider the following PCTL property with a nested probability

operator:

$$P_{\geq 0.8}(P_{\geq 0.1}(\top\, U^{\leq 1}\, \texttt{end})\, U^{\leq 3}\, \texttt{send})$$

Construct a DTMC with at least five states for which this property is true in the starting state. Ensure that there is at least one path from the starting state with measure greater than zero for your DTMC where the topmost path property (obtained by removing $P_{\geq 0.8}$) is false.

You do not need to directly prove the nested property using the PCTL semantics on your DTMC, but you should explain carefully why the property holds.

13. PCTL Path Prefix Bound (20p). Consider again the usual course PCTL fragment, which allows nesting of probability operators:

$$\begin{aligned}\phi &::= \top \mid a \mid \neg\phi \mid \phi \wedge \phi \mid P_{\geq\theta}(\psi)\\ \psi &::= \phi\, U^{\leq t}\, \phi\\ & \quad t \in Z^{\geq 0},\ \ \theta \in [0,1]\end{aligned}$$

Recall that to determine whether a path π satisfies a path property ψ, we only need to observe a finite prefix of π. We now want to compute an *upper bound* on the size of this prefix for an arbitrary ψ. For example, for $\psi = \top\, U^{\leq 2} a$, we need (at most) a prefix of length 2, since a is atomic.

a) Define a general algorithm that computes the path prefix upper bound as an integer, given an arbitrary formula ψ. It is highly recommended that your algorithm is defined recursively on formula syntax. Briefly motivate why your algorithm is correct, i.e., why we would never need to look further in a path to determine whether ψ holds.

b) Motivate briefly why the output of your algorithm is indeed just an *upper* bound, and how you could get away with using smaller prefixes in special cases.

14. Hypotheses and random variables for hypothesis testing (10p). Let $\mathcal{M}$ be a DTMC, and $P_{\geq 0.9}(\psi)$ be the specification for (the initial state of) $\mathcal{M}$. Let the indifference interval be given by $\delta = 0.02$, and call the underlying path measure/probability p (for ψ).

a) Formulate as concretely as possible the two competing hypotheses one should use for performing statistical hypothesis testing on $\mathcal{M}$ to determine whether the specification holds.

b) Let X_i be the Bernoulli random variable and x_i be the actual outcome (1 for true and 0 for false) associated with checking ψ on the ith sampled trace from $\mathcal{M}$, out of a total of n. Let Y be the random variable that is the sum of all X_i, i.e., $Y = \sum_{i=1}^{n} X_i$. Consider the probability that $Y \leq c$, for c a constant. Give an expression of this probability in terms of c, n, and p.

15. Implementation of simulation (10p). Suppose you want implement a simulator to use in statistical model checking (SMC) of PCTL properties on DTMCs.

a) Explain in detail the performance advantages of implementing DTMC simulation at the level of a programming language, i.e., why users may want to take the time to define a DTMC in this way rather than as a matrix.

b) Assume your simulator has been shown to guarantee that, given a DTMC representation, the output is a trace for the input DTMC. Is this in itself enough for your simulator to be applicable for use in SMC? Explain why or why not, and if not, give an example of what can go wrong.