Counterexamples in Probabilistic Model Checking

Tingting Han[1,2] and Joost-Pieter Katoen[1,2]

[1] Software Modelling and Verification, RWTH Aachen, Germany
[2] Formal Methods and Tools, University of Twente, The Netherlands
Email: {tingting.han, katoen}@cs.rwth-aachen.de

Abstract. This paper considers algorithms for counterexample generation for (bounded) probabilistic reachability properties in fully probabilistic systems. Finding the strongest evidence (i.e, the most probable path) violating a (bounded) until-formula is shown to be reducible to a single-source (hop-constrained) shortest path problem. Counterexamples of smallest size that are mostly deviating from the required probability bound can be computed by adopting (partially new hop-constrained) k shortest paths algorithms that dynamically determine k.

1 Introduction

A major strength of model checking is the possibility to generate counterexamples in case a property is violated. The shape of a counterexample depends on the checked formula and the used temporal logic. For logics such as LTL, typically paths through the model suffice. The violation of linear-time safety properties is indicated by finite path fragments that end in a "bad" state. Liveness properties, instead, require infinite paths ending in a cyclic behavior indicating that something "good" will never happen. LTL model checkers usually incorporate breadth-first search algorithms to generate *shortest* counterexamples, i.e., paths of minimal length. For branching-time logics such as CTL, paths may act as counterexample for a subclass of universally quantified formulae, ACTL$\cap$LTL, to be exact. To cover a broader spectrum of formulae, though, more advanced structures such as trees of paths [11], proof-like counterexamples [18] (for ACTL$\setminus$LTL) or annotated paths [26] (for ECTL) are used.

Counterexamples are of utmost importance in model checking: first, and for all, they provide diagnostic feedback even in cases where only a fragment of the entire model can be searched. They constitute the key to successful abstraction-refinement techniques [10], and are at the core of obtaining feasible schedules in e.g., timed model checking [8]. As a result, advanced counterexample generation and analysis techniques have intensively been investigated, see e.g., [21,7,13].

This paper considers the generation of counterexamples in probabilistic model checking. Probabilistic model checking is a technique to verify system models in which transitions are equipped with random information. Popular models are discrete- and continuous-time Markov chains (DTMCs and CTMCs, respectively), and variants thereof which exhibit nondeterminism. Efficient model-checking algorithms for these models have been developed, have been implemented in a variety of software tools, and have been applied to case studies from various application areas ranging from randomized distributed algorithms, computer systems and security protocols to biological systems

and quantum computing. The crux of probabilistic model checking is to appropriately combine techniques from numerical mathematics and operations research with standard reachability analysis. In this way, properties such as "the (maximal) probability to reach a set of goal states by avoiding certain states is at most 0.6" can be automatically checked up to a user-defined precision. Markovian models comprising millions of states can be checked rather fast.

In probabilistic model checking, however, counterexample generation is almost not developed; notable exception is the recent heuristic search algorithm for CTMCs and DTMCs [3,4] that works under the assumption that the model is unknown. Instead, we consider a setting in which it has already been established that a certain state refutes a given property. This paper considers algorithms and complexity results for the generation of counterexamples in probabilistic model checking. The considered setting is probabilistic CTL [19] for discrete-time Markov chains (DTMCs), a model in which all transitions are equipped with a probability. In this setting, typically there is no single path but rather a *set* of paths that indicates why a given property is refuted. We concentrate on properties of the form $\mathcal{P}_{\leqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi)$ where p is a probability and h a (possibly infinite) bound on the maximal allowed number of steps before reaching a goal (i.e., a Ψ-) state. In case state s refutes this formula, the probability of all paths in s satisfying $\Phi \mathcal{U}^{\leqslant h} \Psi$ exceeds p. We consider two problems that are aimed to provide useful diagnostic feedback for this violation: generating strongest evidences and smallest, most indicative counterexamples.

Strongest evidences are the most probable paths that satisfy $\Phi \mathcal{U}^{\leqslant h} \Psi$. They "contribute" mostly to the property refutation and are thus expected to be informative. For unbounded until (i.e., $h{=}\infty$), determining strongest evidences is shown to be equivalent to a standard single-source shortest path (SP) problem; in case h is bounded, we obtain a special case of the (resource) constrained shortest path (CSP) problem [2] that can be solved in $\mathcal{O}(hm)$ where m is the number of transitions in the DTMC. Alternatively, the Viterbi algorithm can be used for bounded h yielding the same time complexity.

Evidently, strongest evidences may not suffice as true counterexamples, as their probability mass lies (far) below p. As a next step, therefore, we consider the problem of determining most probable subtrees (rooted at s). Similar to the notion of shortest counterexample in LTL model checking, we consider trees of *smallest size* that exceed the probability bound p. Additionally, such trees, of size k, say, are required to *maximally* exceed the lower bound, i.e., no subtrees should exist of size at most k that exceed p to a larger extent. The problem of generating such *smallest, most indicative counterexamples* can be casted as a k shortest paths problem. For unbounded-until formulae (i.e., $h{=}\infty$), it is shown that the generation of such smallest counterexamples can be found in pseudo-polynomial time by adopting k shortest paths algorithms [15,24] that compute k on the fly. For bounded until-formulae, we propose an algorithm based on the recursive enumeration algorithm of Jiménez and Marzal [20]. The time complexity of this adapted algorithm is $\mathcal{O}(hm{+}hk \log(\frac{m}{n}))$, where n is the number of states in the DTMC.

Finally, we show how the algorithms for $\mathcal{P}_{\leqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi)$ can be exploited for generating strongest evidences and counterexamples for lower bounds on probabilities, i.e., $\mathcal{P}_{\geqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi)$.

2 Preliminaries

DTMCs. Let AP denote a fixed, finite set of atomic propositions ranged over by $a, b, c, \ldots$. A (labelled) *discrete-time Markov chain* (DTMC) is a Kripke structure in which all transitions are equipped with discrete probabilities such that the sum of outgoing transitions of each state equals one. Formally, DTMC $\mathcal{D} = (S, \mathbf{P}, L)$ where S is a finite set of states, $\mathbf{P} : S \times S \to [0,1]$ is a stochastic matrix, and $L : S \to 2^{AP}$ is a labelling function which assigns to each state $s \in S$ the set $L(s)$ of atomic propositions that are valid in s. A state s in $\mathcal{D}$ is called absorbing if $\mathbf{P}(s,s) = 1$. W.l.o.g. we assume a DTMC to have a unique initial state.

Definition 1 (Paths). *Let $\mathcal{D} = (S, \mathbf{P}, L)$ be a DTMC.*

- *An* infinite path *σ in $\mathcal{D}$ is an infinite sequence $s_0 \cdot s_1 \cdot s_2 \cdot \ldots$ of states such that $\mathbf{P}(s_i, s_{i+1}) > 0$ for all $i \geqslant 0$.*
- *A* finite path *in $\mathcal{D}$ is a finite prefix of an infinite path.*

For state s and finite path $\sigma = s_0 \cdot s_1 \cdot \ldots \cdot s_n$ with $\mathbf{P}(s_n, s) > 0$, let $\sigma \cdot s$ denote the path obtained by extending σ by s. Let $|\sigma|$ denote the length of the path σ, i.e., $|s_0 \cdot s_1 \cdot \ldots \cdot s_n| = n$, $|s_0| = 0$ and $|\sigma| = \infty$ for infinite σ. For $0 \leqslant i \leqslant |\sigma|$, $\sigma[i] = s_i$ denotes the $(i{+}1)$-st state in σ. $\mathit{Path}(s)$ denotes the set of all infinite paths that start in state s and $\mathit{Path}_{\mathit{fin}}(s)$ denotes the set of all finite paths of s.

A DTMC $\mathcal{D}$ enriched with an initial state s_0 induces a probability space. The underlying σ-algebra from the basic cylinder is induced by the finite paths starting in s_0. The probability measure $\Pr^{\mathcal{D}}_{s_0}$ (briefly Pr) induced by $(\mathcal{D}, s_0)$ is the unique measure on this σ-algebra where:

$$\Pr\{\underbrace{\sigma \in \mathit{Path}(s_0) \mid s_0 \cdot s_1 \cdot \ldots \cdot s_n \text{ is a prefix of } \sigma}_{\text{basic cylinder of the finite path } s_0 \cdot s_1 \cdot \ldots \cdot s_n}\} \;=\; \prod_{0 \leqslant i < n} \mathbf{P}(s_i, s_{i+1}).$$

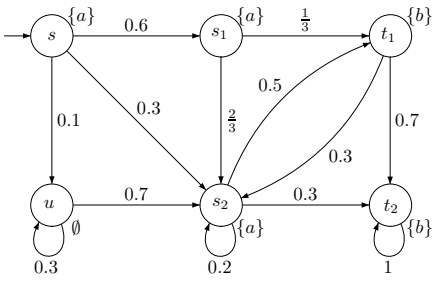


Fig. 1. An example DTMC

Example 1. Fig. 1 illustrates a simple DTMC with initial state s. $AP = \{a, b\}$ and L is given through the subsets of AP labelling the states as $L(s) = L(s_i) = \{a\}$, for $1 \leqslant i \leqslant 2$; $L(t_1) = L(t_2) = \{b\}$ and $L(u) = \emptyset$. t_2 is an absorbing state. $\sigma_1 = s \cdot u \cdot s_2 \cdot t_1 \cdot t_2$ is a finite path with $\Pr\{\sigma_1\} = 0.1 \times 0.7 \times 0.5 \times 0.7$ and $|\sigma_1| = 4$, $\sigma_1[3] = t_1$. $\sigma_2 = s \cdot (s_2 \cdot t_1)^{\omega}$ is an infinite path.

PCTL. Probabilistic computation tree logic (PCTL) [19] is a probabilistic extension of CTL in which state-formulae are interpreted over states of a DTMC and path-formulae are interpreted over paths in a DTMC. The syntax of PCTL is as follows:

$$\Phi ::= \mathsf{tt} \mid a \mid \neg\Phi \mid \Phi \wedge \Phi \mid \mathcal{P}_{\trianglelefteq p}(\phi)$$

where $p \in [0,1]$ is a probability, $\trianglelefteq \in \{<, \leqslant, >, \geqslant\}$ and ϕ is a path formula defined according to the following grammar:

$$\phi ::= \Phi \mathcal{U}^{\leqslant h} \Phi \mid \Phi \mathcal{W}^{\leqslant h} \Phi.$$

where $h \in \mathbb{N} \cup \{\infty\}$. The path formula $\Phi \mathcal{U}^{\leqslant h} \Psi$ asserts that Ψ is satisfied within h transitions and that all preceding states satisfy Φ. For $h{=}\infty$ such path-formulae are standard (unbounded) until-formulae, whereas in other cases, these are bounded until-formulae. $\mathcal{W}^{\leqslant h}$ is the weak counterpart of $\mathcal{U}^{\leqslant h}$ which does not require Ψ to eventually become true. For the sake of simplicity, we do not consider the next-operator. The temporal operators $\Diamond^{\leqslant h}$ and $\Box^{\leqslant h}$ are obtained as follows:

$$\mathcal{P}_{\trianglelefteq p}(\Diamond^{\leqslant h} \Phi) = \mathcal{P}_{\trianglelefteq p}(\mathrm{tt}\, \mathcal{U}^{\leqslant h}\, \Phi) \quad \text{and} \quad \mathcal{P}_{\trianglelefteq p}(\Box^{\leqslant h} \Phi) = \mathcal{P}_{\trianglelefteq p}(\Phi \mathcal{W}^{\leqslant h} \mathrm{ff}).$$

Note that $\mathrm{ff} = \neg \mathrm{tt}$. Some example formulae are $\mathcal{P}_{\leqslant 0.5}(a \mathcal{U} b)$ asserting that the probability of reaching a b-state via an a-path is at most $\frac{1}{2}$, and $\mathcal{P}_{>0.001}(\Diamond^{\leqslant 50} \mathit{error})$ stating that the probability for a system error to occur within 50 steps exceeds 0.001. Dually, $\mathcal{P}_{\leqslant 0.999}(\Box^{\leqslant 50} \neg \mathit{error})$ states that the probability for no error in the next 50 steps is at most 0.999.

Semantics. Let DTMC $\mathcal{D} = (S, \mathbf{P}, L)$. The semantics of PCTL is defined by a satisfaction relation, denoted $\models$, which is characterized as the least relation over the states in S (paths in $\mathcal{D}$, respectively) and the state formulae (path formulae) satisfying:

$$\begin{array}{lll} s \models \mathrm{tt} \quad \text{iff} \quad \mathit{true} & s \models a \quad \text{iff} \quad a \in L(s) & s \models \neg \Phi \quad \text{iff} \quad \text{not } (s \models \Phi) \\ s \models \Phi \wedge \Psi \quad \text{iff} \quad s \models \Phi \text{ and } s \models \Psi & s \models \mathcal{P}_{\trianglelefteq p}(\phi) \quad \text{iff} \quad \mathit{Prob}(s, \phi) \trianglelefteq p & \end{array}$$

Let $\mathit{Path}(s, \phi)$ denote the set of infinite paths that start in state s and satisfy ϕ. Formally, $\mathit{Path}(s, \phi) = \{\sigma \in \mathit{Path}(s) \mid \sigma \models \phi\}$. Here, $\mathit{Prob}(s, \phi) = \Pr\{\sigma \mid \sigma \in \mathit{Path}(s, \phi)\}$ denotes the probability of $\mathit{Path}(s, \phi)$. Let σ be an infinite path in $\mathcal{D}$. The semantics of PCTL path formulae is defined as:

$$\begin{array}{lll} \sigma \models \Phi \mathcal{U}^{\leqslant h} \Psi & \text{iff} & \exists i \leqslant h \text{ such that } \sigma[i] \models \Psi \text{ and } \forall j : 0 \leqslant j < i . (\sigma[j] \models \Phi). \\ \sigma \models \Phi \mathcal{W}^{\leqslant h} \Psi & \text{iff} & \text{either } \sigma \models \Phi \mathcal{U}^{\leqslant h} \Psi \text{ or } \sigma[i] \models \Phi \text{ for all } i \leqslant h. \end{array}$$

For finite path σ, $\models$ is defined in a similar way by changing the range of i to $i \leqslant \min\{h, |\sigma|\}$. Let $\mathit{Path}_{\mathit{fin}}(s, \phi)$ denote the set of finite paths starting in s that fulfill ϕ.

The until and weak until operators are closely related. This follows from the following equations. For any state s and all PCTL-formulae Φ and Ψ we have:

$$\begin{aligned} \mathcal{P}_{\geqslant p}(\Phi \mathcal{W}^{\leqslant h} \Psi) &\equiv \mathcal{P}_{\leqslant 1-p}((\Phi \wedge \neg \Psi) \mathcal{U}^{\leqslant h} (\neg \Phi \wedge \neg \Psi)) \\ \mathcal{P}_{\geqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi) &\equiv \mathcal{P}_{\leqslant 1-p}((\Phi \wedge \neg \Psi) \mathcal{W}^{\leqslant h} (\neg \Phi \wedge \neg \Psi)) \end{aligned}$$

For the rest of the paper, we explore counterexamples for PCTL formulae of the form $\mathcal{P}_{\leqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi)$. In Section 7, we will show how to generate counterexamples for formulae of the form $\mathcal{P}_{\geqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi)$.

3 Strongest evidences and counterexamples

Let us first consider what a counterexample in our setting actually is. To that end, consider the formula $\mathcal{P}_{\leqslant p}(\phi)$, where we denote $\phi = \Phi \mathcal{U}^{\leqslant h} \Psi$ ($h \in \{\infty\} \cup \mathbb{N}$) for the rest of the paper. It follows directly from the semantics that:

$$s \not\models \mathcal{P}_{\leqslant p}(\phi) \quad \text{iff} \quad \text{not } (\mathit{Prob}(s, \phi) \leqslant p) \quad \text{iff} \quad \Pr\{\sigma \mid \sigma \in \mathit{Path}(s, \phi)\} > p.$$

So, $\mathcal{P}_{\leqslant p}(\phi)$ is refuted by state s whenever the total probability mass of all ϕ-paths that start in s exceeds p. This indicates that a counterexample for $\mathcal{P}_{\leqslant p}(\phi)$ is in general a *set* of paths starting in s and satisfying ϕ. As ϕ is an until-formula whose validity (regardless of the value of h) can be witnessed by finite state sequences, *finite* paths do suffice in counterexamples. A counterexample is defined as follows:

Definition 2 (Counterexample). *A* counterexample *for $\mathcal{P}_{\leqslant p}(\phi)$ in state s is a set C of finite paths such that $C \subseteq \mathit{Path}_{\mathit{fin}}(s, \phi)$ and* $\Pr(C) > p$.

A counterexample for state s is thus a set of finite paths that all start in s. We will not dwell further upon how to represent this set, being it a finite tree (or dag) rooted at s, or a bounded regular expression (over states), and assume that an abstract representation as a set suffices. Note that the measurability of counterexamples is ensured by the fact that they just consist of finite paths; hence, $\Pr(C)$ is well-defined. Let $CX_p(s, \phi)$ denote the set of all counterexamples for $\mathcal{P}_{\leqslant p}(\phi)$ in state s. For $C \in CX_p(s, \phi)$ and C's superset C': $C \subseteq C' \subseteq \mathit{Path}_{\mathit{fin}}(s, \phi)$, it follows that $C' \in CX_p(s, \phi)$, since $\Pr(C') \geqslant \Pr(C) > p$. That is to say, any extension of a counterexample C with paths in $\mathit{Path}_{\mathit{fin}}(s, \phi)$ is a counterexample.

Definition 3 (Minimal counterexample). *$C \in CX_p(s, \phi)$ is a* minimal counterexample *if $|C| \leqslant |C'|$, for any $C' \in CX_p(s, \phi)$.*

Note that what we define as being minimal differs from minimality w.r.t. $\subseteq$. As a counterexample should exceed p, a maximally probable ϕ-path is a strong evidence for the violation of $\mathcal{P}_{\leqslant p}(\phi)$. For minimal counterexamples such maximally probable paths are essential.

Definition 4 (Strongest evidence). *A* strongest evidence *for violating $\mathcal{P}_{\leqslant p}(\phi)$ in state s is a finite path $\sigma \in \mathit{Path}_{\mathit{fin}}(s, \phi)$ such that* $\Pr\{\sigma\} \geqslant \Pr\{\sigma'\}$ *for any $\sigma' \in \mathit{Path}_{\mathit{fin}}(s, \phi)$.*

Dually, a strongest evidence for violating $\mathcal{P}_{\leqslant p}(\phi)$ is a strongest witness for fulfilling $\mathcal{P}_{>p}(\phi)$. Evidently, a strongest evidence does not need to be a counterexample as its probability mass may be (far) below p.

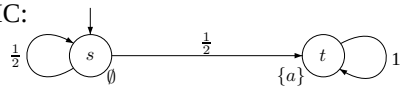
As in conventional model checking, we are not interested in generating arbitrary counterexamples, but those that are easy to comprehend, and provide a clear evidence of the refutation of the formula. So, akin to shortest counterexamples for linear-time logics, we consider the notion of a smallest, most indicative counterexample. Such counterexamples are required to be succinct, i.e., minimal, allowing easier analysis of the cause of refutation, and most distinctive, i.e., their probability should mostly exceed p among all minimal counterexamples.

Definition 5 (Smallest counterexample). $C \in CX_p(s, \phi)$ *is a* smallest (most indicative) counterexample *if it is minimal and* $\Pr(C) \geqslant \Pr(C')$ *for any minimal counterexample* $C' \in CX_p(s, \phi)$.

The intuition is that a smallest counterexample is mostly deviating from the required probability bound given that it has the smallest number of paths. Thus, there does not exist an equally sized counterexample that deviates more from p. Strongest evidences, minimal counterexamples or smallest counterexamples may not be unique, as paths may have equal probability. As a result, not every strongest evidence is contained in a minimal (or smallest) counterexample. Whereas minimal counterexamples may not contain any strongest evidence, any smallest counterexample contains at least one strongest evidence. Using some standard mathematical results we obtain:

Lemma 1. *A smallest counterexample for* $s \not\models \mathcal{P}_{\leqslant p}(\phi)$ *is finite.*

Remark 1 (Finiteness). For until path formulae, smallest counterexamples are always *finite* sets of paths if we consider *non-strict* upper-bounds on the probability, i.e., probability bounds of the form $\leqslant p$. In case of strict upper-bounds of the form $< p$, finiteness of counterexamples is no longer guaranteed as C for which $\Pr(C)$ equals p is a smallest counterexample, but may contain infinitely many paths. For instance, consider the following DTMC:



The violation of $\mathcal{P}_{<1}(\Diamond a)$ in state s can only be shown by an infinite set of paths, viz. all paths that traverse the self-loop at state s arbitrarily often.

Example 2. Consider the DTMC in Fig. 1, for which s violates $\mathcal{P}_{\leqslant \frac{1}{2}}(a \mathcal{U} b)$. Evidences are, amongst others, $\sigma_1 = s{\cdot}s_1{\cdot}t_1$, $\sigma_2 = s{\cdot}s_1{\cdot}s_2{\cdot}t_1$, $\sigma_3 = s{\cdot}s_2{\cdot}t_1$, $\sigma_4 = s{\cdot}s_1{\cdot}s_2{\cdot}t_2$, and $\sigma_5 = s{\cdot}s_2{\cdot}t_2$. Their respective probabilities are 0.2, 0.2, 0.15, 0.12 and 0.09. Paths σ_1 and σ_2 are strongest evidences. The set $C_1 = \{\sigma_1, \ldots, \sigma_5\}$ with $\Pr(C_1) = 0.76$ is a counterexample, but not a minimal one, as the removal from either σ_1 or σ_2 also yields a counterexample. $C_2 = \{\sigma_1, \sigma_2, \sigma_4\}$ is a minimal but not a smallest counterexample, as $C_3 = \{\sigma_1, \sigma_2, \sigma_3\}$ is minimal too with $\Pr(C_3) = 0.56 > 0.52 = \Pr(C_2)$. C_3 is a smallest counterexample.

In the remainder of the paper, we consider the strongest evidence problem (SE), that for a given state s with $s \not\models \mathcal{P}_{\leqslant p}(\phi)$, determines the strongest evidence for this violation. Subsequently, we consider the corresponding smallest counterexample problem (SC). For both cases, we distinguish between until-formulae for which $h{=}\infty$ (unbounded until) and $h \in \mathbb{N}$ (bounded until) as distinctive algorithms are used for these cases.

4 From a DTMC to a weighted digraph

Prior to finding strongest evidences or smallest counterexamples, we modify the DTMC and turn it into a weighted digraph. Let $\mathit{Sat}(\Phi) = \{s \in S \mid s \models \Phi\}$ for any Φ. Due to the bottom-up traversal of the model-checking algorithm over the formula $\phi = \Phi \mathcal{U}^{\leqslant h} \Psi$, we may assume that $\mathit{Sat}(\Phi)$ and $\mathit{Sat}(\Psi)$ are known.

Step 1: Adapting the DTMC. First, we make all states in the DTMC $\mathcal{D} = (S, \mathbf{P}, L)$ that neither satisfy Φ nor Ψ absorbing. Then we add an extra state t so that all outgoing transitions from a Ψ-state are replaced by a transition to t with probability 1. State t can thus only be reached via a Ψ-state. The obtained DTMC $\mathcal{D}' = (S', \mathbf{P}', L')$ has state space $S \cup \{t\}$ for $t \notin S$. The stochastic matrix $\mathbf{P}'$ is defined as follows:

$$\begin{array}{ll} \mathbf{P}'(s,s) = 1 \text{ and } \mathbf{P}'(s,s') = 0 \text{ for } s' \neq s & \text{if } s \notin \mathit{Sat}(\Phi) \cup \mathit{Sat}(\Psi) \text{ or } s = t \\ \mathbf{P}'(s,t) = 1 \text{ and } \mathbf{P}'(s,s') = 0 \text{ for } s' \neq t & \text{if } s \in \mathit{Sat}(\Psi) \\ \mathbf{P}'(s,s') = \mathbf{P}(s,s') \text{ for } s' \in S \text{ and } \mathbf{P}'(s,t) = 0 & \text{otherwise.} \end{array}$$

$L'(s) = L(s)$ for $s \in S$ and $L'(t) = \{at_t\}$, where $at_t \notin L(s')$ for any $s' \in S$, i.e., at_t uniquely identifies being at state t. Remark that all the $(\neg\Phi \wedge \neg\Psi)$-states could be collapsed into a single state, but this is not further explored here. The time complexity of this transformation is $\mathcal{O}(n)$ where $n = |S|$. It is evident that the validity of $\Phi\mathcal{U}^{\leqslant h}\Psi$ is not affected by this amendment of the DTMC. By construction, any finite path $\sigma{\cdot}t$ in $\mathcal{D}'$ satisfies $(\Phi \vee \Psi)\mathcal{U}^{\leqslant h+1} at_t$ and has the form $s_0{\cdot}...{\cdot}s_i{\cdot}s_{i+1}{\cdot}t$ where $s_j \models \Phi$ for $0 \leqslant j \leqslant i < h$, $s_{i+1} \models \Phi$; the prefix σ (in $\mathcal{D}$) satisfies $\Phi\mathcal{U}^{\leqslant h}\Psi$ where σ' and σ are equally probable.

Step 2: Conversion into a weighted digraph. As a second preprocessing step, the DTMC obtained in the first phase is transformed into a weighted digraph. Recall that a weighted digraph is a tuple $\mathcal{G} = (V, E, w)$ where V is a finite set of vertices, $E \subseteq V \times V$ is a set of edges, and $w : E \to \mathbb{R}_{\geqslant 0}$ is a weighted function.

Definition 6 (Weighted digraph of a DTMC). *For DTMC $\mathcal{D} = (S, \mathbf{P}, L)$, the weighted digraph $\mathcal{G}_D = (V, E, w)$ where:*

$$V = S \quad \textit{and} \quad (v, v') \in E \ \textit{iff} \ \mathbf{P}(v, v') > 0 \quad \textit{and} \quad w(v, v') = \log(\mathbf{P}(v, v')^{-1}).$$

Note that $w(s, s') \in [0, \infty)$ if $\mathbf{P}(s, s') > 0$. Thus, we indeed obtain a non-negatively weighted digraph. Note that this transformation can be done in $\mathcal{O}(m)$, where $m = |\mathbf{P}|$, i.e., the number of non-zero elements in $\mathbf{P}$.

A path σ from s to t in $\mathcal{G}$ is a sequence $\sigma = v_0{\cdot}v_1{\cdot}...{\cdot}v_j \in V^+$, where $v_0 = s, v_j = t$ and $(v_i, v_{i+1}) \in E$, for $0 \leqslant i < |\sigma|$. As for paths in DTMCs, $|\sigma|$ denotes the length of σ. The *distance* of finite path $\sigma = v_0{\cdot}v_1{\cdot}...{\cdot}v_j$ in graph $\mathcal{G}$ is $d(\sigma) = \sum_{i=0}^{j-1} w(v_i, v_{i+1})$. Due to the fact that multiplication of probabilities in $\mathcal{D}$ corresponds to addition of weights in $\mathcal{G}_D$, and that weights are based on taking the logarithm of the reciprocal of the transition probabilities in $\mathcal{D}$, distances in $\mathcal{G}$ and path-probabilities in DTMC $\mathcal{D}$ are related as follows:

Lemma 2. *Let σ and σ' be finite paths in DTMC $\mathcal{D}$ and its graph $\mathcal{G}_D$. Then:*

$$\Pr\{\sigma'\} \geqslant \Pr\{\sigma\} \quad \textit{iff} \quad d(\sigma') \leqslant d(\sigma).$$

The correspondence between path probabilities in the DTMC and distances in its weighted digraph as laid down in the following lemma, constitutes the basis for the remaining algorithms in this paper.

Lemma 3. *For any path σ from s to t in DTMC $\mathcal{D}$, $k > 0$, and $h \in \mathbb{N} \cup \{\infty\}$: σ is a k-th most probable path of at most h hops in $\mathcal{D}$ iff σ is a k-th shortest path of at most h hops in $\mathcal{G}_D$.*

5 Finding strongest evidences

Unbounded until. Based on the results of Lemma 3 where $k = 1$ and $h = \infty$, we consider the well-known shortest path problem. Recall that:

Definition 7 (SP problem). *Given a weighted digraph $\mathcal{G} = (V, E, w)$ and $s, t \in V$, the* shortest path *(SP) problem is to determine a path σ from s to t such that $d(\sigma) \leqslant d(\sigma')$ for any path σ' from s to t in $\mathcal{G}$.*

From Lemma 3 together with the transformation of a DTMC into a weighted digraph, it follows that there is a polynomial reduction from the SE problem for unbounded until to the SP problem. As the SP problem is known to be in PTIME, it follows:

Theorem 1. *The SE problem for unbounded until is in PTIME.*

Various efficient algorithms [14,9,12] exist for the SP problem, e.g., when using Dijkstra's algorithm, the SE problem for unbounded until can be solved in time $\mathcal{O}(m + n \log n)$ if appropriate data structures such as Fibonacci heaps are used.

Bounded until. Lemma 3 for $k = 1$ and $h \in \mathbb{N}$ suggests to consider the hop-constrained SP problem.

Definition 8 (HSP problem). *Given a weighted digraph $\mathcal{G} = (V, E, w)$, $s, t \in V$ and $h \in \mathbb{N}$, the* hop-constrained SP *(HSP) problem is to determine a path σ in $\mathcal{G}$ from s to t with $|\sigma| \leqslant h$ such that $d(\sigma) \leqslant d(\sigma')$ for any path σ' from s to t with $|\sigma'| \leqslant h$.*

The HSP problem is a special case of the constrained shortest path (CSP) problem [25,2], where the only constraint is the hop count.

Definition 9 (CSP problem). *Given a weighted digraph $\mathcal{G} = (V, E, w)$, $s, t \in V$ and resource constraints λ^i, for $1 \leqslant i \leqslant c$. Edge $e \in E$ uses $r^i(e) \geqslant 0$ units of resource i. The* (resource) constrained shortest path *problem (CSP) is to determine a shortest path σ in $\mathcal{G}$ from s to t such that $\sum_{e \in \sigma} r^i(e) \leqslant \lambda^i$ for $1 \leqslant i \leqslant c$.*

The CSP problem is NP-complete, even for a single resource constraint [2]. However, if each edge uses a constant unit of that resource (such as the hop count), the CSP problem can be solved in polynomial time, cf. [17], problem [ND30]. Thus:

Theorem 2. *The SE problem for bounded until is in PTIME.*

For $h \geqslant n-1$, it is possible to use Dijkstra's SP algorithm (as for unbounded until), as a shortest path does not contain cycles. If $h < n-1$, however, Dijkstra's algorithm does not guarantee to obtain a shortest path of at most h hops. We, therefore, adopt the Bellman-Ford (BF) algorithm [9,12] which fits well to our problem as it proceeds by increasing hop count. It can be readily modified to generate a shortest path within a given hop count. In the sequel of the paper, this algorithm is generalized for computing smallest counterexamples. The BF-algorithm is based on a set of recursive equations;

we extend these with the hop count h. For $v \in V$, let $\pi_h(s,v)$ denote the shortest path from s to v of at most h hops (if it exists). Then:

$$\pi_h(s,v) = \begin{cases} s & \text{if } v = s \text{ and } h \geqslant 0; \quad (1a) \\ \bot & \text{if } v \neq s \text{ and } h = 0; \quad (1b) \\ \arg\min_u \{d(\pi_{h-1}(s,u) \cdot v) \mid (u,v) \in E\} & \text{if } v \neq s \text{ and } h > 0. \quad (1c) \end{cases}$$

where $\bot$ denotes nonexistence of such a path. The last clause states that $\pi_h(s,v)$ consists of the shortest path to v's predecessor u, i.e., $\pi_{h-1}(s,u)$, extended with edge (u,v). Note that $\min_u\{d(\pi_{h-1}(s,u)\cdot v) \mid (u,v) \in E\}$ is the distance of the shortest path; by means of $\arg$, the path is obtained. It follows (cf. [22]) that equation $(1a)\sim(1c)$ characterizes the shortest path from s to v in at most h hops, and can be solved in time $\mathcal{O}(hm)$. As $h < n-1$, this is indeed in PTIME. Recall that for $h \geqslant n-1$, Dijkstra's algorithm has a favorable time complexity.

Exploiting the Viterbi algorithm. An alternative to using the BF algorithm is to adopt the *Viterbi algorithm* [16,27]. In fact, to apply this algorithm the transformation into a weighted digraph is not needed. The Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of hidden states (i.e., a finite path) that result in a sequence of observed events (a trace), especially in the context of hidden Markov models. Let $\mathcal{D}$ be a DTMC that is obtained after the first step described in Section 4, and suppose that $L(s)$ contains the set of atomic propositions that are valid in s and all subformulae of the formula under consideration. (Note that these labels are known due to the recursive descent nature of the PCTL model checking algorithm.) Let $tr(\sigma)$ denote the projection of a path $\sigma = s_0{\cdot}s_1{\cdot}\ldots{\cdot}s_h$ on its trace, i.e., $tr(\sigma) = L(s_0){\cdot}L(s_1){\cdot}\ldots{\cdot}L(s_h)$. $\sigma{\downarrow_i}$ denotes the prefix of path σ truncated at length i (thus ending in s_i), formally, $\sigma{\downarrow_i} = \sigma[0]{\cdot}\sigma[1]{\cdot}\ldots{\cdot}\sigma[i]$. Thus, $tr(\sigma{\downarrow_i}) = L(s_0){\cdot}L(s_1){\cdot}\ldots{\cdot}L(s_i)$. $\gamma{\downarrow_i}$ denotes the prefix of trace γ with length i. Let $\rho(\gamma,i,v)$ denote the probability of the most probable path $\sigma{\downarrow_i}$ whose trace equals $\gamma{\downarrow_i}$ and reaches state v. $\rho(\gamma,i,v)$ can be formally defined as follows:

$$\rho(\gamma,i,v) = \max_{tr(\sigma{\downarrow_i})=\gamma_i} \prod_{j=0}^{i-1} \mathbf{P}(s_j,s_{j+1}) \cdot \mathbf{1}_v(s_i),$$

where $\mathbf{1}_v(s_i)$ is the characteristic function of v, i.e., $\mathbf{1}_v(s_i)$ returns 1, if $s_i = v$, and 0 otherwise. The Viterbi algorithm provides an algorithmic solution to compute $\rho(\gamma,i,v)$:

$$\rho(\gamma,i,v) = \begin{cases} 1 & \text{if } s = v \text{ and } i = 0; \\ 0 & \text{if } s \neq v \text{ and } i = 0; \\ \max_{u\in S} \rho(\gamma,i-1,u) \cdot \mathbf{P}(u,v) & \text{otherwise.} \end{cases}$$

By computing $\rho(\Phi^h\Psi,h,s_h)$, the Viterbi algorithm determines the most probable h-hop path $\sigma = s_0{\cdot}s_1{\cdot}\ldots{\cdot}s_h$ that generates the trace $\gamma = L'(s_0)L'(s_1)\ldots L'(s_h) = \Phi^h\Psi$ with length $(h{+}1)$. Here, $L'(s) = L(s) \cap \{\Phi,\Psi\}$, i.e., L' is the labelling restricted to the subformulae Φ and Ψ. For our SE problem for bounded until, the trace of the most probable hop-constrained path from s to t is among $\{\Psi at_t, \Phi\Psi at_t, \ldots, \Phi^h\Psi at_t\}$. The

self-loop at vertex t with probability one ensures that all these paths have length $h{+}1$ while not changing their probabilities. For instance, the path with trace $\Phi^i \Psi\, at_t$ can be extended so that the trace becomes $\Phi^i \Psi\, at_t{}^{h+1-i}$, where $i \leqslant h$. Since the DTMC is already transformed as in Step 1, we can obtain the most probable path for $\Phi \mathcal{U}^{\leqslant h} \Psi$ by computing $\rho((\Phi \vee \Psi \vee at_t)^{h+1} at_t, h{+}1, t)$ using the Viterbi algorithm. The time complexity is $\mathcal{O}(hm)$, as for the BF algorithm.

6 Finding smallest counterexamples

Recall that a smallest (most indicative) counterexample is a minimal counterexample, whose probability—among all minimal counterexamples—deviates maximally from the required probability bound. In this section, we investigate algorithms and complexity bounds for computing such smallest counterexamples. First observe that any smallest counterexample that contains, say k paths, contains the k most probable paths. This follows from the fact that any non-k most probable path can be exchanged with a more probable path, without changing the size of the counterexample, but by increasing its probability.

Unbounded until. Lemma 3 is applicable here for $k > 1$ and $h = \infty$. This suggests to consider the k shortest paths problem.

Definition 10 (KSP problem). *Given a weighted digraph $\mathcal{G} = (V, E, w)$, $s, t \in V$, and $k \in \mathbb{N}$, the k* shortest paths *(KSP) problem is to find k distinct shortest paths between s and t in $\mathcal{G}$, if such paths exist.*

Theorem 3. *The SC problem for unbounded until is a KSP problem.*

Proof. We prove that a smallest counterexample of size k, contains k most probable paths. It is proven by contradiction. Let C be a smallest counterexample for ϕ with $|C| = k$, and assume C does not contain the k most probable paths satisfying ϕ. Then there is a path $\sigma \notin C$ satisfying ϕ such that $\Pr\{\sigma\} > \Pr\{\sigma'\}$ for some $\sigma' \in C$. Let $C' = C \setminus \{\sigma'\} \cup \{\sigma\}$. Then C' is a counterexample for ϕ, $|C| = |C'|$ and $\Pr(C) > \Pr(C')$. This contradicts C being a smallest counterexample. □

The question remains how to obtain k. Various algorithms for the KSP problem require k to be known a priori. This is inapplicable in our setting, as the number of paths in a smallest counterexample is implicitly provided by the probability bound in the PCTL-formula and is not known in advance. We therefore consider algorithms that allow to determine k on the fly, i.e., that can halt at any k and resume if necessary. A good candidate is Eppstein's algorithm [15]. Although this algorithm has the best known asymptotic time complexity, viz. $\mathcal{O}(m{+}n \log n{+}k)$, in practice the recursive enumeration algorithm (REA) by Jiménez and Marzal [20] prevails. This algorithm has a time complexity in $\mathcal{O}(m{+}kn \log \frac{m}{n})$ and is based on a generalization of the recursive equations for the BF-algorithm. Besides, it is readily adaptable to the case for bounded h, as we demonstrate below. Note that the time complexity of all known KSP algorithms depends on k, and as k may be exponential, their complexity is *pseudo-polynomial*.

Bounded until. Similar to the bounded until case for strongest evidences, we now consider the KSP problem where the path length is constrained, cf. Lemma 3 for $h \in \mathbb{N}$.

Definition 11 (HKSP problem). *Given a weighted digraph $\mathcal{G} = (V, E, w)$, $s, t \in V$ and $h, k \in \mathbb{N}$, the* hop-constrained *KSP (HKSP) problem is to determine k shortest paths each of length at most h between s and t.*

Similar to Theorem 3 we obtain:

Theorem 4. *The SC problem for bounded until is a HKSP problem.*

To our knowledge, algorithms for the HKSP problem do not exist. In order to solve the HKSP problem, we propose a new algorithm that is strongly based on Jiménez and Marzal's REA algorithm [20]. The advantage of adapting this algorithm is that k can be determined on the fly, an essential characteristic for our setting. The algorithm is a conservative extension of the REA algorithm.

For $v \in V$, let $\pi_h^k(s, v)$ denote the k-th shortest path from s to v of length at most h (if it exists). As before, we use $\perp$ to denote the non-existence of a path. We establish the following equations:

$$\pi_h^k(s,v) = \begin{cases} s & \text{if } k = 1, v = s \text{ and } h \geqslant 0 \quad (2a) \\ \perp & \text{if } (k > 1, v = s, h = 0) \text{ or } (v \neq s, h = 0) \quad (2b) \\ \arg\min_\sigma \{d(\sigma) \mid \sigma \in Q_h^k(s,v)\} & \text{otherwise} \quad (2c) \end{cases}$$

where $Q_h^k(s, v)$ is a set of candidate paths among which $\pi_h^k(s, v)$ is chosen. The candidate sets are defined by:

$$Q_h^k(s,v) = \begin{cases} \{\pi_{h-1}^1(s,u) \cdot v \mid (u,v) \in E\} \\ \qquad \text{if } k = 1, v \neq s \text{ or } k = 2, v = s \\ (Q_h^{k-1}(s,v) - \{\pi_{h-1}^{k'}(s,u) \cdot v\}) \cup \{\pi_{h-1}^{k'+1}(s,u) \cdot v\} \\ \qquad \text{if } k > 1 \text{ and } u, k' \text{ are the node and index,} \\ \qquad \text{such that } \pi_h^{k-1}(s,v) = \pi_{h-1}^{k'}(s,u) \cdot v \end{cases} \quad (3)$$

Path $\pi_{h-1}^{k'+1}(s,u) \cdot v = \perp$ occurs when $Q_{h-1}^{k'+1}(s,u) = \emptyset$. Note that $\perp \cdot v = \perp$ for any $v \in V$. $Q_h^k(s,v) = \emptyset$ if it only contains $\perp$.

If $k{=}1$, the shortest path to v's predecessor u is extended with the edge to v. In the latter clause, $\pi_{h-1}^{k'}(s,u)$ denotes the selected $(k{-}1)$-st shortest path from s to u, where u is the direct predecessor of v. Paths in $Q_h^k(s,v)$ for $k > 1$ are thus either candidate paths for $k{-}1$ where the selected path is eliminated (first summand) or the $(k'{+}1)$-st shortest path from s to u extended with edge (u,v) (second summand). Note that for the source state s, there is no need to define $Q_h^k(s,s)$ as $\pi_h^k(s,s)$ is defined by equations $(2a)$ and $(2b)$, which act as termination conditions. In a similar way as in [20] it can be proven that:

Lemma 4. *The equations $(2a)$-$(2c)$ and (3) characterize the hop-constrained k shortest paths from s to v in at most h hops.*

The adapted REA. The adapted REA for computing the k shortest paths from s to t which each consist of at most h hops is sketched as follows. The algorithm is based on the recursive equations given just above.

i Compute $\pi_h^1(s,t)$ by the BF algorithm and set $k := 1$.

ii Repeat until $\sum_{i=1}^{k} \Pr\{\pi_h^i(s,t)\} > p$:

(a) Set $k := k{+}1$ and compute $\pi_h^k(s,t)$ by invoking *NextPath*(v,h,k).

For $k{>}1$, and once $\pi_h^1(s,v),\ldots,\pi_h^{k-1}(s,v)$ are available, *NextPath*(t,h,k) computes $\pi_h^k(s,v)$ as follows:

1. If $h{\leqslant}0$, goto step 4.
2. If $k{=}2$, then set $Q[v,h] := \{\pi_{h-1}^1(s,u){\cdot}v \mid (u,v) \in E \text{ and } \pi_h^1(s,v) \neq \pi_{h-1}^1(s,u){\cdot}v\}$.
3. Let u and k' be the node and index such that $\pi_h^{k-1}(s,v) = \pi_{h-1}^{k'}(s,u){\cdot}v$.
   (a) If $\pi_{h-1}^{k'+1}(s,u)$ has not yet been computed, invoke *NextPath*$(u,h{-}1,k'{+}1)$.
   (b) If $\pi_{h-1}^{k'+1}(s,u)$ exists, then insert $\pi_{h-1}^{k'+1}(s,u){\cdot}v$ in $Q[v,h]$.
4. If $Q[v,h] \neq \emptyset$, then select and delete a path with minimum weight from $Q[v,h]$ and assign it to $\pi_h^k(s,v)$, else $\pi_h^k(s,v)$ does not exist.

In the main program, first the shortest path from s to t is determined using, e.g., the BF-algorithm. The intermediate results are recorded. Then, the k shortest paths are determined iteratively using the subroutine *NextPath*. The computation terminates when the total probability mass of the k shortest paths so far exceeds the bound p. Recall that p is the upper bound of the PCTL formula to be checked. Note that $Q[v,h]$ in the algorithm corresponds to $Q_h^k(s,v)$, where k is the parameter of the program. In steps 2 through 3, the set $Q_h^k(s,v)$ is determined from $Q_h^{k-1}(s,v)$ according to equation (3). In the final step, $\pi_h^k(s,v)$ is selected from $Q_h^k(s,v)$ according to equation $(2c)$.

To determine the computational complexity of the algorithm, we assume the candidate sets to be implemented by heaps (as in [20]). The k shortest paths to a vertex v can be stored in a linked list, where each path $\pi_h^k(s,v) = \pi_{h-1}^{k'}(s,u){\cdot}v$ is compactly represented by its length and a back pointer to $\pi_{h-1}^{k'}(s,u)$. Using these data structures, we obtain:

Theorem 5. *The time complexity of the adapted REA is* $\mathcal{O}(hm + hk\log(\frac{m}{n}))$.

Note that the time complexity is pseudo-polynomial due to the dependence on k which may be exponential in n. As in our setting, k is not known in advance, this can not be reduced to a polynomial time complexity.

7 Lower bounds on probabilities

For the violation of PCTL formulae with lower bounds, i.e., $s \not\models \mathcal{P}_{\geqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi)$, the formula and model will be changed so that the algorithms for finding strongest evidences and smallest counterexamples for PCTL can be applied.

Unbounded until. For $h = \infty$, we have:

$$\mathcal{P}_{\geqslant p}\big(\Phi \mathcal{U}\, \Psi\big) \equiv \mathcal{P}_{\leqslant 1-p}\big(\underbrace{(\Phi \wedge \neg\Psi)}_{\Phi^*}\, \mathcal{W}\, \underbrace{(\neg\Phi \wedge \neg\Psi)}_{\Psi^*}\big) \equiv \mathcal{P}_{\leqslant 1-p}\big(\underbrace{(\Phi \wedge \neg\Psi)}_{\Phi^*}\, \mathcal{U}(at_u \vee at_b)\big),$$

where at_u and at_b are two new atomic propositions such that (i) $s \models at_u$ iff $s \models \Psi^*$ (ii) $s \models at_b$ iff $s \in B$ where B is a bottom strongly connected component (BSCC) such that $B \subseteq Sat(\Phi^*)$, or shortly $s \in B_{\Phi^*}$. A BSCC B is a maximal strong component that has no transitions that leave B.

Algorithmically, the DTMC is first transformed such that all the $(\neg\Phi^* \wedge \neg\Psi^*)$-states are made absorbing. Note that once those states are reached, $\Phi^* \mathcal{W} \Psi^*$ will never be satisfied. As a second step, all the Ψ^*-states are labelled with at_u and made absorbing. Finally, all BSCCs are obtained and all states in B_{Φ^*} are labelled with at_b. The obtained DTMC now acts as the starting point for applying all the model transformations and algorithms in Section 4-6 to generate a counterexample for $\mathcal{P}_{\leqslant 1-p}\big(\Phi^* \mathcal{U}(at_u \vee at_b)\big)$.

Bounded until. For $h \in \mathbb{N}$, identifying all states in BSCC B_{Φ^*} is not sufficient, as a path satisfying $\Box^{\leqslant h}\Phi^*$ may never reach such BSCC. Instead, we transform the DTMC and use:

$$\mathcal{P}_{\geqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi) \equiv \mathcal{P}_{\leqslant 1-p}(\underbrace{(\Phi \wedge \neg\Psi)}_{\Phi^*}\, \mathcal{U}^{=h}(at_u \vee at_h)),$$

where at_u and at_h are new atomic propositions such that at_u is labelled as before and $s' \models at_h$ iff there exists $\sigma \in Path_{fin}(s)$ such that $\sigma[h] = s'$ and $\sigma \models \Box^{\leqslant h}\Phi^*$.

Algorithmically, the $(\neg\Phi^* \wedge \neg\Psi^*)$-states and Ψ^*-states are made absorbing; besides, all Ψ^*-states are labelled with at_u. As a second step, all the Φ^*-states that can be reached in exactly h hops are computed by e.g., a breadth first search (BFS) algorithm. The obtained DTMC now acts as the starting point for applying all the model transformations and algorithms in Section 4-6 to generate a counterexample for $\mathcal{P}_{\leqslant 1-p}\big(\Phi^* \mathcal{U}^{=h}(at_u \vee at_h)\big)$. Finite paths of exactly h paths suffice to check the validity of $\sigma \models \Box^{\leqslant h}\Phi^*$, thus $\Phi^* \mathcal{U}^{=h} at_h$ (not $\Phi^* \mathcal{U}^{\leqslant h} at_h$) is needed; besides the validity is unaffected if we change $\Phi \mathcal{U}^{\leqslant h} at_u$ into $\Phi \mathcal{U}^{=h} at_u$, since all at_u states are absorbing. Note that it is very easy to adapt the strongest evidences and smallest counterexamples algorithms for $\mathcal{U}^{\leqslant h}$ to those for $\mathcal{U}^{=h}$ – only the termination conditions need a slight change. The time complexity remains the same.

In the above explained way, counterexamples for (bounded) until-formulae with a lower bound on their probability are obtained by considering formulae on slightly adapted DTMCs with upper bounds on probabilities. Intuitively, the fact that s refutes $\mathcal{P}_{\geqslant p}(\Phi \mathcal{U}^{\leqslant h} \Psi)$ is witnessed by showing that violating paths of s are too probable, i.e., carry more probability mass than p. Alternatively, *all* paths starting in s that satisfy $\Phi \mathcal{U}^{\leqslant h} \Psi$ could be determined as this set of paths has a probability less than p.

8 Conclusion

Summary of results. We have investigated the computation of strongest evidences (maximally probable paths) and smallest counterexamples for PCTL model checking of

DTMCs. Relationships to various kinds of shortest path problems have been established. Besides, it is shown that for the hop-constrained strongest evidence problem, the Viterbi algorithm can be applied. Summarizing we have obtained the following connections and complexities:

counterexample problem	shortest path problem	algorithm	time complexity
SE (until)	SP	Dijkstra	$\mathcal{O}(m + n \log n)$
SE (bounded until)	HSP	BF/Viterbi	$\mathcal{O}(hm)$
SC (until)	KSP	Eppstein	$\mathcal{O}(m + n \log n + k)$
SC (bounded until)	HKSP	adapted REA	$\mathcal{O}(hm + hk \log(\frac{m}{n}))$

where n and m are the number of states and transitions, h is the hop bound, and k is the number of shortest paths.

Extensions. The results reported in this paper can be extended to (weak) until-formulae with minimal or interval bounds on the number of allowed steps. For instance, strongest evidences for $s \not\models \mathcal{P}_{\leqslant p}(\Phi \mathcal{U}^{[h,h']} \Psi)$ with $0 < h \leqslant h'$ can be obtained by appropriately combining maximally probable paths from s to states at distance h from s, and from those states to Ψ-states. Similar reasoning applies to the SC problem. For DTMCs with rewards, it can be established that the SE problem for violating reward- and hop-bounded until-formulae boils down to solving a non-trivial instance of the CSP problem. As this problem is NP-complete, efficient algorithms for finding counterexamples for PRCTL [5], a reward extension to PCTL, will be hard to obtain.

Further research. Topics for further research are: succinct representation and visualization of counterexamples, experimental research of the proposed algorithms in probabilistic model checking and considering loopless paths (see e.g., [23]).

Related work. The SE problem for timed reachability in CTMCs is considered in [3]. Whereas we consider the generation of strongest evidences once a property violation has been established, [3] assumes the CTMC to be unknown. The SE problem for CTMCs is mapped onto an SE problem on (uniformised) DTMCs, and heuristic search algorithms (Z*) are employed to determine the evidences. The approach is restricted to bounded until and due to the use of heuristics, time complexities are hard to obtain. In our view, the main advantage of our approach is the systematic characterization of generating counterexamples in terms of shortest path problems. Recently, [4] generalizes the heuristic approach to obtain failure subgraphs, i.e., counterexamples. To our knowledge, smallest counterexamples have not been considered yet.

Acknowledgement. Christel Baier and David N. Jansen are kindly acknowledged for their useful remarks on the paper. This research has been financially supported by the NWO project QUPES and by 973 and 863 Program of China (2002CB3120022005AA113160, 2004AA112090, 2005AA113030) and NSFC (60233010, 60273034, 60403014).

References

1. A.V. Aho, J.E. Hopcroft and J.D. Ullmann. The design and analysis of computer algorithms. Addison-Wesley, 1974.
2. R.K. Ahuja, T.L. Magnanti and J.B. Orlin. *Network Flows: Theory, Algorithms and Applications*, Prentice Hall, Inc., 1993.
3. H. Aljazzar, H. Hermanns and S. Leue. Counterexamples for timed probabilistic reachability. FORMATS 2005, LNCS 3829: 177-195, 2005.
4. H. Aljazzar and S. Leue. Extended directed search for probabilistic timed reachability. FORMATS 2006, LNCS 4202: 33-51, 2006.
5. S. Andova, H. Hermanns and J.-P. Katoen. Discrete-time rewards model-checked. FORMATS 2003, LNCS 2791: 88-104, 2003.
6. C. Baier, J.-P. Katoen, H. Hermanns and V. Wolf. Comparative branching-time semantics for Markov chains. *Inf. Comput.* 200(2): 149-214 (2005).
7. T. Ball, M. Naik and S. K. Rajamani. From symptom to cause: localizing errors in counterexample traces. POPL: 97-105, 2003.
8. G. Behrmann, K. G. Larsen and J. I. Rasmussen. Optimal scheduling using priced timed automata. *ACM SIGMETRICS Perf. Ev. Review* 32(4): 34-40 (2005).
9. R. Bellman. On a routing problem. *Quarterly of Appl. Math.*, 16(1): 87-90 (1958).
10. E.M. Clarke, O. Grumberg, S. Jha, Y. Lu and H. Veith: Counterexample-guided abstraction refinement. CAV, LNCS 1855: 154-169, 2000.
11. E.M. Clarke, S. Jha, Y. Lu and H. Veith. Tree-like counterexamples in model checking. LICS: 19-29 (2002).
12. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein. *Introduction to Algorithms*, 2001. Section 24.1: The Bellman-Ford algorithm, pp.588-592.
13. L. de Alfaro, T.A. Henzinger and F. Mang. Detecting errors before reaching them. CAV, LNCS 2725: 186-201, 2000.
14. E.W. Dijkstra. A note on two problems in connection with graphs. *Num. Math.*, 1:395-412 (1959).
15. D. Eppstein. Finding the k shortest paths. *SIAM J. Comput.* 28(2): 652-673 (1998).
16. G.D. Forney. The Viterbi algorithm. *Proc. of the IEEE* 61(3): 268-278 (1973).
17. M.R. Garey and D.S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.
18. A. Gurfinkel and M. Chechik. Proof-like counter-examples. TACAS, LNCS 2619: 160-175, 2003.
19. H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Asp. Comput.* 6(5): 512-535 (1994).
20. V.M. Jiménez and A. Marzal. Computing the K shortest paths: A new algorithm and an experimental comparison. WAE 1999, LNCS 1668: 15-29, 1999.
21. H. Jin, K. Ravi and F. Somenzi. Fate and free will in error traces. *STTT* 6(2): 102-116 (2004).
22. E.L. Lawler. *Combinatorial Optimization: Networks and Matroids.* Holt, Reinhart, and Winston, 1976.
23. E.Q.V. Martins and M.M.B. Pascoal. A new implementation of Yen's ranking loopless paths algorithm. *4OR* 1(2): 121-133 (2003).
24. E.Q.V. Martins, M.M.B. Pascoal and J.L.E. Dos Santos. Deviation algorithms for ranking shortest paths. *Int. J. Found. Comput. Sci.* 10(3): 247-262 (1999).
25. K. Mehlhorn and M. Ziegelmann. Resource constrained shortest paths. ESA 2000, LNCS 1879: 326-337, 2000.
26. S. Shoham and O. Grumberg. A game-based framework for CTL counterexamples and 3-valued abstraction-refinement. CAV, LNCS 2725: 275-287, 2003.
27. A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. on Inf. Theory* 13(2):260-269, 1967.