On the Hardness of PCTL Satisfiability

Souymodip Chakraborty, Joost-Pieter Katoen *
RWTH Aachen University, 52056 Aachen, Germany

Abstract. This paper shows that the satisfiability problem for probabilistic CTL (PCTL, for short) is undecidable. By a reduction from 1½-player games with PCTL winning objectives, we establish that the PCTL satisfiability problem is \(\Sigma_1^1\)-hard. We present an exponential-time algorithm for the satisfiability of a bounded, negation-closed fragment of PCTL, and show that the satisfiability problem for this fragment is \(\text{EXPTIME}\)-hard.

1 Introduction

Probabilistic CTL (PCTL) [10] is a probabilistic extension of the well-known branching-time logic [8] for specifying properties of stochastic systems. In PCTL, the existential and universal path quantifiers of CTL are replaced with the probabilistic operator, which allows to quantify the probability of all runs that satisfy a given path formula. The syntax of PCTL is built upon atomic propositions, using Boolean connectives and operators \(\text{next}\) and \(\text{until}\) of the form \([Xf]\) and \([f U g]\), respectively, where \(\bowtie \in \{\leq, <, \geq, >\}\), and \(p \in [0, 1]\) is a rational constant. Other operators such as \(\text{F}, \text{G}\) and \(\text{W}\) can be derived from \(\text{U}\).

The model checking problem for PCTL formulas over Markov chains has been widely studied and it is known to be solvable in polynomial time. By contrast, satisfiability procedures for PCTL is unknown. It has been shown that the satisfiability problem for the qualitative fragment of PCTL (i.e., \(p = 0\) or \(p = 1\)) is \(\text{EXPTIME}\)-complete [11,5]. In this paper we show that satisfiability problem of PCTL is \(\Sigma_1^1\)-hard (in the analytic hierarchy). The \(\Sigma_1^1\)-hardness is shown by reducing any 1½-player game with PCTL winning objective to a satisfiability query. The result then follows from [4], which shows that the existence problem of a winning strategy in a 1½-player game, where the winning criterion is defined by a PCTL formula, is \(\Sigma_1^1\)-hard.

It is well known that PCTL does not have the finite model property, even for the qualitative setting. For example, consider the PCTL formula \([G(\neg a \wedge [Xa]_{>0})]_{>0}\). This formula has no finite model, yet it is satisfiable. We identify a sub-logic called PCTL\(_{X, U}\) which has a finite model property and provide an \(\text{NEXPTIME}\) algorithm to decide its satisfiability problem. PCTL\(_{X, U}\)-contains the next and bounded until operator. In that respect, existence of finite model property is rather obvious. In this paper we investigate various fragments of

* This research is funded by the Excellence Initiative of the German Research Council DFG, and the EU FP7 project SENSATION.
PCTL\_X,\_U and give a hierarchical complexity analysis. This leads to the second contribution of the paper: the satisfiability problem of PCTL\_X,\_U without bounded until is PSPACE-complete. We then show that the full PCTL\_X,\_U (with bounded until) is EXPTIME-hard in the encoding of the problem and finally give an NEXPTIME algorithm in the size of the formula. It is important to note that, bounded until have a natural number to define the bound. Thus the size of the encoding (number of tape cells to store the input) is different from the size of the formula which depends on the value of the bound.

As we will see, the satisfiability of PCTL\_X,\_U ultimately leads to the feasibility of non-linear equations in the real closed field. The general problem is PSPACE-complete, as shown in [6]. We have developed a new variable elimination procedure which is custom-designed to solve the special kind of non-linear equations arising from the satisfiability problem. This could be of independent interest since the satisfiability of these type non-linear equation system is in NP.

The bounded satisfiability problem [1] studies the question whether a given formula has a simple model (where transition probabilities are 0, \( \frac{1}{2} \), or 1) of a specified size. The problem is reduced to SMT queries and solved in NP-time in the size of the model. Interestingly, the satisfiability problem of the PCTL\_X,\_U as defined here, is in NEXPTIME, and models of PCTL\_X,\_U are exponential in the size of the formula.

The rest of the paper is organised as follows. Section 2 introduces some important definitions and preliminaries. Section 3 describes the reduction technique to show \( \Sigma^1_1 \)-hardness. Section 4 introduces the sub-logic PCTL\_X,\_U defines it properties, describes algorithms for the satisfiability problem and mentions hardness results. In Section 5 we summarize our contribution and discuss an open problem. We have moved the hardness proofs and the variable elimination can be found in arxiv.

### 2 Preliminaries

Let \( Y \) be the set of functions from the set \( Y \) to the set \( X \). For \( \varphi \in X^Y \) let \( \text{img}(\varphi) \subseteq X \) be the image and \( \text{dom}(\varphi) = Y \) be the domain of \( \varphi \). The set of probability distributions over set \( X \) is denoted by \( D_X \) where \( d \in D_X \) iff \( d \in \mathbb{R}^X_+ \) and \( d^T \cdot 1 = 1 \) (\( \mathbb{R}_+ \) is the set of non-negative reals). For \( \mu \in D_X \), let \( \text{supp}(\mu) = \{ x \in X \mid \mu(x) > 0 \} \) be the support of distribution \( \mu \).

**Definition 1 (Markov chain).** A Markov chain (MC) \( M \) is a quintuple \( (S, P, AP, L, s_{in}) \) where \( S \) is a (countable) set of states, \( P(s) \in D_S \) for all \( s \in S \), \( AP \) is a set of atomic propositions, \( L : S \rightarrow 2^{AP} \) is a labeling function, and \( s_{in} \in S \) is the initial state.

An infinite path \( \sigma \) through MC \( M \) is a sequence of states \( \sigma = \{ \sigma_i \}_{i \geq 0} \), where for all \( i \geq 0 \), \( P(\sigma_i, \sigma_{i+1}) > 0 \). Let \( \text{path}(s) \) denote the set of (finite or infinite) paths starting from state \( s \). For a path \( \sigma \), let \( \sigma \downarrow \) denote the last state of \( \sigma \) if this exists (i.e., if \( \sigma \) is finite) and \( |\sigma| \) denote the length of \( \sigma \). Let \( \text{succ}(s) = \{ t \mid P(s, t) > 0 \} \) be the successors of state \( s \). A probability measure on sets of infinite paths is...
obtained in a standard way. Let \((\Omega_s, \mathcal{F}, \Pr)\) be the Borel \(\sigma\)-algebra where \(\Omega_s\) is the set of infinite paths from state \(s\), \(\mathcal{F}\) is the smallest \(\sigma\)-field on cylinder sets of \(\Omega_s\), and \(\Pr\) is the probability measure on \(\mathcal{F}\), for a finite path \(\sigma\), \(\Pr(\sigma) = \prod_{0 \leq i < |\sigma|} P(\sigma_{i-1}, \sigma_i)\) [2].

**Definition 3 (Markov decision process).** A finite path of an MDP is a sequence of states \(s\) and infinite paths from the state \(0\).

**Definition 2 (Probabilistic CTL [10]).** Formulas in probabilistic CTL (PCTL, for short) adhere to the following syntax:

\[
f \equiv a | -f | f \land f | [g]_{wp} \quad \text{where} \quad g \equiv Xf | f U f | f W f.
\]

Here \(a \in AP\), \(f\) is a state formula, \(g\) is a path formula, \(\land\) is a binary comparison operator in \(\{<, \leq, >, \geq, =\}\) and \(p\) is a rational number in \([0,1]\). As usual, \(Ff = \text{true} U f\), \(-Gf = F\neg f\) and \(f W g = f \land g \lor Gf\). Every PCTL-formula can be turned into positive normal form where negations only appear adjacent to atomic propositions. The PCTL semantics is defined on MCs. For a MC \(M\), state \(s \in S\), the satisfaction relation of state formulas is defined by:

\[
\begin{align*}
M, s &\models a \quad \text{iff} \quad a \in L(s) \\
M, s &\not\models -f \quad \text{iff} \quad M, s \not\models f \\
M, s &\models f_1 \land f_2 \quad \text{iff} \quad M, s \models f_1 \quad \text{and} \quad M, s \models f_2 \\
M, s &\models [g]_{wp} \quad \text{iff} \quad \Pr\{w \in \Omega_s \mid M, w \models g\} \geq p.
\end{align*}
\]

For infinite path \(w\), the satisfaction relation for path formulas is defined by:

\[
\begin{align*}
M, w &\models Xf \quad \text{iff} \quad M, w_1 \models f \\
M, w &\models f_1 \lor f_2 \quad \text{iff} \quad \exists i : M, w_i \models f_2 \quad \text{and} \quad \forall j < i : M, w_j \models f_1
\end{align*}
\]

Let \([f] = \{ M = (S, P, AP, L, s_{in}) \mid M, s_{in} \models f \}\).

**Definition 3 (Markov decision process).** A Markov decision process (MDP) \(D\) is a quintuple \((S, \Delta, AP, L, s_{in})\) where \(S\), \(AP\), \(L\), and \(s_{in}\) are as before, and \(\Delta : S \to 2^{P^S}\) such that \(\Delta(s)\) is a finite set of distributions. We assume \(S\) and \(\Delta(s)\) for each \(s \in S\) to be finite (unless the contrary is explicitly specified).

A finite path of an MDP is a sequence of states \(\sigma = \sigma_0 \ldots \sigma_n\) such for each \(0 < i \leq n\), \(\sigma_i \in \text{supp}(\mu)\) for some \(\mu \in \Delta(\sigma_{i-1})\). Let \(\text{path}(s)\) be the set of (finite and infinite) paths from the state \(s\). Let \(\text{succ}(s) = \{ t \mid t \in \bigcup_{\mu \in \Delta(s)} \text{supp}(\mu) \}\) be the set of successors of \(s\). As usual, we use **schedulers** to resolve the possible non-determinism in a state.

**Definition 4 (Scheduler).** A scheduler of MDP \(D = (S, \Delta, AP, L, s_{in})\) is a function \(\eta : S^+ \to D_S\) with \(\eta(\sigma) \in \Delta(\sigma_1)\). The scheduler \(\eta\) induces the MC \(D_\eta = (S^+, P, AP, L', s_{in})\) with \(P(\sigma, \sigma\cdot t) = \eta(\sigma)(t)\), and \(L'(\sigma) = L(\sigma_1)\).

These schedulers are history-dependent and deterministic. Let \(\text{HD}(D)\) denote the set of history-dependent deterministic schedulers of MDP \(D\).

**Definition 5 (Probabilistic bisimulation [14]).** Let MC \(M = (S, P, AP, L, s_{in})\) and \(H \subseteq AP\). The equivalence relation \(R_H \subseteq S \times S\) is a probabilistic bisimulation iff for every \((s, s') \in R_H\) it holds:

\[
M, s \models \varphi \quad \text{iff} \quad M, s' \models \varphi
\]
1. \(L(s) \cap H = L(s') \cap H\), and
2. for every \(C \in S/R_H\), we have \(\sum_{i \in C} P(s, t) = \sum_{i' \in C} P(s', t')\).

Let \(\approx_H\) denote the largest probabilistic bisimulation on \(S\). The MCs \(M_1\) and \(M_2\) are probabilistically bisimilar, denoted \(M_1 \approx_H M_2\), if \(s^1_m, s^2_m \in \text{the disjoint union of } M_1 \text{ and } M_2\).

3 MDP to PCTL

For MDP \(D = (S, \Delta, AP, L, s_{in})\), let \([D] = \{M \mid \exists \eta \in \text{HD}(D) : M \approx_{AP} D\}\) the set of MCs that are bisimilar to some MC obtained by a history-dependent deterministic scheduler on \(D\). For MDP \(D\), our aim is to construct a PCTL-formula \(f_D\) such that: \([D] = \llbracket f_D \rrbracket\).

**Labelling-insensitive partitioning.** Before we consider obtaining the formula \(f_D\), we prove a useful property for MCs. Let \(M\) be the MC \((S, P, AP, L, s_{in})\) where AP is finite, but \(S\) can be countably infinite. Let \(\pi = \{S_1, \ldots, S_n\}\) be a finite partitioning of \(S\) such that \(L(s) = L(s')\), for \(s, s' \in S_i (1 \leq i \leq n)\). As AP is finite, such partitioning always exists. This partitioning is said to be labelling insensitive as all states in a block of the partition are equally labelled. Let \(L(S_i) = \{a \mid \exists s \in S_i : a \in L(s)\}\). For labelling-insensitive partitioning \(\pi\), let \(M_\pi\) be the MC \((S, P, AP_\pi, L_\pi, s_{in})\) where \(AP_\pi = \{b_1, \ldots, b_n\}\) with \(b_i \notin AP\), and \(L_\pi(s) = \{b_i\}\) iff \(s \in S_i\). Observe that \(M_\pi\) and \(M\) differ only in their labelling. Now consider the PCTL-formula \(f\) (in positive normal form, i.e. negations only occur adjacent to propositions). Let \(f_\pi\) be the PCTL-formula where for \(a \in AP\), each occurrence of \(a\) in \(f\) is replaced by \(\varphi_a\) and each occurrence of \(\neg a\) in \(f\) is replaced by \(\varphi_{\neg a}\), where:

\[
\varphi_a = \bigvee_{S_i : a \in L(S_i)} (b_i \land \bigwedge_{S_i \neq S_i} \neg b_i) \quad \text{and} \quad \varphi_{\neg a} = \bigvee_{S_i : a \notin L(S_i)} (b_i \land \bigwedge_{S_i \neq S_i} \neg b_i)
\]

Note that the conjunction \(\bigwedge_{S_i \neq S_i} \neg b_i\) is superfluous in this setting, but becomes relevant later when we are considering possible models for \(f_\pi\).

**Proposition 1.** For every labelling-insensitive partitioning \(\pi\) of MC \(M\) and PCTL-formula \(f\) in positive normal form, \(M, s \models f\) iff \(M_\pi, s \models f_\pi\).

**Proof.** The proof is by induction on the structure of the PCTL-formula \(f\):

1. \(f = a\) with \(a \in AP\): Let \(M, s \models a\). Then \(a \in L(s)\). Assume \(s \in S_i\). Then, \(L_\pi(s) = \{b_i\}\) and \(a \in L(S_i)\). We deduce \(M_\pi, s \models \varphi_a\), since \((b_i \land \neg a \land b_i)\) is true at \(s\) in \(M_\pi\). Similarly, if \(M_\pi, s \models \varphi_{\neg a}\) and \(s \in S_i\), then \(a \in L(S_i)\), and since \(\pi\) is labelling insensitive, so \(a \in L(s)\). Hence, \(M, s \models a\).
2. The same argument holds for \(f = \neg a\).
3. \(f = g \land h\) (or \(f = g \lor h\)): Follows directly from the induction hypothesis.
4. \(f = [Xg]_{\pi}\): By induction hypothesis, we have \(\{t \mid M, t \models g\} = \{t \mid M_\pi, t \models g_\pi\}\). Let \(H\) denote this set. Thus, for any \(s\) the quantity \(\sum_{t \in H} P(s, t)\) is independent from \(M\) or \(M_\pi\) and thus, \(M, s \models [Xg]_{\pi}\) iff \(M_\pi, s \models [Xg_\pi]_{\pi}\).
5. \( f = [f^1 \cup f^2]_{\text{mp}} \): Let \( M, s = f \). Assume \( \sigma = f^1 \cup f^2 \) where \( \sigma \) starts in \( s \). Thus, \( M, \sigma_i = f^1 \) for some index \( i \), and for all \( j < i \), \( M, \sigma_j = f^1 \). By the induction hypothesis it follows that \( M, \sigma_i = f^2 \) and for all \( j < i \), \( M, \sigma_j = f^2 \). Thus, \( M, \sigma = f^1 \cup f^2 \). Generalizing this argument yields that every path \( \sigma \) satisfying \( f^1 \cup f^2 \) in \( M \), also satisfies \( f^1 \cup f^2 \) in \( M, \sigma \), and vice-versa. Thus, we have \( M, s = [f^1 \cup f^2]_{\text{mp}} \) iff \( M, s = [f^1 \cup f^2]_{\text{mp}} \).

6. \( f = [f^1 \cap f^2]_{\text{mp}} \): Let \( M, \sigma = f^1 \cap f^2 \). Then either for all \( i \), \( M, \sigma_i = f^1 \), in which case, we conclude from the induction hypothesis \( M, \sigma_i = f^2 \) for every \( i \), or \( M, \sigma = f^1 \cap f^2 \) which is the same as the previous case.

**PCTL-formulas for finite MDPs.** Let MDP \( D = (S, \Delta, AP, L, s_{\text{in}}) \) with \( S \) finite. As \( S \) is finite, we can assume that each state \( s \) of \( D \) has a unique label \( b_s \). Let \( \psi_s = b_s \land \land_{s' \sim b_s} \) for each state \( s \).

**Definition 6 (Characteristic PCTL-formula for MDP \( D \)).** The characteristic PCTL-formula \( f_D \) for MDP \( D \) with uniquely labelled states is defined by:

\[
f_D = \psi_{s_{\text{in}}} \land \left[ G(f_0 \land \land_{s \in S} f_s) \right]_{s=1} \text{ where } f_s = \psi_s \land \land_{\mu \in \Delta(s)} \left[ X\psi_{s'} \right]_{\mu(s')} \land_{s' \in \text{supp}(\mu)}
\]

and \( f_0 = \land_{s \in S} \psi_s \).

The following result asserts that \( f_D \) characterizes the set of MCs (up to probabilistic bisimilarity \( \approx \)) that are obtained from MDP \( D \) under a history-dependent deterministic scheduler.

**Theorem 1.** For any finite MDP \( D \) with uniquely labeled states, we have: \( [f_D] = [D] \).

**Proof.** Let \( D = (S, \Delta, L, AP, s_{\text{in}}) \) be a finite MDP with uniquely labeled states. W.l.o.g., let \( AP = \{ b_s \mid s \in S \} \) and \( L(s) = \{ b_s \} \) for every \( s \in S \). We first show that \([D] \subseteq [f_D] \). It suffices to show that for every \( \eta \in \text{HD}(D) \), \( D_\eta \approx f_D \). This is immediate, as each state \( \rho \) of the MC \( D_\eta \) satisfies \( \land_{s \in S} \psi_s \) and the scheduler \( \eta \) chooses exactly one distribution \( \mu \) from \( \Delta(\rho) \). It thus suffices to prove \([f_D] \subseteq [D] \). Let \( M_{f_D} = (T, \mathcal{P}, \mathcal{A}, L, t_{in}) \) be an MC with \( M_{f_D}, t_{in} = f_D \). We will construct a scheduler \( \eta \in \text{HD}(D) \) such that \( M_{f_D} \approx_{\text{AP}} D_\eta \).

Observe that \( M_{f_D} \) is bisimilar to its unfolding \( (T, \mathcal{P}, \mathcal{A}, L, t_{in}) \), where the set of states \( T \subseteq \{ \sigma \mid \sigma \in \text{path}(t_{in}) \} \), the transition probability and labelling functions are extended to \( T \) as \( P(\sigma, \sigma \downarrow t) = P(\sigma \downarrow t) \) and \( L(\sigma) = L(\sigma \downarrow t) \), respectively, where \( \sigma \downarrow t \) is the last state of \( \sigma \). Henceforth, we only consider the unfolding of \( M_{f_D} \), and for the sake of brevity, let \( M_{f_D} \) denote this unfolding. The proof is now done in several steps. We show that:

1. there is a mapping between states in \( M_{f_D} \) and state sequences of \( D \);
2. \( \eta \in \text{HD}(D) \) for some function \( \eta \) on MDP \( D \) defined using this mapping;
3. \( D_\eta \) and \( M_{f_D} \) are probabilistically bisimilar.

This yields \([f_D] \subseteq [D] \). The proofs of each of these steps are provided below.
1. Every state $\sigma$ of $M_{fD}$ satisfies $f_0$. Hence, by construction of $f_0$, there exists exactly one state $s \in S$ (of the MDP $D$) such that $M_{fD}, \sigma \models \psi_s$. This implies that the label of $\sigma$ has only one atomic proposition (namely, $b_s$). We now define a binary relation $\varphi$ between the states of $M_{fD}$ and the sequences of states of $D$. Let $\varphi \subseteq T \times S^*$ be defined as follows:

$$(t_{in}, s_{in}) \in \varphi \quad \text{and} \quad (\sigma, t, \rho s) \in \varphi \iff (\sigma, \rho) \in \varphi \text{ and } M_{fD}, t \models \psi_s.$$ 

Let $\varphi(\sigma) = \{\rho \mid (\sigma, \rho) \in \varphi\}$. We show that $|\varphi(\sigma)| = 1$ for every $\sigma \in T$. This is done by induction on the partial order $\sigma \subseteq \sigma t$. The base case is $\sigma = t_{in}$ and $\rho = s_{in}$, which is unique by definition. The induction step goes as follows. Assume $|\varphi(\sigma')| = 1$ for all $\sigma' \subseteq \sigma$. Consider $\sigma t$ and let $\varphi(\sigma) = \rho$ (by induction hypothesis) and $M_{fD}, \sigma t \models \psi_s$. We know that there exists no other $\psi_{s'}$ such that $M_{fD}, \sigma t \models \psi_{s'}$ and $s' \neq s''$. Thus, $\varphi(\sigma t) = \{\rho s\}$. Thus $\varphi$ is a well-defined function in $(S^*)^T$.

2. Let $\eta$ be the function defined by:

$$\eta(\rho) = \{\mu \in \Delta(\rho) \mid \exists (\sigma t, \rho s') \in \varphi \text{ with } s' \in \text{supp}(\mu)\}.$$ 

We claim that $\eta$ is a HD-scheduler of the MDP $D$. To prove this, it suffices to show that $\eta$ satisfies the following properties for each $\rho \in \text{dom}(\eta)$:

a. (Progress.) There is a state $s \in \text{supp}(\mu)$ for some $\mu \in \Delta(\rho)$ with $\rho s \in \text{dom}(\eta)$.

b. (Uniqueness.) $|\eta(\rho)| = 1$, i.e., $\eta(\rho)$ defines a unique distribution $\mu \in \Delta(\rho)$.

2.a.) Assume the contrary, i.e., for every $(\sigma t, \rho s) \in \varphi$, $s \not\in \bigcup_{\mu \in \Delta(\rho)} \text{supp}(\mu)$. Since $P(\sigma, \sigma t) > 0$ and $M_{fD}, \sigma t \models \psi_s$, we have $M_{fD}, \sigma \models [X \psi_s]_0$. Furthermore, $M_{fD}, \sigma \models \bigvee_{\mu \in \Delta(\rho)} (\lambda_{s'\in \text{supp}(\mu)} [X \psi_{s'}]_{\mu(s')})$. Hence, $M_{fD}, \sigma t \models \psi_s \land \psi_{s'}$, where $s' \in \bigcup_{\mu \in \Delta(\rho)} \text{supp}(\mu)$. Contradiction.

2.b.) Assume the contrary, $|\eta(\rho)| > 1$, or equivalently, there exist $(\sigma t', \rho s') \in \varphi$ and $(\sigma t'', \rho s'') \in \varphi$ and $s' \in \text{supp}(\mu')$ and $s'' \in \text{supp}(\mu'')$ with $\mu' \neq \mu''$. This implies $\sigma$ satisfies two or more conjuncts of the disjunction $\bigvee_{\mu \in \Delta(\rho)} (\lambda_{s'\in \text{supp}(\mu)} [X \psi_{s'}]_{\mu(s')})$. That is, $M_{fD}, \sigma \models \lambda_{s'\in \text{supp}(\mu')} [X \psi_{s'}]_{\mu'(s')}$ and $\lambda_{s''\in \text{supp}(\mu'')} [X \psi_{s''}]_{\mu''(s'')}$ with $\mu' \neq \mu''$. Hence, $\sigma$ must have a transition to a state $\sigma t$, where $\varphi(\sigma t) = \rho s_1$ and $\varphi(\sigma t) = \rho s_2$ with $s_1 \in \text{supp}(\mu')$ and $s_2 \in \text{supp}(\mu'')$, else the probabilities will not sum up to one. (Figure 1.) Contradiction. Hence, if $\rho s'$ and $\rho s''$ are in $\text{dom}(\eta)$, then $s', s''$ belong to the support of a unique distribution $\mu \in \Delta(\rho)$.

3. To show that $M_{fD} \approx H \approx D$ it suffices to establish a probabilistic bisimulation on the disjoint union of $M_{fD}$ and $D$. Let $\mathcal{MC} D_\eta = (S, P, A P, L, s_{in})$. Let relation $R \subseteq (T \cup S)^2$ be defined by:

- For $(\sigma, \rho) \in T \times S$, $\sigma R \rho$ if $\varphi(\sigma) = \rho$.
- For $\sigma, \rho \in T$, $\sigma R \rho$ if $\varphi(\sigma) = \varphi(\rho)$.
- For $\sigma, \rho \in S$, $\sigma R \rho$ if $\sigma = \rho$.
- For $(\sigma, \rho) \in T \times S$, $\rho R \sigma$ if $\varphi(\sigma) = \rho$.

It is easy to see that $R$ is an equivalence relation. We will show that $R$ is a probabilistic bisimulation. The interesting case is when $\sigma \in T$ and $\rho \in S$ and $\sigma R \rho$. This means $(\sigma, \rho) \in \varphi$, which is equivalent to $\sigma \models \psi_s$, where $s = \rho 1$. Hence,
\[ L(\sigma) = L(\rho) = \{b_s\}. \] If \( M_{f_D}, \sigma \models \psi_{s'} \) and \( M_{f_D}, \sigma \cdot t \models \psi_{s'} \), for some \( s' \in S \), then \( \sigma \cdot t \mathcal{R} \sigma \cdot t' \), since \( \varphi(\sigma \cdot t) = \varphi(\sigma \cdot t') = \rho \cdot s' \). Thus \( \psi_{s'} \) uniquely identifies the equivalence class restricted to the successors of \( \sigma \). Let \( C \) be the equivalence class containing the successors of \( \sigma \) such that for any \( \sigma \cdot t \in C \), \( \sigma \cdot t \models \psi_{s'} \). If \( M_{f_D}, \sigma \models \psi_s \), then there exists a unique \( \mu \in \Delta(s) \) such that \( \sigma = \Lambda_{\text{supp}(\mu)}[X \psi_{s'}]_{\mu}(\sigma') \). Thus \( P(\sigma, C) = \sum_{s' \in \psi_{s'}} P(\sigma, \sigma' \cdot t') = \mu(s') = P(\rho, \rho \cdot s') \). This establishes that \( R \) is a probabilistic bisimulation.

1\(\frac{1}{2}\)-player games with PCTL objectives: The set of 1\(\frac{1}{2}\)-player games with PCTL-winning objectives is defined as 1\(\frac{1}{2}\)PCTL-game = \( \{(D, f) \mid \exists \eta \in \text{HD}(D) : D_\eta \models f \} \). Consider the following problem: Does there exist a winning strategy for a 1\(\frac{1}{2}\)-player game with PCTL-formula \( f \)? This problem can be stated as follows:

**Definition 7 (1\(\frac{1}{2}\)-player PCTL game decision problem).** The problem is to check whether for MDP \( D \) and PCTL-formula \( f \), \( (D, f) \) is in 1\(\frac{1}{2}\)PCTL-game.

We will now show that the above problem can be (effectively) converted into a PCTL satisfiability problem.

**Proposition 2.** For each pair \( (D, f) \), there exists a PCTL-formula \( g \) such that \( (D, f) \in 1\frac{1}{2}\)PCTL-game iff \( g \) is satisfiable.

**Proof.** Let MDP \( D = (S, \Delta, AP, L, s_{in}) \). We construct MDP \( \overline{D} \) from \( D \) such that each state has a unique label, i.e., \( \overline{D} = (S, \Delta, AP', L', s_{in}) \) where \( AP' = \{b_s \mid s \in S \} \) and \( L'(s) = \{b_s \} \). Observe that \( \text{HD}(D) = \text{HD}(\overline{D}) \). The HD scheduler \( \eta \in \text{HD}(D) \) induces the MCs \( D_\eta = (S, P, AP, L, s_{in}) \) and \( \overline{D}_\eta = (S, P, AP', L', s_{in}) \). Note that the set of states and the transition probability functions of these MCs coincide; the MCs only differ in their labelling. Consider the partitioning \( \pi = \{S_s \mid S_s = \{\sigma \mid \sigma \downarrow = s \} \} \) of \( S \). Hence, \( D_\eta, \pi \equiv \overline{D}_\eta, \pi \). This partitioning is independent from the chosen scheduler \( \eta \) and it is insensitive to the labelling function of \( D_\eta \) for any \( \eta \). From Proposition 1 it follows: \( D_\eta, s \models f \) iff \( \overline{D}_\eta, s \models f_\pi \). Thus, it suffices to look for a winning strategy in the problem instance \( (\overline{D}, f_\pi) \). By Def. 6 \([f_\pi] = [\overline{D}]\) for PCTL-formula \( f_\pi \). It follows \( \exists \eta \in \text{HD}(\overline{D}) : \overline{D}_\eta \models f_\pi \) iff \([f_\pi] \cap [f_\pi] \neq \emptyset \). Thus the required formula \( g = f_\pi \wedge f_\pi \).
Theorem 2. The satisfiability problem for PCTL is $\Sigma_1^1$-hard. The finite satisfiability problem for PCTL is $\Sigma_0^1$-complete.

Proof. Theorem 3.4 of [4] states that the existence of a HD strategy in a 1 $\frac{1}{2}$-player game with PCTL objective is $\Sigma_1^1$-hard (analytic hierarchy). By Proposition 2 it then follows that the satisfiability problem for PCTL is $\Sigma_1^1$-hard. The finite satisfiability problem for PCTL is as follows: Does there exist a finite model for

Thus, an infinite path $w$ satisfies $f \land g$ iff $M, w_i \models g$ for some $i \leq n$ and for every $j < i$, $M, w_j \models f$. A tree MC is a Markov chain whose underlying digraph is a tree. Every MC can be converted into a tree MC by unfolding. Let the degree of a tree be the supremum over all out-degrees of its nodes. The finite tree MC $M_{s,n}$ is obtained from $M$ by unfolding starting from state $s$, where each path of the tree is of maximal length $n$. The node $s$ is the root and $n$ is the depth of the tree. The leaves of the tree are made absorbing by adding self-loops with probability one. Observe that the satisfaction relation is monotonic on the unfolding depth $n$, i.e., $M, s \models f$ implies $M, s, m \models f$ for all $m \geq n$. For PCTL$_X$ formula $f$, let $\text{ord}(f)$ be recursively defined as follows:

Thus, for PCTL$_X$ formula $f$ and MC $M$: $M, s \models f$ implies $M_{s,n}, s \models f$ with $n = \text{ord}(f)$.
Proof. The proof is by induction on the structure of the formula $f$. The details can be found in the appendix.

The set of sub-formulas of PCTL$_X$-formula $f$ is denoted by sub($f$). Let sub$_{path}(f) = \{ g U^k h, X g \mid [g U^n h]_p, [X g]_p \in \text{sub}(f), 0 \leq k \leq n \}$. These definitions are lifted to sets of formulas in the usual way, i.e., sub$(H) = \bigcup_{f \in H} \text{sub}(f)$ and sub$_{path}(H) = \bigcup_{f \in H} \text{sub}_{path}(f)$. We will now prove that PCTL$_X$-formulas can be satisfied by MCs of bounded width. A similar result has been obtained in \[9\], though the argument here is simpler on the account that we are dealing with bounded until. First we appeal to an elementary result from computational geometry.

**Proposition 4 (Dual of Helly’s theorem).** Let $T$ be a countable set of vectors in an $n$-dimensional space ($\mathbb{R}^n$). If a vector $y$ is a convex combination of vectors from $T$, then there exists a set $T’ \subseteq T$ such that $y$ is a convex combination of vectors from $T’$ and $|T’| \leq n+1$.

Proof. The vector $y$ is inside the convex polytope defined by $T$. A triangulation of a polytope is a partitioning of the space inside the convex polytope using $(n+1)$-simplexes (tetrahedrons) in $n$-dimensions. Such a triangulation always exists even if the convex polytope is generated by a countable set of points. Thus, $y$ is inside (or on) some $n+1$-simplex whose vertices are in $T’ \subseteq T$. Thus, $y$ can also be defined as a convex combination of vectors in $T’$.

**Proposition 5.** If a set $H$ of PCTL$_X$-formula is satisfiable, then it is satisfiable by a tree MC with degree at most $|\text{sub}(H)|+1$.

Proof. Let $M$ be a tree MC rooted as $s$ such that $M, s \models H$ and $F$ denote $\text{sub}(H) \cup \text{sub}_{path}(H)$. Consider an enumeration $\mathcal{J} : F \to [1, \ldots, |\text{sub}(H)|]$ where:

1. For every $f_1, f_2 \in \text{sub}(H)$, if $f_1 \neq f_2$ then $\mathcal{J}(f_1) \neq \mathcal{J}(f_2)$.
2. $\mathcal{J}(a U^k b) = \mathcal{J}(a U^{k+1} b)$, for any $[a U^n b]_p \in \text{sub}(H)$, $0 \leq k_1, k_2 \leq n$.

Consider the vector space $\mathbb{R}^{\text{img}(\mathcal{J})}$ and define a vector $\bar{t}$ for each state $t$ in $M$:

1. Let $f \in \text{sub}(H)$ be a formula not of the type $[g]_p$. If $M, t \models f$ then $\bar{t}(\mathcal{J}(f)) = 1$ else $\bar{t}(\mathcal{J}(f)) = 0$.
2. Let $f \in \text{sub}_{path}(H)$. If $M, t \models [f]_p$ then $\bar{t}(\mathcal{J}(f)) = p$.

From the semantics of PCTL$_X$, we obtain the following equalities:

1. If $M, s \models [X g]_p$ then $\sum_{t \in \text{succ}(s)} P(s, t) \bar{t}(\mathcal{J}(g)) = p$.
2. If $M, s \models [g U^n h]_p$ and $M, s \not\models h$ then $\sum_{t \in \text{succ}(s)} P(s, t) \bar{t}(\mathcal{J}(g U^{n-1} h)) = p$.

(It follows by construction, if $M, s \models [g U^n h]_p$ and $M, s \models h$ then $\bar{t}(\mathcal{J}(g U^k h)) = 1$ for $k \geq 0$.) Thus, for each $f \in \text{sub}_{path}(H)$ (for $f = g U^n h$ and $s \not\models h$), $\bar{t}(\mathcal{J}(f))$ is defined by a linear combination of $\bar{t}(\mathcal{J}(f))$, for each $t \in \text{succ}(s)$. We now apply Proposition \[4\] to select a subset $T’ \subseteq \text{succ}(s)$ such that $|T’| \leq |\text{sub}(H)| + 1$ and redistribute the probability $P'(s, t)$ on the state $t$ in $T’$ in order to get:

\[1\] We assume each sub-formula in $H$ is unique.
1. $\sum_{t \in T} P'(s, t) = 1$.
2. $M, s \models [Xg]_\sigma$, then $\sum_{t \in \text{succ}(s)} P'(s, t) \triangleright (J(g)) = p$.
3. If $M, s \models [g \cup^nh]_\sigma$ and $M, s \not\models h$ then $\sum_{t \in \text{succ}(s)} P'(s, t) \triangleright (J(g \cup^{n-1}h)) = p$.

This gives us a new tree MC $M'$, such that the out-degree of $s$ is less than $|\text{sub}(H)| + 1$. Straightforward induction on the structure of the formulas in $H$ shows that $\forall f \in H, M, s \models f$ iff $M', s \models f$. We continue this selection process for every state in $M'$ yielding a bounded degree tree.

Form Propositions 3 and 5 we obtain the small model theorem of PCTL $L$.

**Theorem 3.** If a PCTL $X, U^\omega$-formula $f$ is satisfiable then it is satisfiable by a finite tree MC of depth ord($f$) and degree |sub($f$)| + 1.

The size of a PCTL $X, U^\omega$-formula $f$ is defined as size($f$) = |ord($f$)| + |sub($f$)|. Note that the small model theorem states that every formula $f$ is satisfiable in a tree MC whose number of nodes is exponential in size($f$), not the space needed to encode $f$.

**Complexity of $P_{X, \omega}$ satisfiability** We will now show that the satisfiability problem for PCTL $X, U^\omega$ without the bounded until is $PSPACE$-complete. We distinguish the following sub-logics. Let $P_{X, 0}$ be the set of formula defined by the syntax: $\phi ::= a | \phi \land \phi | \neg \phi$ where $a \in AP$. The logic $P_{X, i}$ is defined inductively as follows:

$\phi ::= a | \phi \land \phi | \neg \phi | [X\psi]_p$

where $\psi \in P_{X, i-1}$, $\succeq \in \{<, >, \leq, \geq\}$ and $p \in [0, 1]$. $P_{X, \omega}$ is the set of formula with unbounded number of nested next operators. $P_{X, \omega}$ coincides with PCTL $X, U^\omega$-without bounded until.

**Proposition 6.** The satisfiability problem for $P_{X, \omega}$ is $PSPACE$-hard.

**Proof.** The logspace reduction from quantified boolean formula is given in the appendix. The construction is identical to [15].

Next we present an algorithm to solve the satisfiability problem for formulas in $P_{X, i}$. Let $T_i$ be a non-deterministic Turing machine with an oracle $\Omega_{i-1}$. Oracle $\Omega_j$ can foretell whether a set of formulas in $P_{X,j}$ is satisfiable. Let $H$, the set of formulas in $P_{X, i}$, be the input to $T_i$. The machine proceeds in the following steps:

1. If $f = f_1 \land f_2 \in H$, then remove $f$ from $H$ and add $f_1$ and $f_2$ to $H$.
2. If $f = -(f_1 \land f_2) \in H$, then remove $f$ from $H$ and non-deterministically choose $i \in \{1, 2\}$ and add $\neg f_i$ to $H$.
3. If $f = [Xg]_{sp}$ and $f \in H$, then remove $f$ from $H$ and add $[Xg]_{sp}$ to $H$.

The above steps are repeated until $H$ cannot be changed any further. This can be done in linear time in the size of the input set $H$. At the end, $H$ only contains atomic propositions $a$, negative atomic propositions $\neg a$ or formulas with next operator, $[X\psi]_{sp} \in P_{X, i}$. The machine $T_i$ executes the following steps:

---

2 Reader may refer to [14] for background on oracle Turing machines and polynomial hierarchy.
1. If \( H \cap P_{x_0} \) is unsatisfiable then \( T_i \) moves to a reject state.
2. Else, \( T_i \) chooses a weighted cover \( C = (C, \mu) \), where \( C \subseteq 2^{P_{x_{i-1}}} \), \( \mu \in D_C \). A valid weighted cover \( (C, \mu) \) has the following properties:
   \[
   (1) \forall s \in C : s \subseteq \text{sub}(H) \cap P_{x_{i-1}} ;
   (2) \forall \lfloor Xg \rfloor_{sp} \in H, \sum_{s \in \text{ges}} \mu(s) \equiv p ;
   (3) \sum_{s \in C} \mu(s) = 1 ;
   (4) \forall s \in C : \wedge_{g \in s} g \neq \text{false}.
   \]

The machine first selects a weighted cover \( (C, \mu) \) of \( H \) and then checks whether \( C \) has the properties (1), (2), (3) and (4). By Proposition 5, it suffices to guess a cover where \( |C| \leq |H| + 1 \). Clause (1) can be checked in quadratic time in the size of \( H \). Clause (2) and (3) can be checked by solving linear constraints, this can be done in quadratic time. Clause (4) can be checked by asking the oracle \( \Omega_{i-1} \), whether for every \( s \in C, \wedge_{g \in s} g \) is satisfiable. This is possible since formulas in the set \( s \) are in \( P_{x_{i-1}} \). \( T_i \) moves to accept if such a weighted cover exists else it moves to reject.

The correctness of the above algorithm is straightforward. The algorithm accepts \( H \) by generating a model (tree MC) based on the feasible solution of the linear in-equations (clause (2),(3)) and the decision of the oracle (clause (4)) if and only if \( H \) is satisfiable. We leave the details to the reader. Thus, the satisfiability of a set of \( P_{x_i} \) formulas can be solved by a non-deterministic Turing machine with an oracle \( \Omega_{i-1} \) in polynomial time.

**Proposition 7.** The satisfiability problem for \( P_{x_{\omega}} \) is in \( \text{PSPACE} \).

**Proof.** The satisfiability problem for \( P_{x_{\omega}} \) is in \( \text{NP}^{\text{NP}} \), hence in \( \text{PSPACE} \).

**Theorem 4.** The satisfiability for \( P_{x_{\omega}} \) is \( \text{PSPACE-complete} \).

**Complexity of PCTL\( _X \),\( _U \)-satisfiability** In the rest of the section, we consider the full PCTL\( _X \),\( _U \)-logic (with bounded until).

**Proposition 8.** The satisfiability of PCTL\( _X \),\( _U \)-formula is EXPTIME-hard in the encoding of the formula.

We will need the following machinery to solve the satisfiability problem.

**Proposition 9.** Given a finite tree \( T \) and a PCTL\( _X \),\( _U \)-formula \( f \), we can decide in \( \text{NP} \)-time whether there exists a tree MC \( M \) satisfying \( f \), with \( T \) as the underlying graph.

**Proof.** The satisfiability problem of PCTL\( _X \),\( _U \)-is converted to a satisfiability problem in the theory of reals. In the appendix, we define the algorithm for the conversion and an \( \text{NP} \)-time variable elimination method to solve the satisfiability problem for the theory of reals.

**Theorem 5.** The satisfiability problem for PCTL\( _X \),\( _U \)-is NEXPTIME in the size of the formula.

**Proof.** Theorem 5 and Proposition 9 suggest the following algorithm to solve the satisfiability problem. We non-deterministically guess a tree \( T \) of size \( 2^{O(\text{size}(f))} \). Then check whether there exists an MC with the underlying graph \( T \) that satisfies \( f \). The algorithm works in \( \text{NTIME}(2^{O(\text{size}(f))}) \subseteq \text{NEXPTIME} \) in the size of the formula.
5 Conclusion

We have shown that the PCTL satisfiability problem is $\Sigma^1_1$-hard by reducing it to $1^{\frac{1}{2}}$-player games with PCTL winning objectives [4]. We have presented the sub-logic PCTL$_{X,U_n}$ which possesses the small model property. We have shown that the satisfiability problem for PCTL$_{X,U_n}$ is decidable and given an EXP-TIME algorithm in the size of the formula. We have also considered fragments $(P_{x_0}, \ldots, P_{x_n})$ of PCTL$_{X,U_n}$ and shown the hierarchical complexity of their satisfiability problem.

We observe that if a PCTL$_{X,U_n}$ formula is satisfiable then it is satisfiable in a MC with rational transition probabilities (the variable elimination procedure works with rational). Bertrand et al. [1] show that in the bounded setting (fixing the number of states of a model, a priori) this statement does not hold. The hardness result for PCTL$_{X,U_n}$ satisfiability gives us a polynomial reduction from the acceptance problem of an alternating Turing machine to the encoding (space) of the formula. But, the algorithm runs in NEXPTIME in the size of the formula. Reducing this gap is an open problem.

References

1. Nathalie Bertrand, John Fearnley, and Sven Schewe. Bounded satisfiability for PCTL. In CSL, volume 16 of LIPIcs, pages 92–106. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.
2. Patrick Billingsley. Probability and Measure. Wiley & Sons, 1995.
3. Garrett Birkhoff and Saunders Mac Lane. A Survey of modern algebra. AKP classics. A.K. Peters, Wellesley, Massachusetts, 1997.
4. Tomas Brazdil, Vaclav Brozek, Vojtech Forejt, and Antonin Kucera. Stochastic games with branching-time winning objectives. In LICS, pages 349–358. IEEE Computer Society, 2006.
5. Tomáš Brázdil, Vojtech Forejt, Jan Kretínský, and Antonín Kucera. The satisfiability problem for probabilistic CTL. In LICS, pages 391–402. IEEE Computer Society, 2008.
6. John Canny. Some algebraic and geometric computations in pspace. In STOC, pages 460–467. New York, NY, USA, 1988. ACM.
7. Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. Alternation. J. ACM, 28(1):114–133, January 1981.
8. E. Allen Emerson. Temporal and modal logic. In Handbook of Theoretical Computer Science, pages 995–1072. Elsevier, 1995.
9. Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. Journal of Computer and System Sciences, 18(2):194 – 211, 1979.
10. Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. Formal Aspects of Computing, 6(5):512–535, 1994.
11. Sergiu Hart and Micha Sharir. Probabilistic propositional temporal logics. Information and Control, 70(23):97 – 155, 1986.
12. Dexter C. Kozen. Theory of Computation. Springer, 2006.
13. Richard E. Ladner. The computational complexity of provability in systems of modal propositional logic. SIAM J. Comput., 6(3):467–480, 1977.
14. Kim G. Larsen and Arne Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1 – 28, 1991.

15. Sally Popkorn. *First Steps in Modal Logic*. Cambridge University Press, 1994.
A Bound on the depth of the models of $\text{PCTL}_{\omega, U}$

**Proposition 3.** For every $\text{PCTL}_{\omega, U}$-formula $f$ and MC $M$: $M,s \models f$ implies $M_{s,n}, s \models f$ with $n = \text{ord}(f)$.

**Proof.** We proceed by induction on the structure of the formula $f$. Assume $M,s \models f$.

1. $f = a$ Then, $n = \text{ord}(f) = 1$. By definition, $M_{s,n}$ consists of a single node $s$ equipped with a self-loop. If $M,s \models f$ then $a \in L(s)$. Hence, $M_{s,n}, s \models f$.

2. $f = f_1 \land f_2, n = \max(\text{ord}(f_1), \text{ord}(f_2))$ is larger than or equal to $\text{ord}(f_1)$ and $\text{ord}(f_2)$. By induction hypothesis and monotonicity, it follows $M_{s,n}, s \models f_1$ and $M_{s,n}, s \models f_2$. Thus, $M_{s,n}, s \models f$.

3. $f = \neg g$. For $f = \neg a$ the argument is similar to case 1. For $f = \neg (g_1 \land g_2)$, $M,s \models \neg g_1$ or $M,s \models \neg g_2$ and the rest follows from induction hypothesis. For $f = \neg [h]s$, $f = [h]s$, which is handled below.

4. $f = [Xg]s$. Let $M,s \models f, S' = \{t \mid M,t \models g \land P(s,t) > 0\}$ and $m = \text{ord}(g)$.

   By induction hypothesis, $M_{t,m}, t \models g$ for every $t \in S'$. By construction, $M_{t,m}$ is a subtree of $M_{s,m+1}$ for every $t \in S'$ and $\sum_{t \in S'} P(s,t) > 0$. Thus, $M_{s,n}, s \models f$.

5. $f = [gU^nh]s$. Suppose $M,s \models f, n_1 = \text{ord}(g)$ and $n_2 = \text{ord}(h)$. If $M,s \models h$ then $1 = p$ and the statement follows from the induction hypothesis. Assume $M,s \not\models h$. Consider an infinite path $w$ starting in $s$ with $w \models gU^nh$.

   Thus, there exists a $0 < i < n$ such that $M, w_i \models h$ and for every $j < i$, $M, w_j \models g$. By induction hypothesis, $M_{w_i,n_2}, w_i \models h$ and for any predecessor $w_j$, $M_{w_j,n_1}, w_j \models g$. Or, $M_{w_{i-1}, n_2}, w_{i-1} \models h$ and $M_{w_{i-1}, n_1}, w_{i-1} \models g$, where $n' = \max(\text{ord}(g) + 1, \text{ord}(h))$. For $m = n' + n$, $M_{w_{i-1}, m'}$ is a subtree of $M_{s,m}$, therefore $M_{s,m}, w_{i-1} \models g$ and $M_{s,m}, w_i \models h$. This is true for any path $w$ from $s$, that satisfies $gU^nh$. Thus, $M_{s,m}, s \models f$.

This concludes the proof.

B PSPACE lower bound for $\text{P}_{\omega}$

We will show that satisfiability of $\text{P}_{\omega}$ is PSPACE-hard as well. The hardness proof uses only the operator $\text{[Xg]}_{\omega}$. The semantics of $\text{[Xg]}_{\omega}$ is then similar to the $\square$ operator of modal logic $K$. Henceforth, we will use $\text{Xg}$ to denote $\text{[Xg]}_{\omega}$ and $\Diamond g$ to denote $\neg \text{[Xg]}_{\omega}$ (which is equivalent to $\text{[Xg]}_{\omega}$). We will use the result in [13], which proves that the satisfiability of modal formulas in $K$-system is PSPACE-hard.

The main idea behind the reduction (identical to [13]) is to give a logspace transducer to convert every instance of a QBF to a formula in $\text{P}_{\omega}$. Let $f$ be a QBF $Q_1 x_1 \cdots Q_m x_m \varphi(x_1, \cdots, x_m)$, where $Q_i \in \{\exists, \forall\}$, $x_i$ is a boolean variable ($1 \leq i \leq m$) and $\varphi(x_1, \cdots, x_m)$ is a quantifier free boolean formula with variables $x_1, \cdots, x_m$.

---

3 The more appropriate modal logic system would be with $K$ and serial axioms.
We will use new propositions \( y_0, \ldots, y_m \) to uniquely encode the index \( 0 \leq i \leq m \).

For that purpose, let \( z_1, \ldots, z_n \), where \( n = \lceil \log m \rceil \) be new propositions such that
\[ y_i \equiv \beta_{i,1} z_1 \land \cdots \land \beta_{i,n} z_n \]
for \( 0 \leq i \leq m \), where \( \beta_{i,j} = 1 \) if the \( j \)th bit of (binary) \( i \) is zero else \( \beta_{i,j} \) is a empty string (1 \( \leq j \leq n \)). Let \( Q_1 \) represent the conjunction of all such equivalences. Next we define the \( P_X \) formula \( g \) which uses propositions \( x_1, \cdots, x_m, y_1, \cdots, y_m, z_1, \cdots, z_n \). The formula \( g \) is a conjunction of the following formulas:

\[
\boxdot^m g_1 \quad (F1)
\]
\[
y_0 \quad (F2)
\]
\[
\boxdot^m (y_i \rightarrow \bigodot y_{i+1}) \text{ for each } 0 \leq i < m \quad (F3)
\]
\[
\boxdot^m (y_i \rightarrow ((x_i \rightarrow \boxdot^{m-i} x_{i+1}) \land (\neg x_i \rightarrow \boxdot^{m-i} \neg x_{i+1}))) \text{ for each } 0 \leq i < m \quad (F4)
\]
\[
\boxdot^{m} (y_i \rightarrow (\bigcirc (y_{i+1} \land x_{i+1}) \land (\bigdiamond (y_{i+1} \land \neg x_{i+1})))) \text{ if } Q_i = \forall, \ 0 \leq i < m \quad (F5)
\]
\[
\boxdot^{m} (y_m \rightarrow \phi) \quad (F6)
\]

where \( \boxdot^m h = h \land \boxdot^{m-1} h \) and \( \boxdot^0 h = h \). Intuitively, \( \boxdot^m h \) is true at \( s \) if \( h \) is true at every state reachable from \( s \) within \( m \) steps. The idea behind the reduction is that any model of \( g \) simulates the formula \( f \). Suppose \( s \) satisfies \( g \), the variable \( y_i \) marks the states of the tree (rooted at \( s \)) at depth \( i \), (implemented by (F1), (F2) and (F3)). If the \( i \)th quantifier is universal, then (F5) guarantees that there are two descendants, one of which makes \( x_i \) true and the other makes \( \neg x_i \) true. Once, \( x_i \) (or \( \neg x_i \)) is chosen at a branch, it remains unaltered for every descendant, this is guaranteed by (F4). Finally, we want to evaluate the quantifier free boolean formula \( \phi \). This is implemented by (F6).

To see that only logspace is sufficient to produce the output \( g \), observe that at each step we need to be able to count the index \( i \) \( (0 \leq i \leq m) \), which can be stored in logspace of the working tape, and write the corresponding string (the formula as defined by (F1), (F2), (F3), (F4), (F5) and (F6)) in the output tape.

\[C\] **EXPTIME** lower bound for **PCTL\( X, U^n \)**

We will show EXPTIME-hardness by encoding computations of an alternating Turing machine. Similar technique was also used in [9] to show EXPTIME hardness for PDL. An alternating Turing machine (ATM) [7] is just like a non-deterministic Turing machine except there is a function in the specification of the machine called type. The function type tells us whether a state is an and-state or an or-state. An ATM with only or-states behaves exactly like a non-deterministic Turing machine. Formally, an ATM is a seven tuple \( A = (Q, \Theta, \Gamma, \delta, q_0, \text{type}, F) \).

\( Q \) is a finite set of states, \( \Theta \) is a finite set of input symbols, \( \Gamma \) is a finite set of tape symbols \((\Theta \subseteq \Gamma)\), \( \delta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, R\} \) is a transition relation, \( q_0 \) is an initial state, type: \( Q \rightarrow \{\land, \lor\} \), \( F \subseteq Q \) is the set of accepting states.

Configurations \( \sigma = xqay \in \Gamma^+ \times Q \times \Gamma^+ \), where the tape content is \( xay = \text{tape}(\sigma) \in \Gamma^+ \) with \( a \in \Gamma \), the head is at position \( |x|+1 \), presently reading input \( a \) and the current state is \( q = \text{state}(\sigma) \). A configuration \( \sigma \) is an and-configuration
(or-configuration) if type of state($\sigma$) is $\land$ ($\lor$, resp.). $\sigma$ is accepting if state($\sigma$) $\in F$. For $\sigma = x q a y$ the next configuration $\sigma' = x' q' a' y'$ is defined as follows:

- If ($q, a, q', b, L$) $\in \delta$ then $x' a' = x$ and $y' = by$.
- If ($q, a, q', b, R$) $\in \delta$ then $x' = xb$ and $y = a' y'$.

A trace (or a computation) $C$ of $A$ for an input $x_{in}$ is a set of configuration such that, $q_0 x_{in} \in C$ and for every $\sigma \in C$ with state($\sigma$) $\notin F$, if type(state($\sigma$)) = $\lor$ then one of the next configuration $\sigma'$ of $\sigma$ is in $C$, if type(state($\sigma$)) = $\land$ then every next configuration of $\sigma$ is in $C$. Pictorially, $C$ is a tree where each node is a configuration and edges are defined by the next relation. A trace $C$ is accepting for an input $x$ if $C$ is finite and only configuration without a next configuration in $C$ are accepting.

$$L(A) = \{ x \in \Theta^* \mid \text{there exists an accepting trace } C \text{ for } x \}$$

For some function $S : \mathbb{N} \rightarrow \mathbb{N}$, an ATM $A$ is in ASPACE($S(n)$) if for every input $x \in \Theta^*, and every configuration of every trace of $x$ requires at most $S(|x|)$ space. Furthermore, we assume that no configuration is repeated in any trace $C$ of $x$. This is ensured by enumerating every reachable configurations and the numbering can be encoded into $S(|x|)$ cells of the tape. Thus, the number of steps is less than $|T|^{2S(n)}$ or in $2^{O(S(n))}$, where $n = |x|$. We will need the following identity [7]:

$$\text{ASPACE}(S(n)) = \bigcup_k \text{DTIME}(2^{kS(n)}). \quad (3)$$

Now consider an input $x$ of length $n$ to an ATM $A \in \text{ASPACE}(S(n))$, where $m = S(n) + 2$ and the maximum number of steps needed by the machine to accept (or reject) is $k = 2^m$. Observe that $k$ can be encoded in $m$ space. We will construct a PCTL$_X U$ formula from $A$ and $x$ such that every model of the formula will encode a computation of $A$ with input $x$ if $x \in L(A)$. Each node of the model will encode a configuration of the computation and the relation next will be simulated by $\Box ([X]_{i=1})$. We will use the following set of propositions AP:

1. Cell proposition: for each $a \in \Gamma$ and $0 \leq i \leq m$, $C_{a,i} \in AP$.
2. State proposition: for each $q \in Q$, $Q_q \in AP$.
3. Head proposition: for each $0 \leq i \leq n$, $H_i \in AP$.

Intuitively, $C_{a,i}$ denotes that the $i^{th}$ cell of the tape contains symbol $a$, $Q_q$ denotes that the current symbol is $q$ and $H_i$ denotes that the head is on the $i^{th}$ cell. We will use the following formula to correctly capture the behaviour of $A$.

- One state proposition $Q_q$ is true at every node of the model:

  $$g_1 := \bigvee_{q \in Q} \left( Q_q \land \bigwedge_{q' \in Q \setminus \{q\}} \neg Q_{q'} \right)$$

- One cell proposition is true for any particular $i \leq m$.

  $$g_2 := \bigwedge_{i=0}^{m} \bigvee_{a \in \Gamma} \left( C_{a,i} \land \bigwedge_{a' \in \Gamma \setminus \{a\}} \neg C_{a',i} \right)$$
One head proposition is true at any node of the model. Head cannot cross the first and the last cells.

\[ g_3 := \bigvee_{i=1}^{m-1} \left( H_i \land \bigwedge_{j \neq i} \neg H_j \right) \land \neg H_0 \land \neg H_m \]

Unread cell propositions remain unchanged in the next node of the model.

\[ g_4 := \bigwedge_{i=0}^{m} \bigwedge_{a \in \Gamma} \left( \neg H_i \land C_{a,i} \rightarrow \Box C_{a,i} \right) \]

Transition relation for and-states.

\[ g_5 := \bigwedge_{i=1}^{m-1} \bigwedge_{a \in \Gamma} \land \left( H_i \land C_{a,i} \land Q_q \rightarrow \bigwedge_{(q,a,q',b,R) \in \delta} \Diamond (H_{i+1} \land C_{i,b} \land Q_{q'}) \land \bigwedge_{(q,a,q',b,L) \in \delta} \Diamond (H_{i-1} \land C_{i,b} \land Q_{q'}) \right) \]

Transition relation for or-states.

\[ g_6 := \bigwedge_{i=1}^{m-1} \bigwedge_{a \in \Gamma} \land \left( H_i \land C_{a,i} \land Q_q \rightarrow \bigvee_{(q,a,q',b,R) \in \delta} \Diamond (H_{i+1} \land C_{i,b} \land Q_{q'}) \lor \bigvee_{(q,a,q',b,L) \in \delta} \Diamond (H_{i-1} \land C_{i,b} \land Q_{q'}) \right) \]

The accepting nodes of the model satisfy the following formula:

\[ g_F := \bigvee_{q \in F} Q_q \]

Let the input \( x = a_0, \ldots, a_n \), and \( b \) be the symbol for blank space. The initial configuration is defined as follows:

\[ g_{in} := Q_{q_0} \land H_1 \land \bigwedge_{i=1}^{n} \neg H_i \land \bigwedge_{i=1}^{n} C_{a,i} \land C_{b,0} \land \bigwedge_{i=n+1}^{m} C_{b,i} \]

Let \( g = \bigwedge_{i=1}^{6} g_i \). Thus, the required formula is defined as follows:

\[ f := g_{in} \land [g \cup^k g_F]_{=1} \]

The correctness of the translation can be checked by inspection, since there is a one-to-one correspondence between the models of \( f \) and computations of \( A \) on input \( x \). Observe that the encoding takes \( O(|\Gamma| + |Q| + |\delta|)S(n) \) time. If \( S(n) \) is a polynomial function then \( f \) is constructed in polynomial time of the size of \( A \) and \( x \). Furthermore, if \( S(n) \) is polynomial in \( n \) then \( A \in \text{EXPTIME} \) (equation (3)). This leads to the Proposition 8.
D Tree model of PCTL_X,U^n

For every vertex s we have a finite set of formulas F_s ⊆ PCTL_X,U^n. Initially they are all empty, except F_{s_0} = \{f\}. H' is the set of (in)equations, initially empty. We invoke Model-Checking(s_{in}, F_{s_{in}}) to build H'. The new variables added in step 19. of the algorithm can be removed through substitution, yielding the set H.

Algorithm 1 Model-Checking(s, F_s)

1: if F_s ⊆ Px_0 then
2: Return iff F_s is satisfiable.
3: else
4: for each f ∈ F_s do // Builds the closure of F_s
5: if f = f_1 ∨ f_2 ∈ F_s, then remove f and add f_1 and f_2 to F_s.
6: if f = ¬(f_1 ∨ f_2) ∈ F_s, then remove f and non-deterministically choose i ∈ \{1, 2\} and add ¬f_i to F_s.
7: if f = [g]_{p \in} to F_s.
8: if f = [a U^n b]_{p \in}, n > 0, then non-deterministically choose between (if 1 \bowtie p add b to F_s and remove f) or skip.
9: end if
10: end for
11: for each f ∈ F_s do // Build H' and F_t for all successor t of s.
12: if f = [Xg]_{p \in} then
13: Choose non-deterministically a subset S' ⊆ succ(s).
14: for each t ∈ S' do
15: F_t = F_t ∪ \{g\}.
16: H' = H' ∪ (∑_{t ∈ S'} x_{s,t} \bowtie p)
17: end for
18: else if f = [a U^n b]_{p \in}
19: Choose non-deterministically a subset S' ⊆ succ(s).
20: for each t ∈ S' do
21: (F_t = F_t ∪ \{[a U^{n-1} b]_{p_t}\}) or (non-deterministic) (F_t = F_t ∪ \{b\} and p_t = 1). // p_t is a new variable.
22: H' = H' ∪ (∑_{t ∈ S'} x_{s,t} \bowtie p_t \bowtie p)
23: end for
24: end if
25: end for
26: for each t ∈ succ(s) do
27: Model-Checking(t, F_t)
28: end for
29: end for

Proposition 9. Given a finite tree T and a PCTL_X,U^n-formula f, we can decide in NP-time whether there exists a tree MC M satisfying f, with T as the underlying graph.
Proof. Let tree $T = (V, E, s_0)$, where $V$ is the set of vertices, $E$ is the set of directed edges and $s_0$ is the root. For every edge $e \in E$ assign one indeterminate (variable) $x_e$ as the weight of the edge $e \in E$. Let $\mathcal{P} = \{x_e \mid e \in E\}$. We non-deterministically choose a labeling function $L$ and check whether a sub-formula is satisfied at a state. We use a model checking algorithm for MCs against PCTL formula, except we have variables instead of real values (Algorithm 1).

Thus, a formula $f$ is true at $s_0$ iff a set of (real non-linear) (in)equations $H$, with variables in $\mathcal{P}$ is satisfiable. The number of such equations is in $O(|V| \|\text{sub}(f)\|)$ and the number of variables is $|E|$, i.e., polynomial in the size of the input. Using existential theory of reals \cite{4}, we can determine the feasibility of the (in)equations in PSPACE.

We can improve the complexity by exploiting the special structure of the (in)equations. Observe that, every equation has the following form: $a_0\sigma_0 + a_1\sigma_1 + \cdots + a_k\sigma_k = b$, where $a_0, \cdots, a_k, b \in \mathbb{Q}$ and each $\sigma_i$ ($0 \leq i \leq k$) is a term of a polynomial of the type $x_{e_1}, x_{e_2}, \cdots, x_{e_n}$, where $e_{1, i} e_{2, i} \cdots e_{n, i}$ is a path in the tree $T$. Furthermore, the edges $e_{1, i}$ for every $0 \leq i \leq k$ have the same source vertex. In the next section we show how to solve the satisfiability problem of such a system of (in)equations in NP-time.

E Variable elimination

Consider the ring of polynomials $D[X]$ in the integral domain $D$, where $X$ is the set of indeterminates (or variables) \cite{3}. A polynomial $p(x_1, \cdots, x_n)$, with variables $x_1, \cdots, x_n \in X$, is seen as a sum of products with nonzero coefficients in $D$, where each $x_1^{d_1} \cdots x_n^{d_n}$ is called a term; together with its coefficient it is called a monomial; the degree of the term $x_1^{d_1} \cdots x_n^{d_n}$ is $d_1 + \cdots + d_n$; degree of a polynomial is the maximum degree of its terms. A polynomial is multivariate if $|X| > 1$. The ring of multivariate polynomials $D[X]$ can be viewed as a ring of univariate polynomials $D[X \setminus \{x\}][x]$ with coefficients in the Integral domain $D[X \setminus \{x\}]$ \cite{3} page 63, Theorem 2.). Particularly, the degree of a term of a polynomial in $D[X \setminus \{x\}][x]$ is the power of $x$ in that term.

$E(D[X])$ is the set of (in)equations (e.g. $x_1^2 - x_2 \geq 0.4$) where the left hand side (LHS) is a polynomial (e.g. $x_1^2 - x_2$) in $D[X]$ and the right hand side (e.g. 0.4) is in $D$. A variable $x$ is independent of $H \subseteq E(D[X])$ iff $H = H \cap E(D[X \setminus \{x\}])$ else it is dependent. The quotient domain $Q(D)$ is the rational form of the type $\frac{f}{g}$ where $f, g \in D$.

A weighted tree $T$ is a triple $(V, E, w)$, where $V$ is the set of vertices, $E \subseteq V \times V$ is the set of edges and $w$ is an injective weight function from $E \to V$, where $V$ is a set of variables. Let $X = \text{img}(w)$. Define relations next and parent as follows: for $x, y \in X$, $v, v', v_1, v_2 \in V$, with $w^{-1}(x) = (v_1, v)$ and $w^{-1}(y) = (v', v_2)$, $(x, y) \in \text{next}$ iff $v = v'$, and $(x, y) \in \text{parent}$ iff $v_1 = v'$.

next$^*$ is the transitive closure of next. Consider a term $\sigma = x_1 \cdots x_k$ such that for every $1 \leq i < k$, $(x_i, x_{i+1}) \in \text{next}$. Define head($\sigma$) = $x_1$, tail($\sigma$) = $x_k$ and $x_{i+1} \cdots x_k$ as a suffix of $\sigma$, for $1 \leq i \leq k$. Let $H \subseteq E(Q[X])$ be a set of (in)equations with the following properties. For each
\(\xi \in H:\)

- **P1.** For all \(x \in X, \text{lhs}(\xi) \in \mathbb{Q}[X \setminus \{x\}][x] \rightarrow \text{degree}(\xi) \leq 1\)
- **P2.** For each term \(\sigma = x_1 \cdots x_k \in \xi, x_i, x_{i+1} \in \text{next} .\)
- **P3.** If \(\text{lhs}(\xi) = a_1\sigma_1 + \cdots + a_k\sigma_k,\) where \(a_i \in \mathbb{Q}\) and \(\sigma_i\) are terms, then for all \(1 \leq i, j \leq k, (\text{head}(\sigma_i), \text{head}(\sigma_j)) \in \text{parent}.\)

Suppose \(H \subseteq E(\mathbb{Q}[X])\) satisfies properties P1, P2 and P3 and let \(n\) be the number of variables and \(m\) be the number of (in)equations in \(H.\) We only consider positive variable valuations. Thus for every variable \(x\) we have the in-equation \(x > 0\) in \(H.\) We present a non-deterministic algorithm to decide whether \(H\) is satisfiable. We begin by setting \(H_0 = H\) and at each iteration \(i,\) we eliminate a (particular) variable, say \(x\) and transform the set of equations from \(H_i \subseteq E(\mathbb{Q}[X])\) to \(H_{i+1} \subseteq E(\mathbb{Q}[X \setminus \{x\}]\). We consider comparisons \(\preceq\) to be of the type \(\{\leq, =, \geq\}.\) (Strict inequalities can be removed by adding very small positive quantity \(\epsilon.\) For example \(f < g\) can be transformed to \(f + \epsilon \leq g.\)) The algorithm proceeds in the following steps:

1. If \(H_i\) is independent of all variables, then each (in)equation, involves only rational numbers (and \(\epsilon \rightarrow 0\)). Return true iff each (in)equality in \(H_i\) is true.
2. Choose a variable \(x\) such that every variable \(y\) with \((x, y) \in \text{next}^+\) is independent of \(H_i.\)
3. \(H_x\) is the largest subset of \(H_i\) such that every formula in \(H_x\) is dependent on \(x.\) If \(H_x\) is empty then \(H_{i+1} = H_i.\) Suppose \(H_x\) is not empty, every inequation \(\xi \in H_x\) can be transformed to a form \((\sigma x \preceq a_0 + a_1\sigma_1 + \cdots + a_k\sigma_k),\) where \(\sigma, \sigma_1, \cdots, \sigma_k\) are terms in \(\mathbb{Q}[X \setminus \{x\}]\) and \(a_0, \cdots, a_k \in \mathbb{Q}.\) We will denote this form by \(f x \preceq g.\) Set \(H_{i+1} = H_i \setminus H_x.\)
4. Define \(A_\leq, A_\geq \subseteq \mathbb{Q}(\mathbb{Q}[X \setminus \{x\}]),\) for \(\preceq \in \{\leq, =, \geq\}\) as follows:

\[
A_\leq := \left\{ \frac{g}{f} \mid (f x \leq g) \in H_x \right\} \cup \{1\}, \text{ quotients that are at least as large as } x
\]

\[
A_\geq := \left\{ \frac{g}{f} \mid (f x = g) \in H_x \right\}, \text{ quotients that are equal } x
\]

\[
A_\geq := \left\{ \frac{g}{f} \mid (f x \geq g) \in H_x \right\} \cup \{\epsilon\}, \text{ quotients that are at least as small as } x,
\]

where \(g = a_0 + a_1\sigma_1 + \cdots + a_k\sigma_k\) and \(f = \sigma.\)

5. Non-deterministically choose an ordering of elements in \(A_\leq\) and \(A_\geq\). Then we have the following set of (in)equations:

\[
\frac{g_1}{f_1} \leq \cdots \frac{g_{n_1}}{f_{n_1}} \leq \frac{g_{n_1+1}}{f_{n_1+1}} = \cdots = \frac{g_{n_2}}{h_{n_2}} \leq \frac{g_{n_2+1}}{f_{n_2+1}} \leq \cdots \leq \frac{g_{n_3}}{f_{n_3}}
\]

where, \(\frac{g_i}{f_i}\) is in \(A_\leq\) for \(1 \leq i \leq n_1,\) in \(A_\leq\) for \(n_1 + 1 \leq i \leq n_2\) and in \(A_\geq\) for \(n_2 + 1 \leq i \leq n_3.\)

\[\epsilon\] tends to 0 from the positive side.
6. For each $1 \leq j \leq n_3$, we have $\xi_j := (g_j f_{j+1} \bowtie g_{j+1} f_j)$. $\xi'_j$ is obtained from $\xi_j$ by canceling variables that are common divisors of the polynomials in the left hand side and right hand side of $\xi_j$. Add $\xi'_j$ to $H_{i+1}$ for each $\xi_j$ ($1 \leq j \leq n_3$).

First we will show that $H_{i+1}$ created in step 6, satisfies $P1$, $P2$ and $P3$. Consider,

$$\frac{a_0 + a_1 \sigma_1 + \cdots + a_k \sigma_k}{\sigma} \bowtie \frac{b_0 + b_1 \sigma'_1 + \cdots + b_i \sigma'_i}{\sigma'} \tag{5}$$

Let $\xi := (\sigma \times a_0 + a_1 \sigma_1 + \cdots + a_k \sigma_k)$, $\xi' := (\sigma' \times b_0 + b_1 \sigma'_1 + \cdots + b_i \sigma'_i)$ and $\xi_j, \xi'_j \in H_i$ satisfy $P1$, $P2$ and $P3$. From the choice of the variable $x$ (step 2), it is evident that either $\sigma|\sigma'$ or $\sigma'|\sigma$ ($a|b$ means $a$ divides $b$). W.l.o.g let us assumed $\sigma''|\sigma' = \sigma$.

Therefore, equation (5) can be rewritten as:

$$a_0 + a_1 \sigma_1 + \cdots + a_k \sigma_k \bowtie b_0 + b_1 \sigma''_1 + \cdots + b_i \sigma''_i. \tag{6}$$

$P3$ holds for equation (5), this follows trivially, as $\text{head}(\sigma) = \text{head}(\sigma_i) = \text{head}(\sigma'')$ for $1 \leq i \leq k$. $(\text{head}(\sigma'), \text{tail}(\sigma'')) \in \text{next}$, since $\sigma = \sigma'' \sigma'$ and $(\text{head}(\sigma'_i), \text{head}(\sigma''_i)) \in \text{parent}$ for all $1 \leq i, j \leq l$. Thus, the new equations added to $H_{i+1}$ (after canceling common variables) also satisfy $P1$, $P2$ and $P3$ (cancellation is valid since variables can only take positive value).

Correctness of the algorithm is due to the following arguments:

1. Suppose $H_i$ is feasible and $\nu$ be a satisfying valuation of the variables. Then there exists some order among the rational numbers obtained by substituting the values of the variables in the quotients $\{ \frac{a(x_1, \ldots, x_k)}{f(x_1, \ldots, x_n)} \}$ present in $\Lambda_1$ and $\Lambda_2$. If we choose this order to define the order in the equation (4) and obtain $H_{i+1}$ subsequently, then $\nu$ is also a satisfying valuation for (in)equations $H_{i+1}$.

2. If $H_{i+1}$ is satisfiable then the (in)equations (4) is true for some value of $X \setminus \{x\}$. If $\Lambda_2$ is not empty then set $x = \frac{g_{n+1}}{f_{n+2}}$. Else choose a $x$ such that $\frac{g_{n+1}}{f_{n+2}} \leq x \leq \frac{g_{n+2+1}}{f_{n+3+1}}$. The value thus chosen is made strictly greater than $0$, since $\epsilon \in \Lambda_2$. Hence, rational form and cancellation of variables defined in step 5 and step 6, respectively is valid. This gives us a satisfying valuation of $H_i$.

Observe that at each iteration $i$, the size $H_i$ is of $O(|H|)$ and at each iteration we remove one variable and spend $O(nm)$ in obtaining $H_{i+1}$ (modulo division of rational numbers). Thus the maximum number of iteration is $n$ and total time complexity of the non-deterministic algorithm is $O(mn^2)$. Thus satisfiability of set of polynomial equation with properties $P1$, $P2$ and $P3$ is in $\text{NP}$. 