Branching Time? Pruning Time!

Markus Latte[1]* and Martin Lange[2]

[1] Department of Computer Science, University of Munich, Germany
[2] School of Electrical Engineering and Computer Science, University of Kassel, Germany

Abstract. The full branching time logic CTL* is a well-known specification logic for reactive systems. Its satisfiability and model checking problems are well understood. However, it is still lacking a satisfactory sound and complete axiomatisation. The only proof system known for CTL* is Reynolds' which comes with an intricate and long completeness proof and, most of all, uses rules that do not possess the subformula property.
In this paper we consider a large fragment of CTL* which is characterised by disallowing certain nestings of temporal operators inside universal path quantifiers. This subsumes CTL$^+$ for instance. We present infinite satisfiability games for this fragment. Winning strategies for one of the players represent infinite tree models for satisfiable formulas. These can be pruned into finite trees using fixpoint strengthening and some simple combinatorial machinery such that the results represent proofs in a Hilbert-style axiom system for this fragment. Completeness of this axiomatisation is a simple consequence of soundness of the satisfiability games.

1 Introduction

Temporal logics originate from the philosophical tense logics [14] and are now important specification languages in computer science where they are being used to abstractly describe and verify the behaviour of reactive systems [11]. One of their most prominent examples is the full branching-time temporal logic CTL* [2]. Its satisfiability and model checking problem—i.e. the algorithmic nature of the logic—is well understood by now [18,3]. This cannot necessarily be said about the proof-theoretic nature of CTL*: despite CTL*'s long lifespan no clean and simple sound and complete axiomatisation has been found for it so far.

Various axiomatisations for simpler temporal logics like LTL for instance have been known for a long time, and others have been found since [7,5,10,9]. The same can be said about CTL [1,13,9,8]. These two logics enjoy the property that their formulas are modularly composed of a finite number of temporal operators and an axiomatisation can describe the handling of each of them. In CTL* though, the arbitrary mixture and nesting of path formulas leads to an essentially infinite number of temporal operators of which formulas are composed. A proper axiomatisation would have to capture their nature in a finite number of formula schemes. So far, the only successful attempt at presenting a sound and complete axiomatisation is Reynolds' [15]. However, it features an unsatisfactory system because of a rather intricate and difficult completeness proof.

* Supported by the DFG Graduiertenkolleg 1480 (PUMA)

Most of all though, it contains rules which do not have the subformula property, for instance the rule (AA). The subformula property bounds the search space within a proof search. In particular for this rule, there is no (obvious) upper bound on the possible premises in terms of the conclusion because the rule introduces new atoms in its premise. The problem of finding a simple axiomatisation for CTL* with a neat completeness proof is therefore still open.

In this paper we make a step forward in this direction. We consider a large fragment of CTL* by restricting the nesting of temporal operators under a universal path quantifier. We call this fragment CTL$^\sharp$. Syntactically, it supersedes CTL and even CTL$^+$, and the satisfiability problem for CTL$^\sharp$ is therefore already 2EXPTIME-complete [6] which indicates that it is not an easy fragment. It also exceeds the expressive power of these two logics; for instance, it it possible to express that a path satisfying some fairness constraint exists.

Note that this fragment is not closed under complements in the sense that the negation of a formula with restrictions concerning *universal* path quantifiers has—in positive normal form—restrictions on *existential* path quantifiers. This is not a major problem. It simply means that one has to regard the context in which the logic is being used. We consider CTL$^\sharp$ in the context of satisfiability; in the context of validity one has the dual restrictions on existential path quantifiers.

We present a calculus of infinite games characterising the satisfiability problem for CTL$^\sharp$. We then employ combinatorial arguments and logical principles by which winning strategies in these games, which are infinite trees, can be made finite and loop-free. These are then used to derive a complete Hilbert-style axiomatisation for CTL$^\sharp$.

This is, as far as we know, the first attempt at approaching the CTL* axiomatisation problem on the syntactic route "from below". Other attempts are syntactic and "from above" in the sense that they consider a superlogic, for instance CTL* with past operators [16] which apparently makes things easier, or semantic in the sense that they redefine the class of structures over which the logic is interpreted [17].

The rest of the paper is organised as follows. Section "Branching Time" introduces the mentioned temporal logics; section "Playing Time" presents the satisfiability games; in section "Pruning Time" we show how to transform infinite winning strategies into finite ones; and section "Proving Time" presents the axiomatisation.

2 Branching Time

Transition Systems and Paths. A *transition system* is a tuple $\mathcal{T} = (\mathcal{S}, \longrightarrow, L)$ where $\mathcal{S}$ is a set of states, $\longrightarrow \subseteq \mathcal{S} \times \mathcal{S}$ a transition relation and $L : \mathcal{P} \to 2^{\mathcal{S}}$ a function that assigns to each q in some non-empty set $\mathcal{P}$ of atomic propositions the set of states $L(q)$ in which q holds. Here we assume the transition relation to be *total*: for all $s \in \mathcal{S}$ there is a $t \in \mathcal{S}$ such that $s \longrightarrow t$. A *path* is an infinite sequence $\pi = s_0, s_1, \ldots \in \mathcal{S}^\omega$ such that $s_i \longrightarrow s_{i+1}$ for all $i \in \mathbb{N}$. With π^k we denote the *k-th suffix* of π, namely the path $s_k, s_{k+1}, \ldots$.

CTL*. Formulas of the branching time temporal logic CTL* over $\mathcal{P}$ in positive normal form are given by the following grammar.

$$\varphi ::= q \mid \neg q \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathtt{E}\alpha \mid \mathtt{A}\alpha$$

$$\alpha ::= \varphi \mid \alpha \vee \alpha \mid \alpha \wedge \alpha \mid \mathtt{X}\alpha \mid \alpha\mathtt{U}\alpha \mid \alpha\mathtt{R}\alpha$$

where $q \in \mathcal{P}$. The formulas q and $\neg q$ are called *literals*. The constructors E and A are called *path quantifiers*; X, U and R are called *temporal operators*. Formulas derived as φ are called *state formulas*, those that are derived from α are called *path formulas*. The latter only occur as genuine subformulas of the former, i.e. a CTL* formula is one that is derived from φ in this grammar. Throughout this paper we will adopt the convention that small letters of the end of Greek alphabet, like φ, ψ, denote state formulas and small ones from the beginning, like α, β, denote path formulas. Note that every state formula is also a path formula.

We also use the standard abbreviations from temporal logic: $\mathtt{F}\alpha := \mathtt{tt}\,\mathtt{U}\,\alpha$ and $\mathtt{G}\alpha := \mathtt{ff}\,\mathtt{R}\,\alpha$ where $\mathtt{tt} := q \vee \neg q$ and $\mathtt{ff} := q \wedge \neg q$ for some $q \in \mathcal{P}$.

The *closure* of a formula ϑ is the least set $\mathsf{Cl}(\vartheta)$ that contains ϑ and satisfies the following.

– If $Q\alpha \in \mathsf{Cl}(\vartheta)$ for some $Q \in \{\mathtt{E}, \mathtt{A}\}$ then $\alpha \in \mathsf{Cl}(\vartheta)$.
– If $\alpha \wedge \beta \in \mathsf{Cl}(\vartheta)$ or $\alpha \vee \beta \in \mathsf{Cl}(\vartheta)$ then $\{\alpha, \beta\} \subseteq \mathsf{Cl}(\vartheta)$.
– If $\mathtt{X}\psi \in \mathsf{Cl}(\vartheta)$ then $\psi \in \mathsf{Cl}(\vartheta)$.
– If $\varphi \circ \psi \in \mathsf{Cl}(\vartheta)$ then $\{\psi, \varphi, \mathtt{X}(\varphi \circ \psi)\} \subseteq \mathsf{Cl}(\vartheta)$, for all $\circ \in \{\mathtt{U}, \mathtt{R}\}$.

Thus, the closure is essentially the set of all subformulas with the exception of the fixpoint-operators which we also include with a prefixed X-operator. Note that the size of the closure of some φ is linear in its syntactic length. We therefore use it as a measure for the size of a formula: $|\varphi| := |\mathsf{Cl}(\varphi)|$.

Formulas of CTL* are interpreted over states and paths of a transition system $\mathcal{T} = (\mathcal{S}, \longrightarrow, L)$, reflecting the two types of formulas.

$\mathcal{T}, s \models q$	iff $q \in L(s)$
$\mathcal{T}, s \models \neg q$	iff $q \notin L(s)$
$\mathcal{T}, s \models \varphi \vee \psi$	iff $\mathcal{T}, s \models \varphi$ or $\mathcal{T}, s \models \psi$
$\mathcal{T}, s \models \varphi \wedge \psi$	iff $\mathcal{T}, s \models \varphi$ and $\mathcal{T}, s \models \psi$
$\mathcal{T}, s \models \mathtt{E}\alpha$	iff there is a path $\pi = s, s', \ldots$ with $\mathcal{T}, \pi \models \alpha$
$\mathcal{T}, s \models \mathtt{A}\alpha$	iff for all paths $\pi = s, s', \ldots$ we have $\mathcal{T}, \pi \models \alpha$
$\mathcal{T}, \pi \models \varphi$	iff $\mathcal{T}, s \models \varphi$ when φ is a state formula and $\pi = s, s', \ldots$
$\mathcal{T}, \pi \models \alpha \vee \beta$	iff $\mathcal{T}, \pi \models \alpha$ or $\mathcal{T}, \pi \models \beta$
$\mathcal{T}, \pi \models \alpha \wedge \beta$	iff $\mathcal{T}, \pi \models \alpha$ and $\mathcal{T}, \pi \models \beta$
$\mathcal{T}, \pi \models \mathtt{X}\alpha$	iff $\mathcal{T}, \pi^1 \models \alpha$
$\mathcal{T}, \pi \models \alpha\mathtt{U}\beta$	iff there is a $k \in \mathbb{N}$ with $\mathcal{T}, \pi^k \models \beta$ and for all $j < k : \mathcal{T}, \pi^j \models \alpha$
$\mathcal{T}, \pi \models \alpha\mathtt{R}\beta$	iff for all $k \in \mathbb{N} : \mathcal{T}, \pi^k \models \beta$ or there is $j < k : \mathcal{T}, \pi^j \models \alpha$

Two (state) formulas are equivalent, written $\varphi \equiv \psi$, iff for all $\mathcal{T}$ and all states s we have: $\mathcal{T}, s \models \varphi$ iff $\mathcal{T}, s \models \psi$. A state formula φ is valid, written $\models \varphi$, iff $\mathcal{T}, s \models \varphi$ for any transition system $\mathcal{T}$ and any of its states s. Equally, a path formula α is valid, also written $\models \alpha$, if $\mathcal{T}, \pi \models \alpha$ for any transition system $\mathcal{T}$ and any path π in it. Note that $\models \alpha$ iff $\models \mathtt{A}\alpha$.

Finally, we introduce the *dual* $\neg\vartheta$ of a formula ϑ as follows: q and $\neg q$ are dual to each other; the usual deMorgan law's apply; path quantifiers are dual to each other as in $\neg\mathtt{E}\alpha := \mathtt{A}\neg\alpha$ and vice-versa; the next-operator is self-dual as in $\neg\mathtt{X}\varphi := \mathtt{X}\neg\varphi$; and the temporal fixpoint operators are dual to each other as in $\neg(\varphi\mathtt{U}\psi) := (\neg\varphi)\mathtt{R}(\neg\psi)$ and vice-versa. With negation around we can introduce implication $\alpha \rightarrow \beta$ as $\neg\alpha \vee \beta$ as usual. The next section explains why we have introduced formulas in positive normal form and avoided the use of negation as a first-class operators.

CTL$^\sharp$. The fragment CTL$^\sharp$ is obtained from CTL* in positive normal form by disallowing certain nestings of temporal operators inside a universal path quantifier: the arguments to an *until* formula in there must be state formulas. Formally, the syntax of CTL$^\sharp$ formulas is given by the following grammar.

$$\begin{aligned}
\varphi &::= q \mid \neg q \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathtt{E}\alpha \mid \mathtt{A}\beta \\
\alpha &::= \varphi \mid \alpha \vee \alpha \mid \alpha \wedge \alpha \mid \mathtt{X}\alpha \mid \alpha\mathtt{U}\alpha \mid \alpha\mathtt{R}\alpha \\
\beta &::= \varphi \mid \beta \vee \beta \mid \beta \wedge \beta \mid \mathtt{X}\beta \mid \varphi\mathtt{U}\varphi \mid \beta\mathtt{R}\beta
\end{aligned}$$

All the concepts introduced above for CTL* like the closure and the semantics clearly carry over to CTL$^\sharp$ as well. However, note that the dual $\neg\vartheta$ of a CTL$^\sharp$ formula ϑ need not be a CTL$^\sharp$ formula itself. A syntax for the *dual* of CTL$^\sharp$ is obtained from the one above by switching E and A. We set $\neg\text{CTL}^\sharp := \{\varphi \mid \neg\varphi \in \text{CTL}^\sharp\}$ as a subset of CTL*.

It is easy to see that many of the standard and simple types of properties like *safety*, *liveness*, *fairness*, etc. are expressible in some form or the other in CTL$^\sharp$, for instance through $\mathtt{AG}\, q_{safe}$, $\mathtt{AGEF}\, q_{live}$, $\mathtt{AGF}\, q_{fair}$. Also, it is possible to express a standard requirement for schedulers, namely that all requests need to be served at a later point: $\mathtt{AG}(q_{request} \rightarrow \mathtt{F}\, q_{serve})$.

However, it is for example not possible to say that all paths that are fair w.r.t. some predicate α satisfy some property β. This would be $\mathtt{A}(\mathtt{GF}\alpha \rightarrow \beta)$ which would be $\mathtt{A}(\mathtt{FG}\neg\alpha \vee \beta)$ in positive normal form and thus contain an R-formulas in an argument of a U-formula inside a universal path quantifier.

A prominent example of a CTL* formula which essentially bears much of the difficulty of finding a complete axiomatisation is the *limit closure formula*

$$\varphi_{\mathrm{LC}} \quad := \quad q \wedge \mathtt{AG}\big(q \rightarrow \mathtt{EX}(q\,\mathtt{U}\,p)\big) \rightarrow \mathtt{EG}(q\,\mathtt{U}\,p)\ .$$

It is a valid CTL* formula, hence, its dual $\neg\varphi_{\mathrm{LC}}$ is unsatisfiable which is

$$q \wedge \mathtt{AG}\big(\neg q \vee \mathtt{EX}(q\,\mathtt{U}\,p)\big) \wedge \mathtt{AF}(\neg q\,\mathtt{R}\,\neg p)$$

in positive normal form. This is not a CTL$^\sharp$ formula though because the last conjunct is universally path quantified and contains a U-formula (of the abbreviated form F) which itself contains an R-formula in one of its arguments.

$$(\wedge)\ \frac{\varphi,\psi,\Phi}{\varphi\wedge\psi,\Phi} \qquad (\vee)\ \frac{\varphi_i,\Phi}{\varphi_0\vee\varphi_1,\Phi}\ \exists,\, i\in\{0,1\} \qquad (\mathtt{ESt})\ \frac{\varphi,\mathtt{E}\Pi,\Phi}{\mathtt{E}(\varphi,\Pi),\Phi}$$

$$(\mathtt{E}\wedge)\ \frac{\mathtt{E}(\alpha,\beta,\Pi),\Phi}{\mathtt{E}(\alpha\wedge\beta,\Pi),\Phi} \qquad (\mathtt{Ett})\ \frac{\Phi}{\mathtt{E}\emptyset,\Phi} \qquad (\mathtt{EU})\ \frac{\mathtt{E}(\beta,\Pi),\Phi \quad | \quad \mathtt{E}(\alpha,\mathtt{X}(\alpha\mathtt{U}\beta),\Pi),\Phi}{\mathtt{E}(\alpha\mathtt{U}\beta,\Pi),\Phi}\ \exists$$

$$(\mathtt{E}\vee)\ \frac{\mathtt{E}(\alpha_i,\Pi),\Phi}{\mathtt{E}(\alpha_0\vee\alpha_1,\Pi),\Phi}\ \exists,\, i\in\{0,1\} \qquad (\mathtt{ER})\ \frac{\mathtt{E}(\alpha,\beta,\Pi),\Phi \quad | \quad \mathtt{E}(\beta,\mathtt{X}(\alpha\mathtt{R}\beta),\Pi),\Phi}{\mathtt{E}(\alpha\mathtt{R}\beta,\Pi),\Phi}\ \exists$$

$$(\mathtt{A}\wedge)\ \frac{\mathtt{A}(\alpha,\Sigma),\mathtt{A}(\beta,\Sigma),\Phi}{\mathtt{A}(\alpha\wedge\beta,\Sigma),\Phi} \qquad (\mathtt{A}\vee)\ \frac{\mathtt{A}(\alpha,\beta,\Sigma),\Phi}{\mathtt{A}(\alpha\vee\beta,\Sigma),\Phi} \qquad (\mathtt{ASt})\ \frac{\varphi,\Phi \quad | \quad \mathtt{A}\Sigma,\Phi}{\mathtt{A}(\varphi,\Sigma),\Phi}\ \exists$$

$$(\mathtt{AU})\ \frac{\psi,\Phi \quad | \quad \varphi,\mathtt{A}(\mathtt{X}(\varphi\mathtt{U}\psi),\Sigma),\Phi \quad | \quad \mathtt{A}\Sigma,\Phi}{\mathtt{A}(\varphi\mathtt{U}\psi,\Sigma),\Phi}\ \exists,\text{ if } \Sigma \text{ not } \mathtt{U}\text{-pure}$$

$$(\mathtt{A}\vec{\mathtt{U}})\ \frac{\psi_j,\Phi \quad | \quad \{\varphi_i \mid i\in I\},\mathtt{A}(\{\mathtt{X}(\varphi_i\mathtt{U}\psi_i)\mid i\in I\}),\Phi}{\mathtt{A}(\varphi_1\mathtt{U}\psi_1,\ldots,\varphi_n\mathtt{U}\psi_n),\Phi}\ \exists,\, j\in[n], I\subseteq[n]$$

$$(\mathtt{AR})\ \frac{\mathtt{A}(\beta,\Sigma),\mathtt{A}(\alpha,\mathtt{X}(\alpha\mathtt{R}\beta),\Sigma),\Phi}{\mathtt{A}(\alpha\mathtt{R}\beta,\Sigma),\Phi}$$

$$(\mathtt{X}_0)\ \frac{\mathtt{A}\Sigma_1,\ldots,\mathtt{A}\Sigma_m}{\mathtt{AX}\Sigma_1,\ldots,\mathtt{AX}\Sigma_m,\Lambda} \qquad (\mathtt{X}_1)\ \frac{\mathtt{E}\Pi_i,\mathtt{A}\Sigma_1,\ldots,\mathtt{A}\Sigma_m}{\mathtt{EX}\Pi_1,\ldots,\mathtt{EX}\Pi_n,\mathtt{AX}\Sigma_1,\ldots,\mathtt{AX}\Sigma_m,\Lambda}\ \forall,\, i\in[n]$$

Fig. 1. Rules for the CTL$^\sharp$ satisfiability game.

3 Playing Time

Configurations, Rules, Plays, and Winning Conditions. We present a game-theoretic characterisation of the satisfiability problem for CTL$^\sharp$. The game $\mathcal{G}(\vartheta)$ is played by two players $\exists$ and $\forall$ who want to show that ϑ is satisfiable, resp. is unsatisfiable. We fix a state formula $\vartheta \in$ CTL$^\sharp$ for the rest of this section.

A *block* is an element of $\{\mathtt{E},\mathtt{A}\} \times 2^{\mathsf{Cl}(\vartheta)}$ written $\mathtt{E}\Pi$ or $\mathtt{A}\Sigma$. They represent the state formulas $\mathtt{E}\bigwedge\Pi$ and $\mathtt{A}\bigvee\Sigma$. Conversely, we identify a state formula $Q\alpha$ with the block $Q\{\alpha\}$ for $Q\in\{\mathtt{A},\mathtt{E}\}$. A *configuration* is a set of state formulas and of blocks, for instance $\varphi_1,\ldots,\varphi_l,\mathtt{E}\Pi_1,\ldots,\mathtt{E}\Pi_n,\mathtt{A}\Sigma_1,\ldots,\Sigma_m$. The intended formula of such a configuration is the conjunction of its elements.

A formula set Σ is called $\mathtt{U}$-pure if it consists of formulas of the form $\varphi_1\mathtt{U}\varphi_2$ only. We write $\mathtt{X}\Sigma$ to denote the set $\{\mathtt{X}\psi \mid \psi\in\Sigma\}$ and equally for $\mathtt{X}\Pi$. A configuration Φ is *propositionally inconsistent* if there is a proposition q s.t. $\{q,\neg q\}\subseteq\Phi$.

The game $\mathcal{G}(\vartheta)$ starts in the initial configuration ϑ and proceeds according to the rules presented in Fig. 1. We write $[n]$ to denote $\{1,\ldots,n\}$. There are a few important comments to regard when reading the rules.

– They are to be read bottom-up, i.e. if the current configuration in a play is an instance of the pattern below then the player annotated to the right of a rule chooses one of the configurations on top to continue with. The respective player can choose from the alternatives which are separated by "|". Some rules are deterministic, i.e. no player is making a choice. The configuration on the top of a rule is called *premise* and that on the bottom *conclusion*.

- Formulas denoted $\varphi, \psi, \varphi_1, \ldots$ are state formulas according to the syntax of CTL*; formula denoted $\alpha, \beta, \alpha_1, \ldots$ are path (and therefore possibly also state) formulas. Λ always stands for a set of literals, Φ denotes an arbitrary set of blocks and state formulas, and Σ and Π denote a set of path formulas.
- As we identify the state formula $\mathtt{E}\alpha$ with the block $\mathtt{E}\{\alpha\}$—for instance—the rules $(\wedge)$ and $(\vee)$ can generate blocks.
- Although configurations and blocks are sets in the main, they are written as lists. However, a notation like $\mathtt{A}(\alpha \wedge \beta, \Sigma), \Phi$ implicitly states that $\alpha \wedge \beta \notin \Sigma$ and $\mathtt{A}(\alpha \wedge \beta, \Sigma) \notin \Phi$. Otherwise, a rule application could be repeated ad infinitum and hinders progress.

Note that in certain configurations several rules may apply, even rules for both players to perform choices. We therefore assume an arbitrary but fixed ordering on the rules which determines uniquely the rule that applies to a configuration. The exact ordering is irrelevant for the theory developed here.

In an application of a rule, the formula and the block which get transformed are called *principal formula* and *block*, respectively. Examples are the formula $\alpha \wedge \beta$ and the block $\mathtt{A}(\alpha \wedge \beta, \Sigma)$ in the instance of the rule $(\mathtt{A}\wedge)$ as shown in Fig. 1. In the rule $(\mathtt{A}\vec{\mathtt{U}})$ *all* formulas in the principal block are principal. The unaffected blocks and formulas are called *side formulas* taking blocks for formulas. Continuing the example, these are the formulas in Σ and in Φ.

A *play* is a possibly infinite sequence of configurations starting in the initial one and resulting from successive rule applications. Note that in every play, the intended formula of a configuration is in CTL$^\sharp$. Before we can define the winner of a play we need a technical definition capturing the unfulfilledness of least fixed point constructs.

Definition 1. A *component* of a configuration C is a state formula in C, an $\mathtt{A}$-block in C or a single formula inside an $\mathtt{E}$-block contained in C. Let $\Phi_0, \Phi_1, \ldots$ be an infinite play. The rules induce a connection relation on components of adjacent configurations in this play, obtained from the game rules in a straightforward way. A component C in Φ_i is connected to a component C' in Φ_{i+1}, written $\langle \Phi_i, C\rangle \rightsquigarrow \langle \Phi_{i+1}, C'\rangle$, if either

- C is not principal and $C = C'$, or
- C is principal in this rule application and gets transformed into C'.

Example 2. To illustrate the second item consider an instance of the rule $(\mathtt{ASt})$ as shown Fig. 1 for $\varphi = \mathtt{E}\alpha$. For the left alternative, $\mathtt{E}\alpha$ becomes part of the configuration both as a state formula and as a block $\mathtt{E}\{\alpha\}$. Therefore, $\langle \cdot, \mathtt{A}(\mathtt{E}\alpha, \Sigma)\rangle \rightsquigarrow \langle \cdot, \mathtt{E}\alpha\rangle$ and, if α is a state formula, $\langle \cdot, \mathtt{A}(\mathtt{E}\alpha, \Sigma)\rangle \rightsquigarrow \langle \cdot, \alpha\rangle$ hold. For the other alternative, we have $\langle \cdot, \mathtt{A}(\mathtt{E}\alpha, \Sigma)\rangle \rightsquigarrow \langle \cdot, \mathtt{A}\Sigma\rangle$. On the other hand, a $\mathtt{U}$- and an $\mathtt{R}$-formula can grow by unfolding these fixed points. For example, the instance of the rule $(\mathtt{EU})$ in Fig. 1 yields $\langle \cdot, \alpha\mathtt{U}\beta\rangle \rightsquigarrow \langle \cdot, \mathtt{X}(\alpha\mathtt{U}\beta)\rangle$.

The following lemma is not hard to see. Note that only the unfolding rules for $\mathtt{U}$- and $\mathtt{R}$-formulas create in some sense larger configurations, but they introduce an $\mathtt{X}$-operator which has to be dealt with before the respective formula can be unfolded again.

Lemma 3. *Every infinite play contains infinitely many applications of rules* $(\mathtt{X}_0)$ *or* $(\mathtt{X}_1)$.

Definition 4. A *thread* in $\Phi_0, \Phi_1, \ldots$ is a sequence $C_0, C_1, \ldots$ of components such that $\langle \Phi_i, C_i \rangle \rightsquigarrow \langle \Phi_{i+1}, C_{i+1} \rangle$ for all $i \in \mathbb{N}$. It is called a *bad thread* if either

– there is a $\varphi \mathtt{U} \psi \in \mathsf{Cl}(\vartheta)$ s.t. $C_i = \varphi \mathtt{U} \psi$ for infinitely many i, or
– there is a block $\mathtt{A}\Sigma$ with Σ being $\mathtt{U}$-pure, s.t. $C_i = \mathtt{A}\Sigma$ for infinitely many i.

In the first case we also speak of a bad $\mathtt{E}$-thread, in the second case of a bad $\mathtt{A}$-thread.

Hence, a play contains a *bad thread* if there is either a $\mathtt{U}$-formula inside some $\mathtt{E}$-blocks that regenerates itself infinitely often via the unfolding in rule ($\mathtt{EU}$), or there is an $\mathtt{A}$-block which contains no $\mathtt{R}$-formula that regenerates itself in a similar way along this play. Player $\forall$ *wins a play* $\pi = \Phi_0, \Phi_1, \ldots$ if
($\forall$-1) there is an $n \in \mathbb{N}$ s.t. Φ_n is propositionally inconsistent, or
($\forall$-2) there is an $n \in \mathbb{N}$ s.t. $\mathtt{A}\emptyset \in \Phi_n$, or
($\forall$-3) π contains a bad thread.
In all other cases, player $\exists$ wins the play.

Determinacy. An important game-theoretic concept is *determinacy* meaning that for every game exactly one of the players has a winning strategy. The games presented here are determined. The proof is relatively simple by appealing to known determinacy results about games in general. We only need to identify the winning plays as being of a certain type, namely being recognisable by a co-Büchi automaton.

Lemma 5. *For a bad thread $C_0, C_1, \ldots$ in $\Phi_0, \Phi_1, \ldots$ either*

– *there is a $k \in \mathbb{N}$ and $\varphi \mathtt{U} \psi \in \mathsf{Cl}(\vartheta)$ s.t. $C_i \in \{\varphi \mathtt{U} \psi, \mathtt{X}(\varphi \mathtt{U} \psi)\}$ for all $i \geq k$, or*
– *there is a $k \in \mathbb{N}$ and a $\mathtt{U}$-pure set $\Sigma \subseteq \mathsf{Cl}(\vartheta)$ s.t. $C_i \in \{\mathtt{A}\Sigma, \mathtt{A}(\mathtt{X}\Sigma)\}$ for all $i \geq k$.*

Proof. The case distinction follows Definition 4. For k we take one of the infinitely many values i mentioned in that definition. Finally, the game rules entail the properties along the corresponding suffix. □

Theorem 6. *The* CTL$^\sharp$ *satisfiability games are determined, i.e. for every ϑ, either $\exists$ or $\forall$ has a winning strategy for the game $\mathcal{G}(\vartheta)$.*

Proof. Following Lemma 5 and 3, the winning conditions can be represented as a co-Büchi condition and are therefore in the Borel hierarchy. The result then follows immediately from Martin's Theorem [12]. □

Soundness and Completeness. Due to lack of space we only sketch how one can prove that the games correctly characterise satisfiability in CTL$^\sharp$. It is possible to do this via explicit constructions of a model from a winning strategy for player $\exists$, etc. Instead, we appeal to a very similar system that is known to correctly characterise satisfiability for CTL* [4]. However, the syntactical restrictions of CTL* considered here supersede the distinction between traces and threads as exploited in [4].

Theorem 7. *Player $\exists$ has a winning strategy for the game $\mathcal{G}(\vartheta)$ iff ϑ is satisfiable.*

The games here differ from the system in [4] in the rules for universally path quantified blocks and in the winning conditions. There, the rules are simply dual to the ones for existentially path quantified blocks; in detail: rule $(\mathtt{AU})$ and $(\mathtt{A}\vec{\mathtt{U}})$ here replace the dual version of $(\mathtt{EU})$. Furthermore, in the system for full CTL*, a bad thread of $\mathtt{A}$-blocks must not contain any infinitely regenerating $\mathtt{R}$-formula. In order to prove soundness and completeness of the games for CTL$^\sharp$ it suffices to see that the CTL$^\sharp$ games essentially behave like the CTL* games when applied to a CTL$^\sharp$ formula. Now note that the defining property of being a CTL$^\sharp$ formula is having no genuine path formulas as arguments to a $\mathtt{U}$-formula inside an $\mathtt{A}$-block. State formulas, though, get removed from $\mathtt{A}$-blocks with rule $(\mathtt{ASt})$. This justifies rule $(\mathtt{AU})$. Furthermore, if a sequence of connected $\mathtt{A}$-blocks through a play contains no regenerating $\mathtt{R}$-formula then the set of formulas in those blocks must eventually become $\mathtt{U}$-pure because no $\mathtt{U}$-formula in there can spawn off anything that remains in this set. Then the unfolding of $\mathtt{U}$-formulas in a $\mathtt{U}$-pure set can be synchronised which justifies rule $(\mathtt{A}\vec{\mathtt{U}})$. Finally, a bad thread in the sense of Def. 4 is a bad trace in the system for CTL*, and vice versa.

4 Pruning Time

By Thm. 6 and 7, $\neg\vartheta$ is a tautology iff player $\forall$ has a winning strategy in the game $\mathcal{G}(\vartheta)$. In the next section we will present an axiomatisation for CTL$^\sharp$, and in this section we develop the necessary tools in order to prove completeness thereof. We consider $\forall$'s winning strategy as a tree, namely the tree of all plays which conform to this strategy. I.e. at every position in which player $\exists$ makes a choice or a deterministic rule applies all successors are preserved in the tree. At position in which player $\forall$ makes a choice only the choice prescribed by the strategy is preserved in the tree. Clearly, such a tree is in general infinite. We turn it into a finite tree which essentially is a finite derivation for $\neg\vartheta$ in the axiom system of the next section.

As this axiomatisation should also be sound it clearly does not suffice to truncate the tree at arbitrary positions. Instead, the resulting finite tree should satisfy the following properties: (1) leaves should be unsatisfiable; and (2) unsatisfiability should be preserved in the direction towards the root. This is enough to yield completeness since it constructs a proof for an unsatisfiable $\neg\vartheta$, i.e. a valid ϑ.

The principles used to achieve (1) and (2) are the following. At nodes which are inconsistent or contain $\mathtt{A}\emptyset$, it suffices to simply truncated the tree. This is possible on all plays that player $\forall$ wins with his winning conditions ($\forall$-1) or ($\forall$-2). The remaining paths in the tree contain bad threads. We use the principle of fixpoint strengthening in order to preserve satisfiability but disable infinite unfoldings of $\mathtt{U}$-operators. In essence, this principle forbids the unfolding of a (set of) $\mathtt{U}$-formulas in a certain context for the second time. Instead, the node becomes inconsistent and can be truncated as well.

An additional difficulty is the fact that this has to be done in the tree as a whole rather than on each branch separately—even though bad threads are properties of branches. Note that two branches with bad threads may have a common prefix but these threads may differ on that prefix. This is basically handled by a scheduling mechanism which strengthens least fixpoint formulas one-by-one.

$$(\mathtt{EU})^{\dagger}\ \frac{}{\mathtt{E}(\varphi\mathtt{U}_{\Pi\cup\Phi}\psi,\Pi),\Phi}\qquad\qquad (\mathtt{EU})^{*}\ \frac{\mathtt{E}(\beta,\Pi),\Phi\quad\mid\quad\mathtt{E}(\alpha,\mathtt{X}(\alpha\mathtt{U}_{\Gamma}\beta),\Pi),\Phi}{\mathtt{E}(\alpha\mathtt{U}_{\Gamma}\beta,\Pi),\Phi}$$

$$(\mathtt{EU})^{\natural}\ \frac{\mathtt{E}(\varphi\mathtt{U}\psi,\Pi),\Phi}{\mathtt{E}(\varphi\mathtt{U}_{\Gamma}\psi,\Pi),\Phi}\qquad\qquad (\mathtt{EU})^{\flat}\ \frac{\mathtt{E}(\beta,\Pi),\Phi\quad\mid\quad\mathtt{E}(\alpha,\mathtt{X}(\alpha\mathtt{U}_{\Pi\cup\Phi}\beta),\Pi),\Phi}{\mathtt{E}(\alpha\mathtt{U}\beta,\Pi),\Phi}$$

$$(\mathtt{A}\vec{\mathtt{U}})^{\dagger}\ \frac{}{\mathtt{A}(\varphi_1\mathtt{U}_{\Phi}\psi_1,\dots,\varphi_n\mathtt{U}_{\Phi}\psi_n),\Phi}\qquad\qquad (\mathtt{A}\vec{\mathtt{U}})^{\natural}\ \frac{\mathtt{A}(\varphi_1\mathtt{U}\psi_1,\dots,\varphi_n\mathtt{U}\psi_n),\Phi}{\mathtt{A}(\varphi_1\mathtt{U}_{\Gamma}\psi_1,\dots,\varphi_n\mathtt{U}_{\Gamma}\psi_n),\Phi}$$

$$(\mathtt{A}\vec{\mathtt{U}})^{*}\ \frac{\psi_j,\Phi\quad\mid\quad\varphi_1,\dots,\varphi_n,\mathtt{A}(\mathtt{X}(\varphi_1\mathtt{U}_{\Gamma}\psi_1),\dots,\mathtt{X}(\varphi_n\mathtt{U}_{\Gamma}\psi_n)),\Phi}{\mathtt{A}(\varphi_1\mathtt{U}_{\Gamma}\psi_1,\dots,\varphi_n\mathtt{U}_{\Gamma}\psi_n),\Phi}\ \exists j\in[n]$$

$$(\mathtt{A}\vec{\mathtt{U}})^{\flat}\ \frac{\psi_j,\Phi\quad\mid\quad\varphi_1,\dots,\varphi_n,\mathtt{A}(\mathtt{X}(\varphi_1\mathtt{U}_{\Phi}\psi_1),\dots,\mathtt{X}(\varphi_n\mathtt{U}_{\Phi}\psi_n)),\Phi}{\mathtt{A}(\varphi_1\mathtt{U}\psi_1,\dots,\varphi_n\mathtt{U}\psi_n),\Phi}\ \exists j\in[n]$$

Fig. 2. Rules for annotated U-formulas in E-blocks.

4.1 Annotations and Their Rules

For a set of formulas $\Gamma\subseteq\mathsf{Cl}(\vartheta)$ define $\varphi\mathtt{U}_{\Gamma}\psi := (\varphi\wedge\neg\bigwedge\Gamma)\mathtt{U}(\psi\wedge\neg\bigwedge\Gamma)$. The set Γ is called an *annotation* to the formula $\varphi\mathtt{U}\psi$. Note that annotating formulas can take us out of the CTL$^\sharp$ fragment. This, however, is just an observation and has no negative effect since its semantics is well-defined as a CTL*-formula.

The annotation is used to remember a set of side formulas. Informally, once the annotated formula occurs in a configuration with the same side formulas again, we like to truncate the play right after this repetition.

On an infinite play the U-formulas are eventually handled by the rules $(\mathtt{EU})$ and $(\mathtt{A}\vec{\mathtt{U}})$ for $I=[n]$ only. Both rules are extended to operate on annotated formulas, introducing four new rules for each occurrence of U-formulas inside E- or A-quantifiers: one rule to create an annotation, one to keep it through the usual unfolding, one to erase it for following different branches with different bad threads, and one to terminate the play. These new rules are shown in Fig. 2.

The annotation in rule $(\mathtt{A}\vec{\mathtt{U}})^{\flat}$ is placed on all formulas in the block simultaneously. None of the rules can change or remove the annotation of a formula in such a block without affecting the other formulas. Hence, we actually annotate the whole block rather than each formula. However, for simplicity we focus on the annotation of formulas.

Lemma 8. *The conclusion in rule* $(\mathtt{EU})^{\dagger}$ *is unsatisfiable, and for the rules* $(\mathtt{EU})^{\natural}$, $(\mathtt{EU})^{*}$ *and* $(\mathtt{EU})^{\flat}$ *we have that if all premises are unsatisfiable then so is the conclusion.*

Proof. We detail the proof for the most difficult rule only, that is for $(\mathtt{EU})^{\flat}$. Assume that there is a transition system $\mathcal{T}$ and a path π such that

$$\mathcal{T},\pi\models\alpha\mathtt{U}\beta\ \wedge\ \bigwedge\Pi\ \wedge\ \bigwedge\Phi. \tag{1}$$

Let $k\in\mathbb{N}$ be such that $\mathcal{T},\pi^k\models\beta$, and $\mathcal{T},\pi^i\models\alpha\wedge\neg\beta$ for all $i<k$. Among all such paths satisfying (1), we choose a path π with a minimal k-value. Suppose that none of

the premises is satisfiable. Thus, we have in particular

$$\mathcal{T}, \pi \not\models (\beta \vee (\alpha \wedge \mathtt{X}(\alpha \mathtt{U}_{\Pi \cup \Phi} \beta))) \ \wedge \ \bigwedge \Pi \ \wedge \ \bigwedge \Phi.$$

Then there is a $0 < \ell \leq k$ such that $\mathcal{T}, \pi^\ell \not\models \neg \bigwedge(\Pi \cup \Phi)$. Additionally, (1) yields $\mathcal{T}, \pi^\ell \models \alpha \mathtt{U} \beta$. Therefore, Eq. (1) holds for π^ℓ instead of π. But the k-value of π^ℓ is $k - \ell < k$. Thus, this is a contradiction to the minimality of k. $\square$

Lemma 9. *The conclusion in rule* $(\mathtt{A}\vec{\mathtt{U}})^\dagger$ *is unsatisfiable, and for the rules* $(\mathtt{A}\vec{\mathtt{U}})^\natural$, $(\mathtt{A}\vec{\mathtt{U}})^*$ *and* $(\mathtt{A}\vec{\mathtt{U}})^\flat$ *we have that if all premises are unsatisfiable then so is the conclusion.*

Proof. Consider the rule $(\mathtt{A}\vec{\mathtt{U}})^\flat$. Assume that

$$\psi_j \wedge \bigwedge \Phi \text{ is unsatisfiable for all } j \in [n], \tag{2}$$

and that

$$\mathcal{T}, s \models \mathtt{A}\left(\bigvee\nolimits_{i \in [n]} \varphi_i \mathtt{U} \psi_i\right) \wedge \bigwedge \Phi \tag{3}$$

holds for a transition system $\mathcal{T}$ and a state s in $\mathcal{T}$.

For any such pair $(\mathcal{T}, s)$ satisfying (3) we associate a tree with unordered children. The tree is the unwinding of $\mathcal{T}$ which begins at s and ends at a node t whenever there is an $i \in [n]$ such that $\mathcal{T}, t \models \psi_i$ and $\mathcal{T}, r \models \varphi_i$ for any node r along the path from s, including, to t, excluding. Since an associated tree does not show any infinite path—nevertheless the tree might be infinite—the strict subtree-order is well-founded on associated trees.

Now, let $\mathcal{T}$ and s be such that they satisfy (3) and that their associated tree is a minimal one w.r.t. the strict subtree-order among all associated trees which originate from pairs which satisfy (3). By (2) the associated tree cannot consist of its root only. For the sake of contradiction, assume that the pair of $\mathcal{T}$ and s does not model the right premise of $(\mathtt{A}\vec{\mathtt{U}})^\flat$. Then there is a state r different from s, which corresponds to a node in the associated tree, such that $\mathcal{T}, r \not\models \neg \bigwedge \Phi$. However, the subtree at r is the associate tree of $\mathcal{T}$ and r. But this situation contradicts the choice of $\mathcal{T}$ and s.

The argument for the other rules are simpler. $\square$

Remark 10. Although the rules in Fig. 2 are sound w.r.t. unsatisfiability they are *not* invertible in general. Consider the rule $(\mathtt{EU})^\natural$: the configuration $\mathtt{E}(p\mathtt{U}_{\{p\}}(p \wedge q))$ is unsatisfiable whereas $\mathtt{E}(p\mathtt{U}(p \wedge q))$ is satisfiable. Therefore, these rules are unsuitable for an incorporation into the game defined in Sect. 3 in the first place.

4.2 Truncating Infinite Trees

The following constructions consider (labelled) trees. To simplify the presentation we introduce some notations. For a tree T and nodes u, v in T we write $u <_T v$ iff u is a proper ancestor of v, $u \leq_T v$ iff $u <_T v$ or $u = v$, $T|_v$ for the subtree located at v such that v is the root of $T|_v$, and u is a *child* of v (v is *parent* of u, resp.) iff $v <_T u$ and $v <_T w <_T u$ for no node w in T. A path in T is a (finite or infinite) sequence such that any node in the sequence is a parent of the succeeding node if the latter exists. A branch is a path with begins at the root. Since $<_T$ is well-founded—but not a linear order in general—, $\min_T(V)$ denotes the set of minimal nodes w.r.t. $<_T$ in a set V of nodes.

For the remaining section assume that $\neg\vartheta$ is a tautology. Hence, player $\forall$ has a winning strategy for the game $\mathcal{G}(\vartheta)$, cf. Thm. 6 and 7. From now on, formulas and sets of formulas are assumed to be in or subsets of $\mathsf{Cl}(\vartheta)$.

Definition 11. We say that a tree T is a *(ϑ-)tree which follows a set of rules R* iff each node is labelled with a configuration for ϑ, the root is labelled with ϑ, and for each node v one of the following items holds.

- v is a leaf and the node is propositionally inconsistent or contains $\mathtt{A}\emptyset$.
- v has exactly one child, say w, such that v is the conclusion and w the premise of the same instance of the rule $(\mathtt{X}_0)$ or $(\mathtt{X}_1)$.
- For a rule in $R \setminus \{(\mathtt{X}_0), (\mathtt{X}_1)\}$ and for a principal block and formula(s), the set of children is the set of possible successor configurations which player $\exists$ can choose with this rule, principal block and formula(s). For instance, a node for the rule $(\mathtt{A}\vec{\mathtt{U}})$ has exactly $n + 2^n$ children if n is the number of $\mathtt{U}$-formulas in the principal $\mathtt{A}$-block: The left hand of "|" admits n possibilities and the right hand 2^n.

Note that in such a tree any inner node uniquely determines the rule which was used in the justification for the second and the last item.

Lemma 12. *There is a ϑ-tree which follows the rules in Fig. 1.*

Proof. The winning strategy of player $\forall$ is taken for a tree. However, not all possible moves of player $\exists$ need to be considered. For the last item in Def. 11 it suffices to consider just one rule, principal block and formula(s) on $\exists$'s turn. Moreover, on every branch the first node is changed into a leaf iff the node is propositionally inconsistent or contains $\mathtt{A}\emptyset$. The obtained tree follows the said rules. ☐

We fix such a tree and call it T_ϑ. The label of a node v is written as $\ell(v)$. We may take the node for its label. As each branch in T_ϑ forms a game in the sense of Sect. 3—at least a prefix of a game for which the winner is already determined—, we may use the game-theoretic notations for the tree as well. For instance, every infinite branch in T_ϑ contains a bad thread.

To handle repetitions in an infinite branch of T_ϑ we set

$$\begin{aligned}\mathsf{Rep}(\vartheta) \;&:=\; \{\Sigma \mid \Sigma \in 2^{\mathsf{Cl}(\vartheta)} \text{ and } \Sigma \text{ is } \mathtt{U}\text{-pure}\}\\ &\;\cup\; \{\varphi \mid \varphi \in \mathsf{Cl}(\vartheta) \text{ and } \varphi \text{ is a } \mathtt{U}\text{-formula}\} \times 2^{\mathsf{Cl}(\vartheta)}.\end{aligned}$$

Definition 13. Let $\rho \in \mathsf{Rep}(\vartheta)$. A node v in the tree T_ϑ is a *repeated node for ρ* iff there is a path $v_1, \ldots, v_K$ in T_ϑ for some $K > 1$ such that $v_1 = v$, $\ell(v_1) = \ell(v_K)$ and one of the following items holds.

- $\rho \in 2^{\mathsf{Cl}(\vartheta)}$, v is the conclusion of an instance of $(\mathtt{A}\vec{\mathtt{U}})$, $\mathtt{A}\rho$ is principal for that application, and there is a sequence $\Sigma_1, \ldots, \Sigma_K$ of sets of formulas such that $\Sigma_1 = \Sigma_K = \rho$ and $\langle \ell(v_i), \mathtt{A}(\Sigma_i)\rangle \rightsquigarrow \langle \ell(v_{i+1}), \mathtt{A}(\Sigma_{i+1})\rangle$ for all $i \in [K-1]$.

– $\rho = (\varphi, \Pi)$ for $\{\varphi\} \cup \Pi \subseteq \mathsf{Cl}(\vartheta)$, v is the conclusion of an instance of $(\mathtt{EU})$, φ and $\mathtt{E}(\varphi, \Pi)$ are principals for that instance, and there are a sequence $\varphi_1, \dots, \varphi_K$ of formulas and a sequence $\Pi_1, \dots, \Pi_K$ of sets of formulas such that $\varphi_1 = \varphi_K = \varphi$, $\Pi_1 = \Pi_K = \Pi$, and $\langle \ell(v_i), \mathtt{E}(\varphi_i, \Pi_i)\rangle \rightsquigarrow \langle \ell(v_{i+1}), \mathtt{E}(\varphi_{i+1}, \Pi_{i+1})\rangle$ for all $i \in [K-1]$.

A node v_K is called *repeating node for v and ρ.*

Lemma 14. *Let $(v_i)_{i\in\mathbb{N}}$ be an infinite branch in T_ϑ. If the play $(\ell(v_i))_{i\in\mathbb{N}}$ contains a bad thread then there is a $\rho \in \mathsf{Rep}(\vartheta)$ such that for every i there are $j^-, j^+ \in \mathbb{N}$ with $i < j^- < j^+$, v_{j^-} is a repeated node for ρ, and v_{j^+} is their repeating node.*

Proof. Let $(C_i)_{i\in\mathbb{N}}$ be a bad thread in $(\ell(v_i))_{i\in\mathbb{N}}$. Lem. 5 yields two possibilities. We consider the first case only—the other is very similar. So, let k, φ, ψ as written in that alternative. Let $i \in \mathbb{N}$ be given. At least one of the rules $(\mathtt{X}_0)$ and $(\mathtt{X}_1)$ is applied infinitely often, cf. Lem. 3. Therefore and as only the rule $(\mathtt{EU})$ can modify a $\mathtt{U}$-formula, the rule $(\mathtt{EU})$ is also applied infinitely often with $\varphi\mathtt{U}\psi$ as principal formula. Because the amount of different instances of the rule $(\mathtt{EU})$ is finite, there are j^- and j^+ such that $\max(k, i) < j^- < j^+$, v_{j^-} is a repeated node for $\varphi\mathtt{U}\psi$, and v_{j^+} is their repeating node. Indeed, $(C_{i+j^--1})_{i\in[j^+-j^-+1]}$ is the φ-sequence as required in Def. 13. To this end, the i^{th} element sequence of the Π-sequence is the set of side formula in the $\mathtt{E}$-block which hosts C_{i+j^--1}. $\square$

The truncation of T_ϑ to a finite tree is realized by the operation $\cdot \Downarrow_\vartheta \cdot$. To this end, the operation considers the elements of $\mathsf{Rep}(\vartheta)$ in some order. So, let $(\mathsf{Rep}_i)_{i\in[|\mathsf{Rep}(\vartheta)|]}$ be an enumeration of $\mathsf{Rep}(\vartheta)$.

Definition 15 **($\cdot \Downarrow_\vartheta \cdot$).** For a tree S and $i \in \mathbb{N}$, the application $S \Downarrow_\vartheta i$ returns a tree. If S is finite or $i > |\mathsf{Rep}(\vartheta)|$, $S \Downarrow_\vartheta i := S$. Otherwise, let

$$V^- := \min_S\{v \mid v \text{ is a repeated node for } \mathsf{Rep}_i \text{ in } S\},$$
$$V^+_{v^-} := \min_S\{v' \mid v' \text{ is a repeating node for } v^- \text{ and } \mathsf{Rep}_i \text{ in } S\}, \text{ and}$$
$$V^\perp := \min_S\{v \mid v \text{ and } v^+ \text{ are } \leq_S\text{-incomparable for all } v^- \in V^- \text{ and } v^+ \in V^+_{v^-}\}$$

where $v^- \in V^-$. For every $v \in V^\perp$ the operation replaces $S|_v$ by $(S|_v) \Downarrow_\vartheta (i+1)$. And for every $v \in V^-$ the operation annotates formulas and truncates subtree in S_v depending on Rep_i. If Rep_i is a set of $\mathtt{U}$-pure formulas $S \Downarrow_\vartheta i$ does the following. By definition, v is the conclusion of the rule $(\mathtt{A}\vec{\mathtt{U}})$ such that Rep_i is principal. The said rule got replaced by an instance of the rule $(\mathtt{A}\vec{\mathtt{U}})^\flat$. As this rule annotates formulas the operation proceeds away from the root as long as the current node is a proper ancestor of an $v' \in V^+_v$. Along this traversal, the rules got adjusted to the annotation. In particular, if the rule is $(\mathtt{A}\vec{\mathtt{U}})$ and the annotated formulas are principal, the rule got replaced by $(\mathtt{A}\vec{\mathtt{U}})^*$. When the traversal reaches an $v' \in V^+_v$ the node is replaced by the rule $(\mathtt{A}\vec{\mathtt{U}})^\dagger$. And as soon as the current node is not an ancestor of any $v' \in V^+_v$, the operation inserts the rule $(\mathtt{A}\vec{\mathtt{U}})^\natural$ to got rid of the annotation and skips the remaining subtree. The procedure for the other case—that is, Rep_i is a pair—is similar but replaces instances of the rule $(\mathtt{EU})$ by $(\mathtt{EU})^\flat$, $(\mathtt{EU})^*$, $(\mathtt{EU})^\natural$ and $(\mathtt{EU})^\dagger$. Since the set V^- is defined in terms of $\min_S$, the adjustments for the annotations do not interfere for different elements in V^-. This completes the description of $\cdot \Downarrow_\vartheta \cdot$.

Theorem 16. *$T_\vartheta \Downarrow_\vartheta 1$ is a finite ϑ-tree which follows the rules in Fig. 1 and 2.*

Proof. Each call $S \Downarrow_\vartheta i$ returns a tree such that its root and S's root share the same label. Therefore by construction, the tree $T_\vartheta \Downarrow_\vartheta 1$ follows the rules. If $\mathsf{Rep}(\vartheta) = \emptyset$ then $T_\vartheta \Downarrow_\vartheta 1 = T_\vartheta$ and by Lem. 14 the tree does not contain any bad thread. Since player $\forall$ wins $\mathcal{G}(\vartheta)$, T_ϑ must be finite. Now, suppose $\mathsf{Rep}(\vartheta) \neq \emptyset$. For the sake of contradiction, assume that $T_\vartheta \Downarrow_\vartheta 1$ is infinite. Since the tree is finitely branching, König's Lemma yields an infinite branch. The execution of $T_\vartheta \Downarrow_\vartheta 1$ leads to at most $|\mathsf{Rep}(\vartheta)| + 1$ nested invocations at a time. For each branch and each invocation the operation $\cdot \Downarrow_\vartheta \cdot$ inserts at most one node, namely the premise to the rule $(\mathtt{A}\vec{\mathtt{U}})^\natural$ or $(\mathtt{EU})^\natural$. Therefore, T_ϑ has an infinite branch. For simplicity we shall neither count nor name these additional nodes—hence the infinite branches in T_ϑ and in $T_\vartheta \Downarrow_\vartheta 1$ are identically equal. By definition, this branch contains a bad thread. Lem. 14 names a $\rho \in \mathsf{Rep}(\vartheta)$. Let i be such that $\mathsf{Rep}_i = \rho$. Since the branch is infinite, there was an invocation $T_\vartheta|_v \Downarrow_\vartheta i$ such that v lies on the infinite branch. Additionally, the same application of Lem. 14 yields two nodes v^- and v^+ on the infinite branch such that $v <_{T_\vartheta} v^- <_{T_\vartheta} v^+$, v^- is a repeated node for ρ, and v^+ is their repeating node. Among all such pairs (v^-, v^+) we minimize v^- and then v^+. Therefore, in the invocation of $T_\vartheta|_v \Downarrow_\vartheta i$ we have that $v^- \in V^-$ and $v^+ \in V^+_{v^-}$. The algorithm truncates the tree at v^+. This is a contradiction to the assumption that the infinite path in T_ϑ passes v^+. □

5 Proving Time

We present a Hilbert-style axiomatisation of $\neg$CTL$^\sharp$ and prove it to be sound and complete. An axiomatisation is a set of axioms and a set of rules. The axioms and rules may contain formula variables. A *proof* for some formula φ is a finite sequence $\varphi_0, \dots, \varphi_n$ s.t. $\varphi = \varphi_n$ and for all $i = 0, \dots, n$ we have: φ_i is either an instance of an axiom or follows from some $\varphi_0, \dots, \varphi_{i-1}$ via an instantiation of one of the rules. We write $\vdash \varphi$ to denote that φ is provable.

The axiomatisation is derived from the satisfiability game rules in Fig. 1 and the amended rules in Fig. 2 in the following way. Take a rule in which player $\exists$ makes a choice among premisses $P_1, \dots, P_n$ from a conclusion C. Then $\neg C$ should be provable from $\neg P_1, \dots, \neg P_n$. The axioms and rules are presented in Fig. 3. All formula variables α, β and γ range over arbitrary CTL*-formula and φ, χ, and ψ over CTL*-state formula. The axiom (Ax-1) can be made finite using a textbook-like axiomatisation of propositional logic where path-quantified formulas are taken for propositions in the purely propositional logic.

Using the previous sections, soundness and completeness of this axiomatisation is relatively easy to establish.

Theorem 17 (Soundness). *For all $\varphi \in$ CTL*: if $\vdash \varphi$ then $\models \varphi$.*

Proof. By induction on the length of a proof. One easily establishes that all axioms are valid and all the rules preserves validity. In particular, the rules (Ru-3) and (Ru-4) are

(Ax-1) All substitution instances of propositional tautologies
(Ax-2) $\vdash \mathtt{E}(\alpha \vee \beta) \leftrightarrow \mathtt{E}\alpha \vee \mathtt{E}\beta$ (Ax-3) $\vdash \mathtt{E}(\varphi \wedge \alpha) \leftrightarrow \varphi \wedge \mathtt{E}\alpha$
(Ax-4) $\vdash \alpha\mathtt{U}\beta \leftrightarrow \beta \vee (\alpha \wedge \mathtt{X}(\alpha\mathtt{U}\beta))$ (Ax-5) $\vdash (\alpha \to \beta) \to (\mathtt{E}\alpha \to \mathtt{E}\beta)$
(Ax-6) $\vdash \mathtt{X}(\alpha \vee \beta) \leftrightarrow \mathtt{X}\alpha \vee \mathtt{X}\beta$ (Ax-7) $\vdash \mathtt{Ett}$
(Ax-8) $\vdash \mathtt{EXtt}$ (Ax-9) $\vdash \mathtt{EXE}\alpha \leftrightarrow \mathtt{EX}\alpha$
(Ax-10) $\vdash (\alpha \wedge \gamma)\mathtt{U}(\beta \wedge \gamma) \to \alpha\mathtt{U}\beta$ (Ru-1) If $\vdash \alpha \to \beta$ and $\vdash \alpha$ then $\vdash \beta$
(Ru-2) If $\vdash \mathtt{E}\alpha \to \mathtt{E}\beta$ then $\vdash \mathtt{EX}\alpha \to \mathtt{EX}\beta$
(Ru-3) If $\vdash \mathtt{A}((\gamma \vee \beta) \wedge (\gamma \vee \alpha \vee \mathtt{X}((\alpha \vee \neg\gamma)\mathtt{R}(\beta \vee \neg\gamma))))$ then $\vdash \mathtt{A}(\gamma \vee (\alpha\mathtt{R}\beta))$
(Ru-4) $\begin{cases} \text{If} \vdash \chi \to (\bigwedge_{j\in[n]} \psi_j \wedge (\bigvee_{j\in[n]} \varphi_j \vee \mathtt{E}(\bigwedge_{j\in[n]} \mathtt{X}((\varphi_j \vee \chi)\mathtt{R}(\psi_j \vee \chi)))) \\ \text{then} \vdash \chi \to \mathtt{E}(\bigwedge_{j\in[n]} \varphi_j \mathtt{R} \psi_j) \end{cases}$

Fig. 3. Axioms and rules of the $\neg$CTL$^\sharp$ axiomatisation.

the negations of the rules $(\mathtt{EU})^\flat$ and $(\mathtt{A}\vec{\mathtt{U}})^\flat$ in the main. Indeed, the proofs of Lem. 8 and 9 also hold for arbitrary formulas in CTL* as long as the configuration is a state formula. □

Lemma 18. – *If Φ is propositionally inconsistent or contains $\mathtt{A}\emptyset$ then $\vdash \neg \bigwedge \Phi$.*
– *If Φ' is a premise and Φ the conclusion of $(\mathtt{X}_0)$ or $(\mathtt{X}_1)$, then $\vdash \neg \bigwedge \Phi'$ implies $\vdash \neg \bigwedge \Phi$.*
– *Let R be a rule in Fig. 1 or 2 apart from $(\mathtt{X}_0)$ and $(\mathtt{X}_1)$. For a fixed principal block and fixed principal formula(s), let $\Phi_1, \ldots \Phi_n$ be all possible premises to a conclusion Φ for the rule R. If $\vdash \neg \bigwedge \Phi_i$ for all $i \in [n]$ then we have $\vdash \neg \bigwedge \Phi$.*

Proof. The argument is mainly straightforward. For the rules of the satisfiability game it suffices to show that the conclusion implies the disjunction of the premises. Exceptions are $(\mathtt{X}_0)$, $(\mathtt{X}_1)$, $(\mathtt{EU})^\flat$ and $(\mathtt{A}\vec{\mathtt{U}})^\flat$. These rules can be proven sound by (Ru-2), (Ru-3) and (Ru-4). In some cases it is necessary to move side formulas first into and later out of the principal block. □

Theorem 19 (Completeness). *For all $\varphi \in \neg$CTL$^\sharp$: if $\models \varphi$ then $\vdash \varphi$.*

Proof. Suppose $\models \varphi$, i.e. $\neg\varphi$ is unsatisfiable. According to Thm. 7, player $\forall$ has a winning strategy for the game $\mathcal{G}(\neg\varphi)$. Thanks to Thm. 16 there is a $\neg\varphi$-tree T which follows the rules of Fig. 1 and 2 and whose root is $\neg\varphi$. By induction on the tree we can construct a proof for φ using Lemma 18. □

Altogether, the axiomatisation is sound and complete for $\neg$CTL$^\sharp$, and hence also for its fragments CTL and CTL$^+$.

6 Conclusion

A task for further work is obviously to extend these techniques to even larger fragments, let alone CTL* itself. It remains to be seen whether the syntactic restriction in CTL$^\sharp$, namely the fact that $\mathtt{U}$-formulas inside of $\mathtt{A}$-quantifiers must have state formulas as arguments, can be relaxed. If these are path formulas to some extend then a connected sequence of A-blocks which does not contain a regenerating R-formula need not become U-pure eventually. The problem is then to find a logical principle which is sound w.r.t. unsatisfiability and which allows one to strengthen such non-U-pure formula sets in order to become unsatisfiable after too many unwindings.

Another problem in this context is the fact that Lemma 8 cannot be made to work for universally path quantified formulas, i.e. by replacing each E in the corresponding rules with A. The reader is invited to attempt to prove the resulting statement. The problem essentially is that path formulas may hold somewhere in a path starting in a state but not in other paths starting in the same state. This, however, is true for state formulas.

References

1. E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computer and System Sciences*, 30:1–24, 1985.
2. E. A. Emerson and J. Y. Halpern. “Sometimes” and “not never” revisited: On branching versus linear time temporal logic. *Journal of the ACM*, 33(1):151–178, 1986.
3. E. A. Emerson and C. S. Jutla. The complexity of tree automata and logics of programs. *SIAM Journal on Computing*, 29(1):132–158, 2000.
4. O. Friedmann, M. Latte, and M. Lange. A decision procedure for CTL* based on tableaux and automata. In *Proc. 5th Int. Joint Conf. on Automated Reasoning, IJCAR'10*, volume 6173 of *LNCS*, pages 331–345. Springer, 2010.
5. D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. The temporal analysis of fairness. In *Proc. 7th Symp. on Principles of Programming Languages, POPL'80*, pages 163–173. ACM, 1980.
6. J. Johannsen and M. Lange. CTL$^+$ is complete for double exponential time. In *Proc. 30th Int. Coll. on Automata, Logics and Programming, ICALP'03*, volume 2719 of *LNCS*, pages 767–775. Springer, 2003.
7. F. Kröger. *Temporal Logic of Programs*. Springer, 1987.
8. F. Kröger and S. Merz. *Temporal Logic and State Systems*. Texts in Theoretical Computer Science. Springer, 2008.
9. M. Lange and C. Stirling. Focus games for satisfiability and completeness of temporal logic. In *Proc. 16th Symp. on Logic in Computer Science, LICS'01*, Boston, MA, USA, 2001. IEEE.
10. O. Lichtenstein and A. Pnueli. Propositional temporal logics: Decidability and completeness. *Logic Journal of the IGPL*, 8(1):55–85, 2000.
11. Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems Specification*. Springer, 1992.
12. D. A. Martin. Borel determinacy. *Ann. Math.*, 102:363–371, 1975.
13. W. Penczek. Branching time and partial order in temporal logics. In L. Bolc and A. Szałas, editors, *Time and Logic – A Computational Approach*, pages 179–228. UCL Press, London, 1995.
14. A. N. Prior. *Time and modality*. Oxford University Press, Oxford, UK, 1957.
15. M. Reynolds. An axiomatization of full computation tree logic. *Journal of Symbolic Logic*, 66(3):1011–1057, 2001.
16. M. Reynolds. An axiomatization of PCTL*. *Information and Computation*, 201(1):72–119, 2005.
17. M. Reynolds. A tableau for bundled CTL*. *Journal of Logic and Computation*, 17(1):117–132, 2007.
18. A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *Journal of the Association for Computing Machinery*, 32(3):733–749, 1985.