PSPACE-completeness of LTL/CTL* model checking

Peter Lohmann

April 10, 2007

Abstract

This paper will give a proof for the PSPACE-completeness of LTL-satisfiability and for the PSPACE-completeness of the problem of determining if a given LTL/CTL* formula is true in a given system. CTL*, a very expressive temporal logic, combines and extends both LTL and CTL and is used in model checking.

The fact that CTL* model checking is PSPACE-complete was shown by Clarke, Emerson and Sistla in 1986 but most of the work had already been done by Sistla and Clarke in 1985.

Before proving the main results the basics of Kripke structures and temporal formulas will be explained shortly as we need them later on.

This paper was written as a Studienarbeit at the Institute of Theoretical Computer Science of the Leibniz University Hannover, Germany.

I want to thank Thomas Zeume for his advice on the first draft of this paper.

Contents

1 Introduction

Model checking is the problem of *proving* that a given system (computer program, network protocol, hardware design etc.) fulfills or does not fulfill a given specification. If the latter is the case a counter example is given which violates the specification. The system in question is hereby modeled by a *Kripke structure*, a kind of finite state machine, and the specification is represented by a *temporal logic formula.* This formula may contain specific operators and quantifiers to model the specifications for the dynamic behaviour of the system.

Different temporal logics have been invented – each with its own strengths and weaknesses. Two common logics are LTL (Linear Temporal Logic) and CTL (Computation Tree Logic). This paper will examine the complexities of LTL and of CTL* (Computation Tree Logic*), a quite powerful temporal logic combining both the expresiveness of LTL and CTL in a single model.

We will prove that model checking for LTL is PSPACE-complete which was first shown by Sistla and Clarke [SC85]. In fact they even showed that the problem of determining whether a Kripke structure fulfilling a given formula exists is also PSPACE-complete. This we will prove, too. From there it is only a small step to see that model checking for CTL* is also PSPACE-complete, as shown by Clarke, Emerson and Sistla [CES86].

The remainder of this first chapter will introduce the basics of Kripke structures and temporal logic formulas.

1.1 Preliminaries

Definition 1.1.
A *Kripke structure* is a triple $M = (S, R, P)$ where S is the finite set of states, $R \subseteq S \times S$ is the total transition relation (*total* meaning that $\forall s \in S : \exists s' \in S$ such that $(s, s') \in R$) and $P : S \to 2^{AP}$ is the label function (AP is the finite set of atomic propositions).

The intuitive function is that the system is in exactly one state $s \in S$ in every moment and can step to all states $s' \in S$ for which $(s, s') \in R$. P is needed because Kripke structures are not connected to temporal logic formulas directly with the states but via atomic propositions. $P(s)$ is the set of atomic propositions which are true in state s.

These atomic propositions will now be used in the definition of temporal logic formulas of which there are two different kinds, namely *temporal state formulas* and *temporal path formulas*.

Definition 1.2.
A *temporal state formula* is inductively defined as follows:

- If $f \in AP$, then f is a state formula.
- If f and g are state formulas, then $\neg f$, $f \wedge g$, $f \vee g$ and $f \to g$ are state formulas.
- If f is a path formula, then $\mathbf{A}f$ and $\mathbf{E}f$ are state formulas.

Definition 1.3.
A *temporal path formula* is inductively defined as follows:

- If f is a state formula, then f is a path formula.
- If f and g are path formulas, then $\neg f$, $f \wedge g$, $f \vee g$, $f \to g$, $\mathbf{X}f$, $\mathbf{F}f$, $\mathbf{G}f$ and $f\mathbf{U}g$ are path formulas.

The letters stand for the following:

- **A** means "on **a**ll paths"
- **E** means "a path **e**xists"

- **X** means "in the ne**x**t state on the path"
- **F** means "eventually in a state along the path"
- **G** means "**g**lobally in all states along the path"
- **U** means "**u**ntil"

Now we can define CTL*:

Definition 1.4.
CTL* (Computation Tree Logic*) is the set of all temporal state formulas.

For a Kripke structure $M = (S, R, P)$, a state $s \in S$, a sequence of states $u = (u_0, u_1, u_2, \dots) \in S^{\mathbb{N}}$ with $(u_k, u_{k+1}) \in R \; \forall k \geq 0$ (such a u is called a *path* in M), a temporal state formula f and a temporal path formula g we write

$M, s \models f$	for	"the Kripke structure M fulfills the formula f in state s"
	or	"f is true in state s in M"
and $M, u \models g$	for	"the Kripke structure M fulfills the formula g along the path u"
	or	"g is true along path u in M".

If $u = (u_0, u_1, u_2, \dots)$ we write u^k for the suffix $(u_k, u_{k+1}, \dots)$.

Now that we know the syntax of temporal logic formulas we can define their semantics and hereby define the truth of a formula in a Kripke structure.

Definition 1.5.
The truth of a formula in a Kripke structure is inductively defined as follows:
(let f_1 and f_2 be state formulas and g_1 and g_2 path formulas; and let $s \in S$ and $u = (u_0, u_1, \dots)$ a path in M)

$$\begin{array}{lcl}
M, s \models p \; (p \in AP) & \Leftrightarrow & p \in P(s) \\
M, s \models \neg f_1 & \Leftrightarrow & M, s \not\models f_1 \\
M, s \models f_1 \wedge f_2 & \Leftrightarrow & M, s \models f_1 \text{ and } M, s \models f_2 \\
M, s \models \mathbf{E} g_1 & \Leftrightarrow & \exists \text{ path } v = (v_0 = s, v_1, v_2, \dots) \text{ in } M : M, v \models g_1 \\
M, u \models f_1 & \Leftrightarrow & M, u_0 \models f_1 \\
M, u \models \neg g_1 & \Leftrightarrow & M, u \not\models g_1 \\
M, u \models g_1 \wedge g_2 & \Leftrightarrow & M, u \models g_1 \text{ and } M, u \models g_2 \\
M, u \models \mathbf{X} g_1 & \Leftrightarrow & M, u^1 \models g_1 \\
M, u \models g_1 \mathbf{U} g_2 & \Leftrightarrow & \exists k \geq 0 : \left[M, u^k \models g_2 \;\wedge\; \forall 0 \leq j < k : M, u^j \models g_1 \right]
\end{array}$$

The remaining types of formulas are only abbreviations and are defined as follows:

$$\begin{array}{rcl}
f \vee g & \equiv & \neg(\neg f \wedge \neg g) \\
f \rightarrow g & \equiv & \neg f \vee g \\
\mathbf{A} f & \equiv & \neg \mathbf{E} \neg f \\
\mathbf{F} f & \equiv & \mathit{true} \mathbf{U} f \\
\mathbf{G} f & \equiv & \neg \mathbf{F} \neg f
\end{array}$$

Sometimes we are not only interested in whether a given formula is true in a given Kripke structure but we want to know if there is a Kripke structure at all in which the formula is true:

Definition 1.6.
A CTL* formula f is *satisfiable* $:\Leftrightarrow \exists M = (S, R, P) \wedge s_0 \in S$ such that $M, s_0 \models f$.

CTL* has an important subset:

Definition 1.7.
LTL (Linear Temporal Logic) is the subset of CTL* which only contains the formulas of the form $\mathbf{E}f$ where f is a path formula with only atomic propositions as state subformulas.

Or in other w: LTL is the set of all formulas $\mathbf{E}f$ where f is a *LTL path formula.* A LTL path formula is inductively defined as follows:

- If f is an atomic proposition, then f is a path formula.
- If f and g are path formulas, then $\neg f$, $f \wedge g$, $f \vee g$, $f \rightarrow g$, $\mathbf{X}f$, $\mathbf{F}f$, $\mathbf{G}f$ and $f\mathbf{U}g$ are path formulas.

We will now leave the basics of Kripke structures and temporal formulas.
For a wider introduction into Kripke structures and temporal formulas and for some examples the reader may refer to [CGP99, pp. 13-33].

2 PSPACE-completeness of LTL/CTL* model checking and LTL-satisfiability

The following three problems are to be examined:

Definition 2.1.
LTL-satisfiability $:= \{f \in \text{LTL} \mid f \text{ is satisfiable}\}$
LTL-truth $:= \{\langle M, s_0, f\rangle \mid M$ is a Kripke structure, s_0 is a state in M, $f \in \text{LTL}$, $M, s_0 \models f\}$
CTL*-truth $:= \{\langle M, s_0, f\rangle \mid M$ is a Kripke structure, s_0 is a state in M, $f \in \text{CTL}^*$, $M, s_0 \models f\}$

We will now show that LTL-satisfiability, LTL-truth and CTL*-truth are all PSPACE-complete. Therefor we will first show that LTL-truth is polynomial time reducible to LTL-satisfiability. Trivially LTL-truth is reducible to CTL*-truth (the reduction function is the identity). Then we will show that LTL-truth is PSPACE-hard. From this it follows by the above reductions that the other two problems are also PSPACE-hard. Then we will show that LTL-satisfiability $\in$ PSPACE and from this it follows (again by the above reduction) that LTL-truth $\in$ PSPACE. Finally we will show that CTL*-truth $\in$ PSPACE. Therefor we will use the former result that LTL-truth $\in$ PSPACE.

Note that *CTL*-satisfiability* $:= \{f \in \text{CTL}^* \mid f \text{ is satisfiable}\} \notin$ PSPACE but in fact 2EXPTIME-complete as shown by Emerson and Lei.

2.1 LTL-truth is reducible to LTL-satisfiability

Let $M = (S, R, P)$ be a Kripke structure, let $s_0 \in S$ and let $\mathbf{E}f \in \text{LTL}$. The idea of the reduction function is to model the behaviour of M by an appropriate formula f_M, model the beginning state $s_0 \in S$ by an atomic proposition p_{s_0} and then conjunct f, f_M and p_{s_0}.

Let AP be the set of all atomic propositions appearing in f and let $AP_S := \{p_s | s \in S\}$ and let w.l.o.g. $AP \cap AP_S = \emptyset$, i.e. AP_S contains one new atomic proposition for each state in S.

For a given state $s \in S$ we will now construct a formula f_s that ensures

- that p_s is true,
- that for no $s' \in S \backslash \{s\}$ $p_{s'}$ is true (meaning that at a given point in time the system is in state s and in no other state),
- that exactly the atomic propositions in $P(s)$ are true
- and that in the next state a $p_{s''}$ will be true for a $s'' \in S$ such that $(s, s'') \in R$ (meaning that the state will change according to the transition relation R).

Then the formula f_M is simply the globally quantified disjunction of all f_s $\forall s \in S$ (meaning that at every point in time the system will be in a valid state).

Let $s \in S$ arbitrarily chosen. Let

$$\begin{aligned}
g_s &:= p_s \wedge \left(\neg \bigvee_{s' \in S \backslash \{s\}} p_{s'} \right) \\
h_s &:= \left(\bigwedge_{p \in P(s)} p \right) \wedge \left(\neg \bigvee_{p \in AP \backslash P(s)} p \right) \\
d_s &:= \mathbf{X} \left(\bigvee_{\substack{s'' \in S \\ (s,s'') \in R}} p_{s''} \right) \\
f_s &:= g_s \wedge h_s \wedge d_s
\end{aligned}$$

Let $f_M := \mathbf{G} \left(\bigvee_{s \in S} f_s \right)$ and finally $r(\mathbf{E}f) := \mathbf{E}(f \wedge f_M \wedge p_{s_0})$.

Theorem 2.2.

Let M, s_0, f be defined as above and let $r : LTL \rightarrow LTL$ be defined by the above transformation. Then $M, s_0 \models \mathbf{E}f \Leftrightarrow r(\mathbf{E}f)$ is satisfiable. And thus LTL-truth $\leq_m^p$ LTL-satisfiability since r is clearly computable in polynomial time.

Proof. "$\Rightarrow$":
Let $M, s_0 \models \mathbf{E}f$.
Proposition: $M', s_0 \models r(\mathbf{E}f)$ where $M' := (S, R, P')$ with $P'(s) := P(s) \cup \{p_s\}$ $\forall s \in S$.
Proof (of proposition): As $M, s_0 \models \mathbf{E}f$ by precondition, there is a path $u = (u_0 = s_0, u_1, u_2, \dots)$ with $M, u \models f$.
$\Rightarrow M', u \models f$ because in every formula which contains only atomic propositions from AP and none from AP_S M and M' behave exactly the same way.
Trivially $M', u \models p_{s_0}$ and so we just need to show that $M', u \models f_M$.

But this can also easily be seen as f_M is constructed to model M:
$\forall k \geq 0 :$ $M', u^k \models g_{u_k} \wedge h_{u_k}$ due to the definition of P' and $M', u^k \models d_{u_k}$ because $(u_k, u_{k+1}) \in R$.
$\Rightarrow \forall k \geq 0 :$ $M', u^k \models f_{u_k}$ and hence $M', u \models f_M$.
$\Rightarrow$ $M', u \models f \wedge f_M \wedge p_{s_0}$ $\Rightarrow$ $M', s_0 \models r(\mathbf{E}f)$.

"$\Leftarrow$":
Let $r(\mathbf{E}f)$ be satisfiable and let $M' = (S', R', P')$, $s'_0 \in S'$, $u' = (u'_0 = s'_0, u'_1, u'_2, \dots) \in S'^{\mathbb{N}}$ such that $M', u' \models f \wedge f_M \wedge p_{s_0}$.

We will now prove by induction on k that there is a path $u = (u_0, u_1, u_2, \dots)$ in M with $u_0 = s_0$ and $\forall k \geq 0 : \; P'(u'_k) = P(u_k) \cup \{p_{u_k}\}$. (We can w.l.o.g. assume that the set of atomic propositions of $M' = AP \cup AP_S$.)

Induction beginning: Because $M', u' \models p_{s_0}$ and $M', u' \models f_M$

$\Rightarrow \; M', u' \models f_{s_0}$

$\Rightarrow \; P'(s'_0) = P(s_0) \cup \{p_{s_0}\}$.

Induction hypothesis: True for $u_0, \dots, u_k$.

Induction conclusion: Because $P'(u'_k) = P(u_k) \cup \{p_{u_k}\}$ and $M', u' \models f_M$

$\Rightarrow \; M', u'^k \models f_{u_k}$

$\Rightarrow \; M', u'^k \models d_{u_k}$

$\Rightarrow \; \exists u_{k+1} \in S : (u_k, u_{k+1}) \in R \wedge M', u'^{k+1} \models p_{u_{k+1}}$

$\Rightarrow \; M', u'^{k+1} \models f_{u_{k+1}}$

$\Rightarrow \; P'(u'_{k+1}) = P(u_{k+1}) \cup p_{u_{k+1}}$.

Now we are able to prove that $M, u \models f$:
Since $\forall k \geq 0 : \; P'(u'_k) = P(u_k) \cup \{p_{u_k}\}$ and f contains only atomic propositions from AP and none from AP_S, M behaves exactly the same way along u as M' does along u'. $\Rightarrow M, u \models f \Rightarrow M, s_0 \models \mathbf{E}f$. $\square$

2.2 LTL-truth is PSPACE-hard

Theorem 2.3.
LTL-truth is PSPACE-hard.

Proof.
Let $T = (Q, \Sigma, \delta, Q_a, Q_r, q_0, \square)$ be a one-tape DTM where Q is the set of states, Σ is the alphabet, $\delta : Q \times \Sigma \to Q \times \Sigma \times \{L, R\}$ is the transition function, Q_a is the set of accepting states, Q_r is the set of rejecting states, q_0 is the initial state and $\square \in \Sigma$ is the blank symbol. Let T immediately stop its computation when reaching a state in Q_a or in Q_r. Let T be $P(n)$-space bounded where P is a polynomial and w.l.o.g. T only visits tape cells to the right of the input word. Let $a = a_1 a_2 a_3 \dots a_n$ be an input to T.

Let $M = (S, R, P)$ be the following Kripke structure:

$$\begin{aligned}
S &= \{\mathit{begin}, \mathit{end}\} \cup \Big(\{1, 2, 3, \dots, P(n)\} \times \big((Q \times \Sigma) \cup \Sigma\big)\Big) \\
R &= \{(\mathit{begin}, (1, \sigma)) \mid \sigma \in (Q \times \Sigma) \cup \Sigma\} \\
&\quad \cup \; \{((P(n), \sigma), \mathit{end}) \mid \sigma \in (Q \times \Sigma) \cup \Sigma\} \\
&\quad \cup \; \{(\mathit{end}, \mathit{begin})\} \\
&\quad \cup \; \{((i, \sigma), (i+1, \tau)) \mid \sigma, \tau \in (Q \times \Sigma) \cup \Sigma, i \in \{1, \dots, P(n) - 1\}\}
\end{aligned}$$

M has $P(n)$ "columns" which are connected as a chain. In each column there are $|(Q \times \Sigma) \cup \Sigma|$ vertices.

$$\begin{aligned}
P(\mathit{begin}) &= \mathit{begin} \\
P(\mathit{end}) &= \mathit{end} \\
P((i, \sigma)) &= p_\sigma \quad \forall \sigma \in (Q \times \Sigma) \cup \Sigma, i \in \{1, \dots, P(n)\}
\end{aligned}$$

The set of atomic propositions is $AP = \{p_\sigma | \sigma \in (Q \times \Sigma) \cup \Sigma\} \cup \{\mathit{begin}, \mathit{end}\}$.

$\Rightarrow$ Each subpath between *begin* and *end* (where exactly one vertex is in $Q \times \Sigma$ and the others are in Σ) represents a configuration of T (the vertex from $Q \times \Sigma$ denotes the current state and head position of T) and a path beginning at *begin* represents a computation of T.

Now we will construct a LTL formula $f = r(T, a)$ which will be true in M at the state *begin* iff T halts in an accepting state when given a as input:

$f_{valid} :=$

$$\mathbf{G}\Bigg(\mathit{begin} \to \Big(\bigvee_{k=1\dots P(n)} \Big(\big(\mathbf{X}\bigvee_{\sigma\in\Sigma} p_\sigma\big) \wedge \big(\mathbf{XX}\bigvee_{\sigma\in\Sigma} p_\sigma\big) \wedge \big(\mathbf{XXX}\bigvee_{\sigma\in\Sigma} p_\sigma\big) \wedge \dots \wedge \big(\underbrace{\mathbf{XX}\dots\mathbf{X}}_{(k-1)-\text{times}}\bigvee_{\sigma\in\Sigma} p_\sigma\big)$$

$$\wedge \big(\underbrace{\mathbf{XX}\dots\mathbf{X}}_{k-\text{times}}\bigvee_{\sigma\in Q\times\Sigma} p_\sigma\big) \wedge \big(\underbrace{\mathbf{XX}\dots\mathbf{X}}_{(k+1)-\text{times}}\bigvee_{\sigma\in\Sigma} p_\sigma\big) \wedge \dots \wedge \big(\underbrace{\mathbf{XX}\dots\mathbf{X}}_{P(n)-\text{times}}\bigvee_{\sigma\in\Sigma} p_\sigma\big)\Big)\Big)\Bigg)$$

asserts that every subpath between *begin* and *end* represents a valid configuration of T.

$f_{initial} :=$

$$\mathbf{X}p_{(q_0,a_1)} \wedge \mathbf{XX}p_{a_2} \wedge \mathbf{XXX}p_{a_3} \wedge \mathbf{XXXX}p_{a_4} \wedge \dots \wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{n-\text{times}}p_{a_n} \wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{(n+1)-\text{times}} p_\Box \wedge$$

$$\dots \wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{P(n)-\text{times}} p_\Box$$

asserts that the first subpath between *begin* and *end* represents the initial configuration of T with input a.

$\forall (z,b) \in Q \times \Sigma :\;\; f_{transition,z,b} := f_{left\_transition,z,b} :=$

$$\bigwedge_{k=2\dots P(n)} \mathbf{G}\Bigg(\Big(\mathit{begin} \wedge \mathbf{X}p_{b_1} \wedge \mathbf{XX}p_{b_2} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k-1}} \wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{k-\text{times}}p_{(z,b)}$$

$$\wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k+1}} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{P(n)}}\Big) \to$$

$$\underbrace{\mathbf{XX}\dots\mathbf{X}}_{(P(n)+2)-\text{times}} \Big(\mathbf{X}p_{b_1} \wedge \mathbf{XX}p_{b_2} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k-2}} \wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{(k-1)-\text{times}} p_{(z',b_{k-1})}$$

$$\wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{k-\text{times}}p_c \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k+1}} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{P(n)}}\Big)\Bigg) \text{ if } \delta((z,b)) = (z', c, \mathrm{L})$$

or $f_{transition,z,b} := f_{right\_transition,z,b} :=$

$$\bigwedge_{k=1\dots P(n)-1} \mathbf{G}\Bigg(\Big(\mathit{begin} \wedge \mathbf{X}p_{b_1} \wedge \mathbf{XX}p_{b_2} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k-1}} \wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{k-\text{times}}p_{(z,b)}$$

$$\wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k+1}} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{P(n)}}\Big) \to$$

$$\underbrace{\mathbf{XX}\dots\mathbf{X}}_{(P(n)+2)-\text{times}} \Big(\mathbf{X}p_{b_1} \wedge \mathbf{XX}p_{b_2} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k-1}} \wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{k-\text{times}}p_c$$

$$\wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{(k+1)-\text{times}} p_{(z',b_{k+1})} \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k+2}} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{P(n)}}\Big)\Bigg) \text{ if } \delta((z,b)) = (z', c, \mathrm{R})$$

or $f_{transition,z,b} := f_{loop,z,b} :=$

$$\bigwedge_{k=1\dots P(n)} \mathbf{G}\Bigg(\Big(\mathit{begin} \wedge \mathbf{X}p_{b_1} \wedge \mathbf{XX}p_{b_2} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k-1}} \wedge \underbrace{\mathbf{XX}\dots\mathbf{X}}_{k-\text{times}}p_{(z,b)}$$

$$\wedge \mathbf{XX}\dots\mathbf{X}p_{b_{k+1}} \wedge \dots \wedge \mathbf{XX}\dots\mathbf{X}p_{b_{P(n)}}\Big) \to$$

$$\underbrace{\mathbf{XX}\ldots\mathbf{X}}_{(P(n)+2)-\text{times}} \Big(\mathbf{X}p_{b_1} \wedge \mathbf{XX}p_{b_2} \wedge \cdots \wedge \mathbf{XX}\ldots\mathbf{X}p_{b_{k-1}} \wedge \underbrace{\mathbf{XX}\ldots\mathbf{X}}_{k-\text{times}}p_{(z,b)}$$
$$\wedge\, \mathbf{XX}\ldots\mathbf{X}p_{b_{k+1}} \wedge \cdots \wedge \mathbf{XX}\ldots\mathbf{X}p_{b_{P(n)}}\Big)\Big) \text{ if } z \in Q_a \cup Q_r$$

assert that every successive subpath represents the configuration which follows from the configuration represented by the previous subpath and that nothing changes anymore after an accepting or rejecting state is reached.

$$f_{\mathit{final}} := \mathbf{F} \bigvee_{\substack{q\in Q_a\\ b\in\Sigma}} p_{(q,b)}$$

asserts that eventually an accepting state will be reached.

$$f := f_{\mathit{valid}} \wedge f_{\mathit{initial}} \wedge f_{\mathit{final}} \wedge \bigwedge_{(z,b)\in Q\times\Sigma} f_{\mathit{transition},z,b}.$$

$\Rightarrow$ $M, \mathit{begin} \models \mathbf{E}f \Leftrightarrow T$ halts in an accepting state when given input a. Clearly M and f are space-bounded by a polynomial in the length of T and a. $\square$

2.3 LTL-satisfiability ∈ PSPACE

Now we will show that LTL-satisfiability ∈ PSPACE. In order to do this we will need some other results in preparation.

Definition 2.4.
Let $M = (S, R, P)$ be a Kripke structure, let $u = (u_0, u_1, \ldots)$ be a path in M and let f be a LTL path formula. Then $[u]_{M,f} := \{g \in \text{subformulas}(f) \mid M, u \models g\}$.

Lemma 2.5.
Let $M = (S, S \times S, P)$ be a Kripke structure, let $u = (u_0, u_1, \ldots)$ be a path in M, let f be a LTL path formula, let $i < j \in \mathbb{N}$, let $[u^i]_{M,f} = [u^j]_{M,f}$ and let $u' = (u_0, u_1, \ldots, u_{i-1}, u_j, u_{j+1}, u_{j+2}, \ldots)$.
$\Rightarrow \forall k \in \mathbb{N}\backslash\{i, i+1, \ldots, j-1\} : [u^k]_{M,f} = [u'^{k'}]_{M,f}$ where $k' \in \mathbb{N}$ such that $u'^{k'}$ begins at u_k.

Proof. (by induction on the length of f)
Notation: $\forall l \in \mathbb{N}\backslash\{i, i+1, \ldots, j-1\}$: l' shall denote the natural number with $u'^{l'} = (u_l, u_{l+1}, u_{l+2}, \ldots, u_{i-1}, u_j, u_{j+1}, \ldots)$.

$$\Rightarrow l' = \begin{cases} l & \text{if } l < i \\ l-(j-i) & \text{if } l \geq j \end{cases}$$

Induction beginning: length$(f) = 1$
$\Rightarrow$ $f \in$ AP
$\Rightarrow$ $SF(f) :=$ subformulas$(f) = \{f\}$
$\Rightarrow$ $\big(M, u^k \models f \Leftrightarrow f \in P(u_k) \Leftrightarrow M, u'^{k'} \models f\big)$

Induction hypothesis: True for length$(f) < n$.

Induction conclusion: Let $\mathit{length}(f) = n$.
Let $g \in SF(f)\backslash\{f\}$ arbitrarily chosen.

$$\overset{SF(g)\subseteq SF(f)}{\Rightarrow} \quad [u^i]_{M,g} = [u^j]_{M,g}$$
$$\overset{\text{I.H.} \wedge \mathit{length}(g) < n}{\Rightarrow} \quad [u^k]_{M,g} = [u'^{k'}]_{M,g}$$

So we just have to prove that
$$M, u^k \models f \Leftrightarrow M, u'^{k'} \models f.$$

Case 1: $f = \neg g \;\Rightarrow$

$$\begin{aligned} M, u^k \models f &\Leftrightarrow M, u^k \not\models g \\ &\overset{\text{I.H.}}{\Leftrightarrow} M, u'^{k'} \not\models g \\ &\Leftrightarrow M, u'^{k'} \models f \end{aligned}$$

Case 2: $f = g \wedge h \;\Rightarrow$

$$\begin{aligned} M, u^k \models f &\Leftrightarrow M, u^k \models g \text{ and } M, u^k \models h \\ &\overset{\text{I.H.}}{\Leftrightarrow} M, u'^{k'} \models g \text{ and } M, u'^{k'} \models h \\ &\Leftrightarrow M, u'^{k'} \models f \end{aligned}$$

Case 3: $f = \mathbf{X}g$

Case 3.1: $k \neq i - 1 \;\Rightarrow$

$$\begin{aligned} M, u^k \models f &\Leftrightarrow M, u^{k+1} \models g \\ &\overset{\text{I.H.}}{\Leftrightarrow} M, u'^{k'+1} \models g \\ &\Leftrightarrow M, u'^{k'} \models f \end{aligned}$$

Case 3.2: $k = i - 1 \;\Rightarrow$

$$\begin{aligned} M, u^{i-1} \models f &\Leftrightarrow M, u^i \models g \\ &\overset{\text{precondition} \;\wedge\; g \in SF(f)}{\Leftrightarrow} M, u^j \models g \\ &\overset{\text{I.H.}}{\Leftrightarrow} M, u'^{j'} \models g \;\; (j' = i) \\ &\Leftrightarrow M, u'^{i-1} \models f \end{aligned}$$

Case 4: $f = g\mathbf{U}h$

Case 4.1: $k \geq j \;\Rightarrow$

$$\begin{aligned} \forall a \geq k: \;\; & M, u^a \models g \overset{\text{I.H.}}{\Leftrightarrow} M, u'^{a'} \models g \;\; (a' = a - (j - i)) \\ & M, u^a \models h \overset{\text{I.H.}}{\Leftrightarrow} M, u'^{a'} \models h \end{aligned}$$

$$\begin{aligned} \Rightarrow \quad M, u^k \models f &\Leftrightarrow \exists m \geq k: \quad M, u^m \models h \\ & \qquad\qquad\qquad \wedge\; \forall k \leq l < m : M, u^l \models g \\ &\Leftrightarrow \exists m' \geq k': \quad M, u'^{m'} \models h \\ & \qquad\qquad\qquad \wedge\; \forall k' \leq l' < m' : M, u'^{l'} \models g \\ &\Leftrightarrow M, u'^{k'} \models f \end{aligned}$$

Case 4.2: $k < i \;\; (\Rightarrow k = k')$

"$\Rightarrow$": Let $M, u^k \models f \;\Rightarrow\; \exists m \geq k : \big(M, u^m \models h \;\wedge\; \forall k \leq l < m : M, u^l \models g\big)$.

Case 4.2.1a: $m < i \overset{\text{I.H.}}{\Rightarrow}$

$$\begin{aligned} & M, u'^m \models h \;\wedge\; \forall k \leq l < m : M, u'^l \models g \\ \Rightarrow \;\; & M, u'^k \models f \end{aligned}$$

Case 4.2.2a: $m \geq i \;\Rightarrow$

$$\begin{array}{rl}
 & M, u^i \models f \text{ (since } M, u^m \models h \;\wedge\; \forall l \text{ with } k < \mathbf{i} \leq l < m : M, u^l \models g) \\
\overset{\text{precondition} \,\wedge\, f \in SF(f)}{\Rightarrow} & M, u^j \models f \\
\Rightarrow & \exists a \geq j : \big(M, u^a \models h \;\wedge\; \forall j \leq l < a : M, u^l \models g\big) \\
\overset{\text{I.H.}}{\Rightarrow} & M, u'^{a'} \models h \;\wedge\; \forall j \leq l < a : M, u'^{l'} \models g \text{ (*)} \\
\text{Because} & \forall k \leq l < m : M, u^l \models g \;\wedge\; i \leq m \\
\Rightarrow & \forall k \leq l < i : M, u^l \models g \\
\overset{\text{I.H.}}{\Rightarrow} & \forall k \leq l < i : M, u'^l \models g \text{ (**)} \\
\text{From (*) and (**)} \;\Rightarrow & \forall k \leq l' < a' : M, u'^{l'} \models g \\
\Rightarrow & M, u'^k \models f
\end{array}$$

"$\Leftarrow$": Let $M, u'^k \models f \;\Rightarrow\; \exists m \geq k : \big(M, u'^{m'} \models h \;\wedge\; \forall k \leq l' < m' : M, u'^{l'} \models g\big)$.

Case 4.2.1b: $m < i \overset{\text{I.H.}}{\Rightarrow}$

$$\begin{array}{rl}
 & M, u^m \models h \;\wedge\; \forall k \leq l < m : M, u^l \models g \\
\Rightarrow & M, u^k \models f
\end{array}$$

Case 4.2.2b: $m \geq i \;\Rightarrow\; m \geq j \;\Rightarrow$

$$\begin{array}{rl}
 & M, u'^{j'} \models f \;\; (j' = i) \\
\Rightarrow & \exists a \geq j : \big(M, u'^{a'} \models h \;\wedge\; \forall j \leq l < a : M, u'^{l'} \models g\big) \\
\overset{\text{I.H.}}{\Rightarrow} & M, u^a \models h \;\wedge\; \forall j \leq l < a : M, u^l \models g \\
\Rightarrow & M, u^j \models f \\
\overset{\text{precondition} \,\wedge\, f \in SF(f)}{\Rightarrow} & M, u^i \models f \text{ (*)} \\
\text{Because} & \forall k \leq l' < m' : M, u'^{l'} \models g \;\wedge\; i \leq m \\
\Rightarrow & \forall k \leq l < i : M, u'^l \models g \\
\overset{\text{I.H.}}{\Rightarrow} & \forall k \leq l < i : M, u^l \models g \text{ (**)} \\
\text{From (*) and (**)} \;\Rightarrow & M, u^k \models f
\end{array}$$

□

Let $f = g\mathbf{U}h$, let $i, j \in \mathbb{N}$, $j > i$ and let $M, p^i \models f$. Then we say that f is *fulfilled before* p_j iff $\exists i \leq k < j : M, p^k \models h$.

Let $M = (S, R, P)$ be a Kripke structure and let $u = (u_0, u_1, \dots)$ be a path in M. Then we say that u is *ultimately periodic* with starting index l and period p iff $\forall k \geq l : P(u_k) = P(u_{k+p})$.

A formula f is called a **U**-*formula* iff it is of the form $f = g\mathbf{U}h$.

Lemma 2.6.
Let $M = (S, S{\times}S, P)$ be a Kripke structure, let $u = (u_0, u_1, \dots)$ be a path in M and let $n, p \in \mathbb{N}$ such that $p > 0$, $[u^n]_{M,f} = [u^{n+p}]_{M,f}$ and every **U**-*formula in $[u^n]_{M,f}$ is fulfilled before u_{n+p}. Let $u' = (u_0, u_1, \dots, u_n, \dots, u_{n+p-1}, u_n, u_{n+1}, \dots, u_{n+p-1}, u_n, \dots)$. u' is an ultimately periodic path in M with starting index n and period p.*

Then the following is true:
(a) $\forall k < n + p : [u^k]_{M,f} = [u'^k]_{M,f}$
(b) $\forall k \geq n : [u'^k]_{M,f} = [u'^{k+p}]_{M,f}$

Proof. (by structural induction on $g \in SF(f)$)
We need to show that $\forall g \in SF(f)$ the following is true:
(a) $\forall k < n + p : M, u^k \models g \;\Leftrightarrow\; M, u'^k \models g$
(b) $\forall k \geq n : M, u'^k \models g \;\Leftrightarrow\; M, u'^{k+p} \models g$
So let $g \in SF(f)$ arbitrarily chosen.

(b) trivially holds because $\forall k \geq n : u'^k = u'^{k+p}$.

(a) Induction beginning: $g \in AP \Rightarrow$ Trivially holds.

Induction hypothesis: holds for $g_1, g_2 \in SF(f)$.

Induction conclusion:

Case 1 & 2: $g = \neg g_1$ or $g = g_1 \wedge g_2 \overset{\text{I.H.}}{\Rightarrow}$ Trivially holds.

Case 3: $g = \mathbf{X} g_1 \ \Rightarrow$

Case 3.1: $k < n+p-1 \overset{\text{I.H.}}{\Rightarrow}$ trivial

Case 3.2: $k = n+p-1 \ \Rightarrow$

$$\begin{array}{rcl} M, u^k \models g & \overset{\text{def. of } \mathbf{X}}{\Leftrightarrow} & M, u^{n+p} \models g_1 \\ & \overset{\text{precondition} \wedge g_1 \in SF(f)}{\Leftrightarrow} & M, u^n \models g_1 \\ & \overset{\text{I.H.}}{\Leftrightarrow} & M, u'^n \models g_1 \\ & \overset{\text{(b)}}{\Leftrightarrow} & M, u'^{n+p} \models g_1 \\ & \overset{\text{def. of } \mathbf{X}}{\Leftrightarrow} & M, u'^k \models g \end{array}$$

Case 4: $g = g_1 \mathbf{U} g_2$

"$\Rightarrow$": Let $M, u^k \models g \ \Rightarrow \ \exists m \geq k : \big(M, u^m \models g_2 \ \wedge \ \forall k \leq l < m : M, u^l \models g_1\big)$.

Case 4.1a: $m < n+p \overset{\text{I.H.}}{\Rightarrow} M, u'^k \models g$

Case 4.2a: $m \geq n+p \ \Rightarrow$

$$\begin{array}{rl} & M, u^{n+p} \models g \\ \overset{\text{precondition}}{\Rightarrow} & M, u^n \models g \\ \Rightarrow & \exists m' \geq n : \big(M, u^{m'} \models g_2 \ \wedge \ \forall n \leq l < m' : M, u^l \models g_1\big) \\ \overset{\text{precondition}}{\Rightarrow} & m' < n+p \\ \overset{\text{I.H.}}{\Rightarrow} & M, u'^n \models g \\ \overset{\text{(b)}}{\Rightarrow} & M, u'^{n+p} \models g \\ \overset{\forall k \leq l < n+p \leq m : M, u'^l \models g_1}{\Rightarrow} & M, u'^k \models g \end{array}$$

"$\Leftarrow$": Let $M, u'^k \models g \ \Rightarrow \ \exists m \geq k : \big(M, u'^m \models g_2 \ \wedge \ \forall k \leq l < m : M, u'^l \models g_1\big)$.

Case 4.1b: $m < n+p \overset{\text{I.H.}}{\Rightarrow} M, u^k \models g$

Case 4.2b: $m \geq n+p \ \Rightarrow$

$$\begin{array}{rl} & M, u'^{n+p} \models g \\ \overset{\text{(b)}}{\Rightarrow} & M, u'^n \models g \\ \Rightarrow & \exists m' \geq n : m' < n+p \ \wedge \ \big(M, u'^{m'} \models g_2 \ \wedge \ \forall n \leq l < m' : M, u'^l \models g_1\big) \\ & \textit{Proof (that there is an } m' < n+p\textit{):} \text{ Suppose there is not.} \\ \Rightarrow & m' \geq n+p \\ \overset{\text{(b)}}{\Rightarrow} & M, u'^{m'-p} \models g_2 \wedge \forall n \leq l < m'-p : M, u'^l \models g_2 \\ \Rightarrow & \text{Contradiction; as we could have chosen } m'-p \text{ instead of } m'. \\ \Rightarrow & \exists m' < n+p \\ \overset{\text{I.H.}}{\Rightarrow} & M, u^n \models g \\ \overset{\text{precondition}}{\Rightarrow} & M, u^{n+p} \models g \\ \overset{\forall k \leq l < n+p \leq m : M, u^l \models g_1}{\Rightarrow} & M, u^k \models g \end{array}$$

□

Theorem 2.7 (Ultimately periodic model theorem)**.**
Let $f \in LTL$. Then f is satisfiable iff it is satisfiable on an ultimately periodic path $v = (v_0, v_1, \dots)$ with starting index $n \leq 2^{length(f)}$, period $p \leq 4^{length(f)}$ and $\forall k \geq n : \big([v^k]_{M,f} = [v^{k+p}]_{M,f} \quad \wedge \quad$ every $\mathbf{U}$-formula in $[v^k]_{M,f}$ is fulfilled before $v_{k+p}\big)$.

Proof. Let $\mathbf{E}f \in$ LTL-satisfiability and let $M = (S, S \times S, P)$, $u = (u_0, u_1, \dots)$ such that $M, u \models f$ (we can choose $S \times S$ as transition relation since we are interested in whether a path exists and if one exists with respect to some transition relation, it stays a valid path after substituting this relation with $S \times S$). Let $l, m \in \mathbb{N}$ such that $[u^l]_{M,f} = [u^{l+m}]_{M,f} \wedge$ (*)every $\mathbf{U}$-formula in $[u^l]_{M,f}$ is fulfilled before u_{l+m}. Such l, m exist because $\forall i \in \mathbb{N} : [u^i]_{M,f} \subseteq SF(f)$ and there are $\leq 2^{|SF(f)|} = 2^{length(f)}$ different subsets of $SF(f)$ (and every of the only finitely many $\mathbf{U}$-formulas in $[u^l]_{M,f}$ is eventually fulfilled after a finite number of states).

Now we do the following:

```
While $\exists i,j \in \mathbb{N}$, $1 < i < j < l$ and $[p^i]_{M,f} = [p^j]_{M,f}$ do
    Delete the subpath $(u_i, u_{i+1}, \dots, u_{j-1})$ from $u$.
End While
While $\exists i,j \in \mathbb{N}$, $l < i < j < l+m$ and $[p^i]_{M,f} = [p^j]_{M,f}$ do
    Delete the subpath $(u_i, u_{i+1}, \dots, u_{j-1})$ from $u$
     if this is possible without violating (*).
End While
```

Let u' be the modified path. In u' there are $\leq 2^{length(f)}$ states before u_l (a) and $\leq length(f) \cdot 2^{length(f)}$ states between u_l and u_{l+m} (b).

Proof:

(a) is true because there are no two states u_i, u_j before u_l with $[u'^i]_{M,f} = [u'^j]_{M,f}$ and $\big|\{[v]_{M,f} | v \in S^{\mathbb{N}}\}\big| \leq 2^{length(f)}$.

Now assume (b) does not hold. $\Rightarrow \exists u_{i_0}, u_{i_1}, \dots, u_{i_{length(f)}}$ between u_l and u_{l+m} : $[u'^{i_0}]_{M,f} = [u'^{i_1}]_{M,f} = \dots = [u'^{i_{length(f)}}]_{M,f}$. Because there are less than length(f) $\mathbf{U}$-formulas in $SF(f)$, there is an interval $\{u_{i_j}, u_{i_j+1} \dots u_{i_{j+1}-1}\}$ (for a $j \in \{0, 1, \dots, length(f) - 1\}$), in which no $\mathbf{U}$-formula, which is true on u^l, is fulfilled. But this means that we could have deleted the subpath $(u_{i_j}, u_{i_j+1}, \dots, u_{i_{j+1}-1})$. This is a contradiction and so (b) does hold.

Now set $n := l' \leq 2^{length(f)}$ (where l' is the index of u_l in u') and $p := m' \leq 4^{length(f)}$ (where $l' + m'$ is the index of u_{l+m} in u'). $\Rightarrow [u'^n]_{M,f} = [u'^{n+p}]_{M,f} \wedge$ every $\mathbf{U}$-formula in $[u'^n]_{M,f}$ is fulfilled before u'_{n+p}. Using Lemma 2.6 we obtain that the ultimately periodic path $v = (u'_0, u'_1, \dots, u'_{n+p-1}, u'_n, u'_{n+1}, \dots)$ fulfills all the wanted conditions and that $[v]_{M,f} = [v^0]_{M,f} \overset{La.\ 2.6(a)}{=} [u'^0]_{M,f} \overset{La.\ 2.5}{=} [u^0]_{M,f} = [u]_{M,f}$. And therefore $M, v \models f$ since $M, u \models f$ by precondition. □

Theorem 2.8.
LTL-satisfiability $\in$ PSPACE

Proof. Let $\mathbf{E}f \in$ LTL. Then Theorem 2.7 states that we only need to find out whether f is satisfiable on an ultimately periodic path. For that we will now give a non-deterministic algorithm which uses space linear in *length*(f). First guess two numbers $l \leq 2^{length(f)}$ and $p \leq 4^{length(f)}$ which are supposed to be the starting index and the period of an ultimately periodic path.

We will now try to non-deterministically find a Kripke structure M and an ultimately periodic path u in M with starting index l and period p such that $M, u \models f$. First note that it is enough to find a "consistent" sequence $(sub_n)_{n\in\mathbb{N}}$ of subsets of $SF(f)$ with $f \in sub_0 \wedge sub_{n+p} = sub_n \; \forall n \geq l$ since then for $M := (2^{AP}, 2^{AP} \times 2^{AP}, P)$ with $P(sub) := sub$ and $u = (u_0, u_1, \dots)$ with $u_i := sub_i \cap AP$: $M, u \models f$. *Consistent* here means that no two or more sub_i and sub_j contradict each other, which would e.g. be the case if $\mathbf{X}p_0 \in sub_0$ and $\neg p_0 \in sub_1$.

On the other hand we can obtain such a consistent sequence $(sub_n)_{n\in\mathbb{N}}$ of subsets of $SF(f)$ if we have a Kripke structure $M = (S, R, P)$ and an ultimately periodic path $u = (u_0, u_1, \dots)$ in M with $M, u \models f$ by defining $sub_i := P(u_i)$.

$\Rightarrow$ f is satisfiable on an ultimately periodic path with starting index l and period p iff there is a consistent sequence of subsets of $SF(f)$ fulfilling the above conditions. Therefore it is enough to give an algorithm for finding out whether such a sequence exists.

The idea for finding such a sequence is to subsequently guess the subformulas, which are true, in every state of the ultimately periodic path to be constructed and check that every guess is consistent with the former guesses.

$Sub_{present} \subseteq SF(f)$ is the set of subformulas guessed to be true in the present state and $Sub_{next} \subseteq SF(f)$ is the set of subformulas guessed to be true in the next state.

The algorithm does the following:

```
Guess $Sub_{present} \subseteq SF(f)$.
If $\neg(f \in Sub_{present})$ then reject.
Set $n := 0$.
While $n < l$ do
    Guess $Sub_{next} \subseteq SF(f)$.
    If $\neg consistent(Sub_{present}, Sub_{next})$ then reject.
    $Sub_{present} := Sub_{next}$.
    $n := n + 1$.
End While
$Sub_{period} := Sub_{present}$.
$\forall g = g_1 \mathbf{U} g_2 \in Sub_{period} : A_g := false$.    (We need to check that every $\mathbf{U}$-formula in
                                                $[u^l]_{M,f}$ is fulfilled before $u_{l+p}$)
$n := 0$
While $n < p$ do
    Guess $Sub_{next} \subseteq SF(f)$.
    If $\neg consistent(Sub_{present}, Sub_{next})$ then reject.
    If $\exists g = g_1 \mathbf{U} g_2 \in Sub_{period}$ with $g_2 \in Sub_{present}$ then $A_g := true$.
    $Sub_{present} := Sub_{next}$.
    $n := n + 1$.
End While
If $\exists g = g_1 \mathbf{U} g_2 \in Sub_{period}$ with $A_g = false$ then reject.
$Sub_{next} := Sub_{period}$.
If $\neg consistent(Sub_{present}, Sub_{next})$ then reject.
Accept.
```

Hereby $consistent\big(Sub_{present}, Sub_{next}\big)$ is computed by the following algorithm:

```
For each $g \in SF(f)$ do
    If $g = \neg g_1$ then
        If $\neg\,(g \in Sub_{present} \Leftrightarrow g_1 \notin Sub_{present})$ then return $false$.
    If $g = g_1 \wedge g_2$ then
        If $\neg\,(g \in Sub_{present} \Leftrightarrow g_1 \in Sub_{present} \wedge g_2 \in Sub_{present})$
         then return $false$.
    If $g = \mathbf{X}g_1$ then
        If $\neg\,(g \in Sub_{present} \Leftrightarrow g_1 \in Sub_{next})$ then return $false$.
    If $g = g_1\mathbf{U}g_2$ then
        If $\neg(g \in Sub_{present} \Leftrightarrow g_2 \in Sub_{present} \vee$
                                   $(g_1 \in Sub_{present} \wedge g_1\mathbf{U}g_2 \in Sub_{next}))$
        then return $false$.
End For
Return $true$.
```

The algorithm accepts an input formula iff it is satisfiable as explained above. □

Corollary 2.9.
LTL-satisfiability and LTL-truth are PSPACE-complete.

Proof. Theorem 2.3 showed that LTL-truth is PSPACE-hard. As LTL-truth $\leq_m^p$ LTL-satisfiability as proved in Theorem 2.2, LTL-satisfiability is also PSPACE-hard.

Theorem 2.8 showed that LTL-satisfiability $\in$ PSPACE. Because of the just mentioned reduction this is also the case for LTL-truth. □

2.4 CTL*-truth $\in$ PSPACE

Theorem 2.10.
CTL*-truth $\in$ PSPACE.

Proof. Let $M = (S, R, P)$ be a Kripke structure with AP as the set of atomic propositions, let $s_0 \in S$, let $f \in$ CTL* and w.l.o.g. let f be free of **A**s. The following algorithm removes any path quantifiers from f:

```
Set $f' := f$.
Set $AP' := AP$.
Set $P' := P$.
While  $f'$ contains an $\mathbf{E}$  do
    Choose a $g \in SF(f')$ such that $g = \mathbf{E}g' \;\wedge\; g'$ does not contain an $\mathbf{E}$.
        (This means that $g \in$ LTL.)
    Compute the set $S_g$ of all states $s \in S$ with $M', s \models g$.
        (Theorem 2.8 states that this can be done in polynomial space.)
    $AP' := AP' \cup \{p_g\}$.
    $P'(s) := \begin{cases} P'(s) \cup \{p_g\} & \text{if} \quad s \in S_g \\ P'(s) & \text{else} \end{cases}$
    $M' := (S, R, P')$ with $AP'$ as the set of atomic propositions.
    Let the new $f'$ be the formula obtained by replacing all occurences
      of $g$ in the old $f'$ with $p_g$.
End While
```

Now f' is a temporal state formula not containing any path quantifiers and therefore is a simple Boolean formula. It can now easily be checked if $M', s_0 \models f'$ which is the case iff $M, s_0 \models f$. □

Corollary 2.11.
CTL-truth is PSPACE-complete.*

Proof. Theorem 2.3 showed that LTL-truth is PSPACE-hard. As trivially LTL-truth $\leq_m^p$ CTL*-truth, the latter is also PSPACE-hard.

Theorem 2.10 showed that CTL*-truth $\in$ PSPACE. □

References

[CES86] E. M. Clarke, Jr., E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, April 1986.

[CGP99] E. M. Clarke, Jr., O. Grumberg, and D. A. Peled. *Model Checking.* The MIT Press, 1999.

[SC85] A. P. Sistla and E. M. Clarke, Jr. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, July 1985.