The Complexity of Satisfiability for Fragments of CTL and CTL*[1]

Arne Meier,[a,2] Martin Mundhenk,[b,3] Michael Thomas[a,4] and Heribert Vollmer[a,5]

[a] *Theoretische Informatik*
Gottfried Wilhelm Leibniz Universität
Appelstr. 4, 30167 Hannover, Germany

[b] *Institut für Informatik*
Friedrich-Schiller-Universität
07737 Jena, Germany

Abstract

The satisfiability problems for CTL and CTL* are known to be EXPTIME-complete, resp. 2EXPTIME-complete (Fischer and Ladner (1979), Vardi and Stockmeyer (1985)). For fragments that use less temporal or propositional operators, the complexity may decrease. This paper undertakes a systematic study of satisfiability for CTL- and CTL*-formulae over restricted sets of propositional and temporal operators. We show that restricting the temporal operators yields satisfiability problems complete for 2EXPTIME, EXPTIME, PSPACE, and NP. Restricting the propositional operators either does not change the complexity (as determined by the temporal operators), or yields very low complexity like NC^1, TC^0, or NLOGTIME.

Keywords: Temporal Logic, Satisfiability, Post's Lattice.

1 Introduction

For reasoning about the ongoing behaviour of programs, in particular non-terminating programs such as operating systems, the branching time logic CTL*, introduced by Emerson and Halpern [7] (see also [6]), has been advocated to be a good language [19], and in the meantime it has proven to be useful even for practical purposes.

A decidable satisfiability problem is central for such logics in order to be a useful tool in program verification. For CTL*, satisfiability was proven to be complete

---

for double exponential time by Vardi and Stockmeyer [19]. For certain fragments, satisfiability is known to be more efficiently decidable: Sistla and Clarke [18] proved that for linear temporal logic LTL, the fragment of CTL* not allowing path quantifiers, the satisfiability problem is complete for polynomial space. For the fragment of LTL that disallows U, satisfiability is NP-complete. Markey [12] extended these results showing essentially that adding operators for the past does not increase complexity ("past is for free"). Further fragments of LTL were classified in [2]. Fischer and Ladner [8] proved that for computation tree logic CTL, the fragment of CTL* in which each path quantifier is followed by exactly one temporal operator that is not a path quantifier (i. e., X, U, F, G), satisfiability is complete for exponential time. However, a systematic study of the complexity of the satisfiability problem for fragments of CTL* has not been undertaken until today. This is the purpose of the present paper.

We first consider fragments of CTL and of CTL* where we restrict the allowed temporal operators. Here, a CTL-operator is a pair of a path quantifier (A and E) and non-path operator (X, U, etc.). We determine the lattice of all sets of temporal operators where one such set T_1 is below another set T_2 ($T_1 \sqsubseteq T_2$) iff the operators from T_1 can be expressed using operators T_2. Then we determine *for each set in the lattice* the complexity of the satisfiability problem restricted to only these temporal operators.

For CTL, we show, e. g., that the complexity of the satisfiability problem drops to NP-complete for the operators sets $\emptyset$ and $\{AF\}$, it is PSPACE-complete for $\{AX\}$, $\{AG\}$, $\{AX, AF\}$, $\{AF, AG\}$, and is complete for exponential time for all other cases. For CTL*, we show, e. g., that the complexity of the satisfiability problem drops to NP-complete for the operators sets $\emptyset$, $\{A\}$, $\{F\}$, and $\{X\}$, it is PSPACE-complete for $\{U\}$, $\{X, F\}$ and $\{U, X\}$, $\{A, F\}$ and $\{A, X\}$, and is complete for double exponential time for all other cases. Figure 1 summarizes these results.

As a second step, we also restrict the allowed propositional operators, following the approach undertaken in [2] for LTL. Let CTL*-SAT(T, B) denote the satisfiability problem for CTL* restricted to the fragments of formulae only allowing temporal operators from T and propositional operators from B. Here, we thus have to consider the lattice of all classes of Boolean functions, and we say that for such classes B_1, B_2, $B_1 \sqsubseteq B_2$ if all functions in B_1 can be obtained by superposition (essentially simple composition or substitution of functions) from functions in B_2. This lattice is the well known *Post's lattice* (cf., e. g., [14,3]), see Fig. 2.

It turns out that if B contains (or can implement) the negation of implication $x \not\rightarrow y$ (that is, $x \wedge \neg y$)—in terms of Post's lattice this means that $S_1 \subseteq B$— then satisfiability is as complicated as if allowing all propositional operators or a complete set such as $\{\wedge, \neg\}$, in other words, the complexity of CTL*-SAT(T, B) is determined by the set T as described above (and independent of the actual B). If on the other hand B cannot implement the negation of implication, then the complexity of CTL*-SAT(T, B) drops to a very low class inside the circuit class NC^1. In this case, the complexity of CTL*-SAT(T, B) astonishingly is independent of the temporal operators we allow. For example, if we consider only monotone formulae,
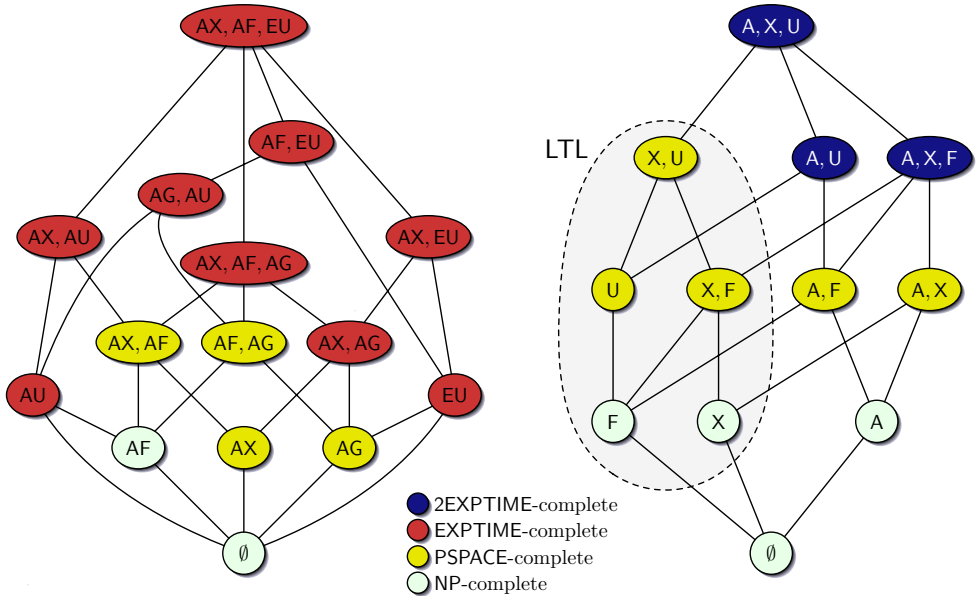
Fig. 1. The lattice of CTL-operators (left) and CTL⋆-operators (right). Nodes are labelled with a minimal set of operators; colors indicate the complexity of the satisfiability problem without restrictions on the Boolean connectives.

i. e., $B = \{\wedge, \vee, \text{true}, \text{false}\}$ (this corresponds to the class M in Post's lattice), then CTL⋆-SAT(T, B) is complete for NC^1 for all T. If $B \subseteq \{\wedge, \text{true}, \text{false}\}$ ($B \subseteq$ E in Post's lattice) then CTL⋆-SAT(T, B) is complete for TC^0 for all T. These results for the case of unrestricted temporal operators are summarized in Figure 2. It should be remarked that also in the case of simple propositional satisfiability, the operator $x \wedge \neg y$ determines if the problem is NP-complete or in P, see [11].

In this vein, we study the complexity of satisfiability for CTL and for CTL⋆ for all combinations of B and T. We give completeness results for 2EXPTIME, EXPTIME, PSPACE, NP, NC^1, and TC^0. However, we have to leave open one scenario: When B consists only of the exclusive-or (plus possibly the constants *true* and *false*) we can only state a trivial upper bound. We come back to this open case in the conclusion.

The reader might expect certain fragments of CTL⋆ to have a trivial satisfiability problem (e. g., since the allowed formulas are always satisfiable)—the lowest complexity in our classification, however, is completeness for TC^0. The reason is that the syntax alone, checking that a given word is a correct formula, leads to TC^0-completeness. In order to determine the cases of trivial satisfiability we therefore also study the *promise problem* to determine, given a syntactically correct formula as input, if it is satisfiable.

The rationale behind our approach is that looking for simpler fragments helps us to understand where the boundary lies between hard and easy fragments. This provides insight into the sources of hardness ($x \wedge \neg y$ on the propositional side, and for instance U on the temporal side). We also hope that our results might lead to improved algorithms for the special cases. One of our technically most involved results concerns the CTL-satisfiability for the operators $\{$AF, AG$\}$: Here we consider quasi-models (models whose labels are certain variants of Hintikka sets) and prove
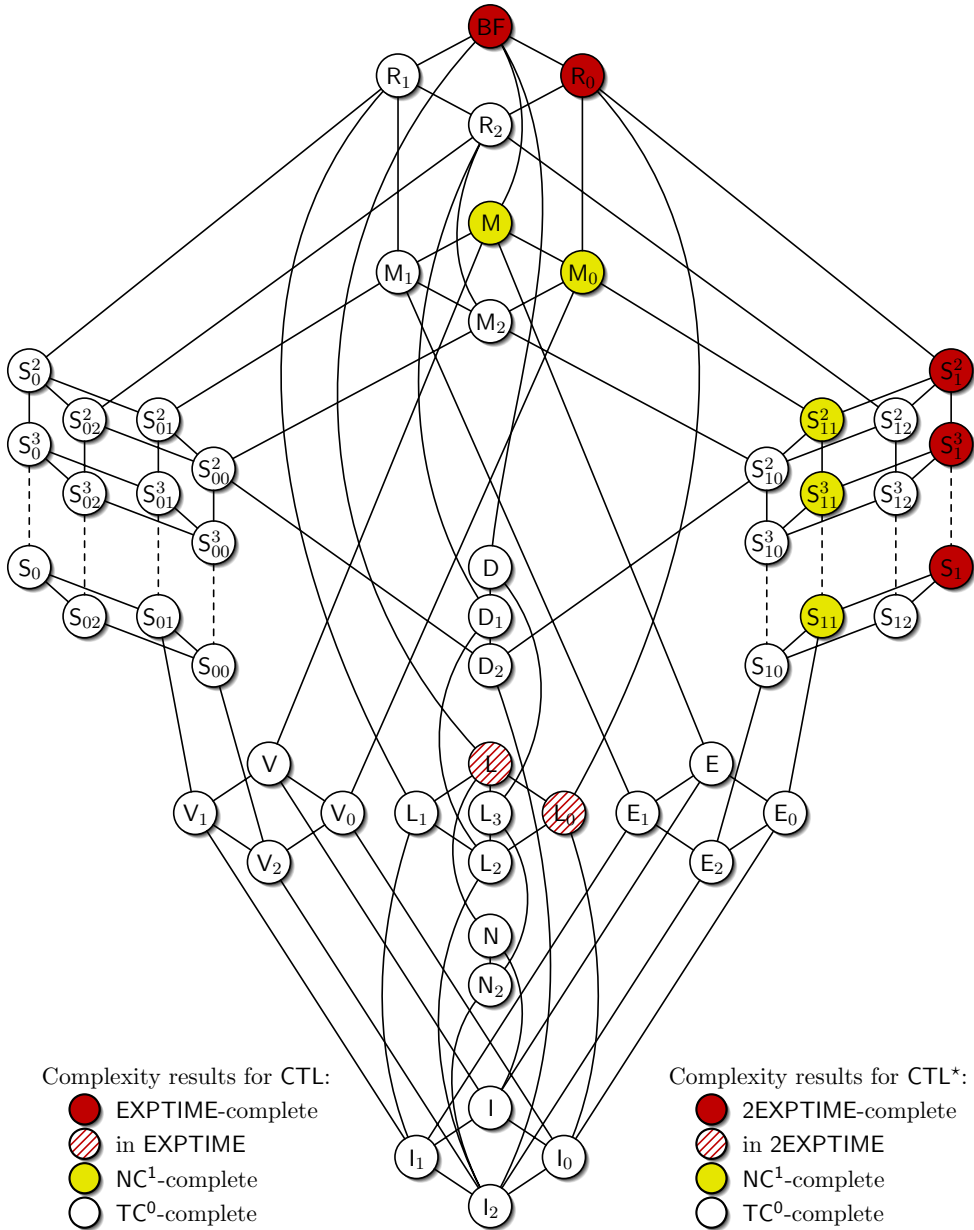
Fig. 2. Post's lattice. Colors indicate the complexity of CTL-SAT and CTL*-SAT without restrictions on the temporal operators.

that a given formula φ is satisfiable iff a quasi-model with certain properties exists. A PSPACE-upper bound then is obtained by solving a certain reachability problem in the graph of quasi-models of φ. A clever implementation of this algorithm (and algorithms for other special cases) could lead to better tools than we have today.

The rest of this paper is organized as follows. Section 2 contains preliminaries. The complexity of the satisfiability problem for the computational tree logic, CTL, is considered in Section 3. Section 4 then extends these considerations to the logic

CTL*. Section 5 concludes with a summary and a discussion.

In the interests of space, proofs are omitted or sketched. Detailed proof will be included in the full version of this paper.

2 Preliminaries

2.1 Complexity Theory

We assume familiarity with the standard notions of complexity theory. In particular, we will make use of the classes LOGSPACE, P, NP, PSPACE, EXPTIME and 2EXPTIME.

We require subtle reductions in order to obtain hardness results for complexity classes below TC^0. Therefore, we introduce the following notion of reducibility (see [21]). Let A and B be languages. Then A is *constant-depth reducible* to B ($A \leq_{\mathrm{cd}} B$) if there exists an AC^0-circuit family $\{C_n\}_{n>0}$ with $\{\wedge, \vee, \neg\}$-gates and oracle gates for B such that for all x, $C_{|x|}(x) = 1$ iff $x \in A$. One of our results even addresses complexity issues inside the class AC^0—hence $\leq_{\mathrm{cd}}$-reducibility is of no use since AC^0 forms the **0**-degree of $\leq_{\mathrm{cd}}$. Instead, we will make use of *dlt-projection reducibility* ($A \leq_{\mathrm{proj}}^{\mathrm{dlt}} B$) as introduced in [16]. We note that TC^0 and NC^1 are closed under $\leq_{\mathrm{cd}}$, and NLOGTIME and coNLOGTIME are closed under $\leq_{\mathrm{proj}}^{\mathrm{dlt}}$.

2.2 Temporal Logic

We inductively define CTL*-formulae as follows. Let Φ be a finite set of atomic propositions. The symbols used are the atomic propositions in Φ, the constant symbols $\top$ and $\bot$, the Boolean connectives $\neg$ and $\wedge$, the temporal operator symbols X, U, and A. A is also called a *path quantifier*, temporal operators aside from A are thence also called *pure temporal operators*. The atomic propositions, $\top$ and $\bot$ are called *atomic formulae*. Each atomic formula is a *state formula*, and each state formula is a *path formula*. Let φ, ψ be state formulae and χ, π be path formulae. Then (φ), $\varphi \wedge \psi$, $\neg\varphi$, Aχ are state formulae, and $\chi \wedge \pi$, $\neg\chi$, Xχ, and $[\chi\mathsf{U}\pi]$ are path formulae. The set of CTL*-formulae (or *formulae*) is the union of all state formulae and of all path formulae. We further define CTL$^\star(T, B)$ to be the set of CTL*-formulae using the Boolean connectives in B and the path quantifiers and temporal operators in T only. The set of proper subformulae of φ will be denoted by SF(φ), the number of pure temporal operators in φ by $\#_\mathsf{T}(\varphi)$.

A *model* is a triple $M = (S, R, l)$, where S is a finite set of states, $R \subseteq S \times S$ a total relation (i. e., for each $s \in S$, there exists an s' such that $(s, s') \in R$), and $l\colon S \to \mathfrak{P}(\Phi)$ is a labelling function. A *path* x is an infinite sequence $x = (x_1, x_2, \ldots) \in S^\omega$ such that $(x_i, x_{i+1}) \in R$, for all $i > 0$.

Let $M = (S, R, l)$ be a model, χ be a state formula, $s \in S$ be a state and $x = (x_1, x_2, \ldots) \in S^\omega$ be a path. The truth of a CTL*-formula w. r. t. M is inductively defined using the following semantics. Let $\varphi, \psi, \chi, \pi \in$ CTL*.

$$M, s \models \top \qquad \text{always,}$$
$$M, s \models \bot \qquad \text{never,}$$

$M, s \models p$ iff $p \in \Phi$ and $p \in l(s)$,
$M, s \models (\varphi)$ iff $M, s \models \varphi$,
$M, s \models \neg\varphi$ iff $M, s \not\models \varphi$,
$M, s \models \varphi \wedge \psi$ iff $M, s \models \varphi$ and $M, s \models \psi$,
$M, s \models \mathsf{A}\chi$ iff for all paths $x = (x_1, x_2, \dots)$ with $x_1 = s$ holds $M, x \models \chi$,
$M, x \models \chi$ iff $M, x_1 \models \chi$,
$M, x \models \mathsf{X}\chi$ iff $M, x_2 \models \chi$,
$M, x \models [\chi\mathsf{U}\pi]$ iff $M, x_k \models \pi$, for some $k \in \mathbb{N}$, and $M, x_i \models \chi$, for all $1 \le i < k$.

The syntax and semantics of each remaining Boolean function f can be expressed through the connectives $\wedge$ and $\neg$. The remaining temporal operators are defined in the following way:

$$\mathsf{E}\varphi \equiv \neg\mathsf{A}\neg\varphi, \quad \mathsf{F}\varphi \equiv \top\mathsf{U}\varphi, \quad \mathsf{G}\varphi \equiv \neg\mathsf{F}\neg\varphi,$$

where E is again also called a path quantifier. A state formula φ is hence said to be *satisfied by model M* if there exists an $s \in S$ such that $M, s \models \varphi$ (written as $M \models \varphi$). Analogously, a path formula is said to be satisfied by a model M if there exists a path $x = (x_1, x_2, \dots)$ such that $M, x \models \varphi$. Finally φ is said to be *satisfiable* if there exists a model M that satisfies φ. We define $\mathsf{CTL}^\star\text{-SAT}(T, B)$ to be the problem of deciding whether a given $\mathsf{CTL}^\star(T, B)$-formula is satisfiable.

A CTL-*formula* is a $\mathsf{CTL}^\star$-formula in which each path quantifier is followed by exactly one pure temporal operator and each pure temporal operator is preceded by exactly one path quantifier. The set of CTL-formulae is a strict subset of the set of $\mathsf{CTL}^\star$-formulae. For example, $\mathsf{AGEF}p$ is a CTL-formula, whereas $\mathsf{A}(\mathsf{GF}p \to \mathsf{F}q)$ is not. Pairs of path quantifiers and pure temporal operators are thence also referred to as CTL-*operators*. Let ALL denote the set of all CTL-operators. We remark the following dualities among CTL-operators:

$$\mathsf{EX}\varphi \equiv \neg\mathsf{AX}\neg\varphi,\ \mathsf{EF}\varphi \equiv \mathsf{E}[\top\mathsf{U}\varphi],\ \mathsf{AF}\varphi \equiv \mathsf{A}[\top\mathsf{U}\varphi],\ \mathsf{AG}\varphi \equiv \neg\mathsf{EF}\neg\varphi,\ \mathsf{EG}\varphi \equiv \neg\mathsf{AF}\neg\varphi,$$

and $\mathsf{A}[\psi\mathsf{U}\chi] \equiv \mathsf{AF}\chi \wedge \neg\mathsf{E}[\neg\chi\mathsf{U}(\neg\psi \wedge \neg\chi)]$. Hence $\{\mathsf{AX}, \mathsf{AF}, \mathsf{EU}\}$ is a minimal set of CTL-operators for CTL (in presence of all Boolean connectives), whereas $\{\mathsf{AX}, \mathsf{AG}, \mathsf{AU}\}$ is not complete for CTL [10]. Alike $\mathsf{CTL}^\star\text{-SAT}$, we define $\mathsf{CTL}(T, B)$ to be the set of all CTL-formulae using the CTL-operators in T and the Boolean connectives in B only, and define $\mathsf{CTL\text{-}SAT}(T, B)$ to be the problem of deciding whether a given $\mathsf{CTL}(T, B)$-formula is satisfiable.

2.3 Boolean Clones

Since there are infinitely many finite sets B of Boolean functions, we introduce some algebraic tools to classify the complexity of the infinitely many arising satisfiability problems. A set B of Boolean functions is called a *clone* if it is closed under superposition, which means B contains all projections and B is closed under arbitrary composition [14, Chapter 1]. For a set B of Boolean functions we denote with $[B]$ the smallest clone containing B and call B a base for $[B]$. In [15] Post classified the

lattice of all clones and found a finite base for each clone, see Fig. 2. In order to introduce the clones relevant to this paper, we define the following notions, where f is an n-ary Boolean function.

- f is 1-*reproducing* if $f(1, \ldots, 1) = 1$.
- f is *monotone* if $a_1 \le b_1, a_2 \le b_2, \ldots, a_n \le b_n \implies f(a_1, \ldots, a_n) \le f(b_1, \ldots, b_n)$.
- f is 1-*separating* if there exists an $i \in \{1, \ldots, n\}$ such that $f(a_1, \ldots, a_n) = 1$ implies $a_i = 1$.
- f is *self-dual* if $f \equiv \mathrm{dual}(f)$, where $\mathrm{dual}(f)(x_1, \ldots, x_n) = \neg f(\neg x_1, \ldots, \neg x_n)$.
- f is *linear* if $f \equiv x_1 \oplus \cdots \oplus x_n \oplus c$ for a constant $c \in 0, 1$ and variables $x_1, \ldots, x_n$.

The clones relevant to this paper are listed in Table 1. The definition of all Boolean clones can be found, e. g., in [3].

Name	Definition	Base
BF	All Boolean functions	$\{\wedge, \neg\}$
R_1	$\{f : f \text{ is 1-reproducing}\}$	$\{\vee, \rightarrow\}$
M	$\{f : f \text{ is monotone}\}$	$\{\vee, \wedge, \bot, \top\}$
S_1	$\{f : f \text{ is 1-separating}\}$	$\{x \wedge \overline{y}\}$
D	$\{f : f \text{ is self-dual}\}$	$\{x\overline{y} \vee x\overline{z} \vee (\overline{y} \wedge \overline{z})\}$
L	$\{f : f \text{ is linear}\}$	$\{\oplus, \bot\}$
V	$\{f : f \equiv c_0 \vee \bigvee_{i=1}^{n} c_i x_i \text{ where the } c_i \text{s are constant}\}$	$\{\vee, \bot, \top\}$
V_0	$[\{\vee\}] \cup [\{\bot\}]$	$\{\vee, \bot\}$
E	$\{f : f \equiv c_0 \wedge \bigwedge_{i=1}^{n} c_i x_i \text{ where the } c_i \text{s are constant}\}$	$\{\wedge, \bot, \top\}$
E_0	$[\{\wedge\}] \cup [\{\bot\}]$	$\{\wedge, \bot\}$
N	$\{f : f \text{ depends on at most one variable}\}$	$\{\neg, \bot, \top\}$
I	$\{f : f \text{ is a projection or a constant}\}$	$\{\mathrm{id}, \bot, \top\}$

Table 1
A list of Boolean clones with definitions and bases.

3 Satisfiability in CTL

We commence by considering the complexity of the satisfiability problem for restricted sets of the CTL-operators and continue with restricted sets of Boolean functions. Recall the previously known results.

Theorem 3.1 ([5], [8]) (i) CTL-SAT$(\emptyset, \mathsf{BF})$ *is* NP-*complete*.
(ii) CTL-SAT$(\{\mathsf{AX}, \mathsf{AU}, \mathsf{EU}\}, \mathsf{BF})$ *is* EXPTIME-*complete*.

3.1   Restricting the CTL-operators

Theorem 3.2 *Let T be a set of* CTL*-operators. Then* CTL-SAT(T, BF) *is*

  (i) NP-*complete under* $\leq_{\mathrm{cd}}$-*reductions if* $T = \{\mathsf{AF}\}$,

 (ii) PSPACE-*complete under* $\leq_{\mathrm{cd}}$-*reductions if* $T = \{\mathsf{AG}\}, \{\mathsf{AX}\}, \{\mathsf{AF},\mathsf{AG}\}, \{\mathsf{AX},\mathsf{AF}\}$,

(iii) EXPTIME-*complete under* $\leq_{\mathrm{cd}}$-*reductions in all other cases.*

Proof (Sketch) For (i), NP-hardness of CTL-SAT$(\{\mathsf{AF}, \mathsf{BF}\})$ is immediate from Theorem 3.1(i). The membership in NP follows from a small model property: $\varphi \in$ CTL$(\{\mathsf{AF}, \mathsf{BF}\})$ is satisfiable iff φ is satisfiable in a model of size $\leq |\varphi|^{O(1)}$. As the model-checking problem for CTL is polynomial-time solvable [4], we can hence simply guess a model M and check whether $M \models \varphi$.

As for (ii), it suffices to show PSPACE-hardness for $T = \{\mathsf{AG}\}, \{\mathsf{AX}\}$, and membership in PSPACE for $T = \{\mathsf{AF}, \mathsf{AG}\}, \{\mathsf{AX}, \mathsf{AF}\}$. The hardness for both $T = \{\mathsf{AG}\}$, and $T = \{\mathsf{AX}\}$ is established using similar $\leq_{\mathrm{cd}}$-reductions f_{AX} and f_{AG} from the satisfiability problem for quantified Boolean formulae, QBF-SAT. For $\varphi \in$ QBF-SAT, the constructed formula $f(\varphi)$ forces any satisfying model to encode in a tree-like structure the set of assignments necessary to fulfill φ. Both proofs are similar to [9, Theorem 3.1]; a proof for CTL-SAT$(\{\mathsf{AX}\}, \mathsf{BF}) \in$ PSPACE can also be found in [13, Theorem 9].

Now consider $T = \{\mathsf{AF}, \mathsf{AG}\}$. To show memberhip in PSPACE, we present an algorithm inspired by the algorithm showing that provability in the modal logic K is in PSPACE [9]. The algorithm is based on the notion of *quasi models*: let $\varphi \in$ CTL$(\{\mathsf{AF}, \mathsf{AG}\}, \mathsf{BF})$ be in negation normal form (i.e., negations occur in front of atomic formulae only), a quasi model for φ is a model $M = (S, R, l)$ with labels $l : S \to \mathfrak{P}(\mathrm{CTL}(\{\mathsf{AF}, \mathsf{AG}\}, \mathsf{BF}))$ such that

  (i) for all $s \in S$, $l(s)$ are minimal sets satisfying the condition that $\psi \wedge \chi \in l(s)$ implies $\psi \in l(s)$ and $\chi \in l(s)$, and the condition that $\psi \vee \chi \in l(s)$ implies $\psi \in l(s)$ or $\chi \in l(s)$,

 (ii) $\varphi \in l(s)$ for some $s \in S$.

(iii) for all $s \in S$ and each $\mathcal{O}\psi \in l(s)$, "M satisfies the constraints imposed by $\mathcal{O}\psi$", i.e., $\mathcal{O} \in \{\mathsf{AF}, \mathsf{EF}, \mathsf{AG}, \mathsf{EG}\}$, $\psi \in l(x_i)$ on all/some paths $x = (x_1, x_2, x_3, \ldots)$, $x_1 = s$, and all/some $1 \leq i \in \mathbb{N}$.

Note that the labels of quasi models bear resemblance to Hintikka sets but differ from the latter in that they are allowed to contain $\bot$. The algorithm is based on the following observation: φ is satisfiable iff there is a quasi-model for φ whose labels are consistent on all path prefixes of linear length from some $s \in S$ with $\varphi \in l(s)$. Thence the algorithm performs a nondeterministic depth-first search for contradictions on the set of quasi models for φ. The space-bound derives from the linear length of path prefixes to be investigated.

For $T = \{\mathsf{AX}, \mathsf{AF}\}$, a straightforward modification of the former algorithm is not possible, since the X operator allows for the construction of "counters" such that contradictions may occur in exponential depth firstly. Nevertheless,

any $\mathsf{CTL}(\{\mathsf{AX}, \mathsf{AF}\}, \mathsf{BF})$-formula may impose at most linearly many temporal constraints. Using the fixpoint-characterisation of EG, we derive an algorithm for $\varphi \in \mathsf{CTL}(\{\mathsf{AX}, \mathsf{AF}\}, \mathsf{BF})$ in a two-steps approach: first verify that φ with all EG operators ignored is satisfiable, then test each of the EG-prefixed subformulae for satisfiability separately.

Finally for (iii), membership in $\mathsf{EXPTIME}$ is due to Theorem 3.1(iii). Hardness for $\mathsf{EXPTIME}$ is obtained from reducing the word problem for polynomial-space alternating Turing machines to $\mathsf{CTL\text{-}SAT}(T, \mathsf{BF})$. The reduction for $T = \{\mathsf{AX}, \mathsf{AG}\}$ is straightforward and can then be modified to prove hardness for the cases $T = \{\mathsf{AU}\}$ and $T = \{\mathsf{EU}\}$, too. The hardness of the remaining fragments follows. $\square$

3.2   Restricting the Boolean connectives

Say that a set B of Boolean connectives is *non-trivial* if B contains a connective of arity ≥ 2. We state an auxiliary lemma.

Lemma 3.3 *Let B be a non-trivial set of Boolean function symbols and let T be a set of CTL-operators. The problem to decide, whether a given string is a $\mathsf{CTL}(T, B)$-formula, is complete for TC^0 under $\leq_{\mathrm{cd}}$-reductions.*

Theorem 3.4 *Let T denote a set of CTL-operators and let B be a set of Boolean functions such that $[B] \notin \{\mathsf{L}, \mathsf{L}_0\}$ and B is non-trivial. Then $\mathsf{CTL\text{-}SAT}(T, B)$ is*

(i) *equivalent to $\mathsf{CTL\text{-}SAT}(T, \mathsf{BF})$ if $\mathsf{S}_1 \subseteq [B]$,*

(ii) NC^1*-complete under $\leq_{\mathrm{cd}}$-reductions if $\mathsf{S}_{11} \subseteq [B] \subseteq \mathsf{M}$, and*

(iii) TC^0*-complete under $\leq_{\mathrm{cd}}$-reductions in all other cases.*

Proof (Sketch) For (i), note that $\mathsf{BF} = [\mathsf{S}_1 \cup \{\top\}] = [B \cup \{\top\}]$. It hence suffices to show that we can generate the constant $\top$ in all sets of Boolean functions B satisfying $[B] = \mathsf{S}_1$.

For (ii), $[B]$ does not contain negations. Hence we can substitute each atomic proposition with $\top$ and evaluate this proposition-free formula alike propositional formulae [17, Theorem 3.2]. As evaluation of propositional S_{11}-formula is NC^1-complete already, the claim follows.

For (iii), we have to distinguish between two cases. First consider the cases $[B] \subseteq \mathsf{R}_1$ and $[B] \subseteq \mathsf{D}$. An induction on the formulae structure shows that all formulae are trivially satisfiable by the model $M = (\{s\}, \{(s, s)\}, l)$, where, for all $s \in S$, either $l(s) = \Phi$ or $l(s) = \emptyset$. If $[B] \subseteq \mathsf{N}$, we can w.l.o.g. transform the given formula φ to be of the form

$$\varphi \equiv \mathcal{O}_1 \mathcal{O}_2 \cdots \mathcal{O}_k \mathcal{P}_1 \Big[\psi \mathsf{U} \mathcal{P}_2 \big[\cdots \mathsf{U} \mathcal{P}_l [\cdots \mathsf{U} \psi'] \cdots \big] \Big],$$

where $\psi \in \mathsf{CTL}(T, B)$, $\psi' \in \mathsf{CTL}(T \setminus \{\mathsf{AU}, \mathsf{EU}\}, B)$, $\mathcal{O}_1, \ldots, \mathcal{O}_k \in T \setminus \{\mathsf{AU}, \mathsf{EU}\}$ and $\mathcal{P}_1, \ldots, \mathcal{P}_l \in \{\mathsf{A}, \mathsf{E}\}$. Hence we only need to count the number of preceding negations of ψ' modulo 2. For the remaining clones we can substitute the propositions with $\top$ and only need to search for a $\top$ (in the $\vee$-case), or ensure absence of $\bot$ (in the

$\wedge$-case). Having established membership in $\mathsf{TC^0}$, completeness for $\mathsf{TC^0}$ stems from Lemma 3.3. $\square$

That is, for the last case of Theorem 3.4, the main complexity thus lies in checking the syntactical correctness of the given formula. In order to classify the complexity of $\mathsf{CTL\text{-}SAT}(T, B)$ beyond the complexity of its syntactical correctness, we restrict our attention to syntactically correct input formulae: Let $\mathsf{CTL\text{-}SAT}_{\mathcal{P}}(T, B)$ denote the promise problem of deciding whether a given syntactically correct $\mathsf{CTL}(T, B)$-formula is satisfiable. The following theorem refines Theorem 3.4 for subclasses of $\mathsf{TC^0}$.

Theorem 3.5 *Let T denote a set of CTL-operators and let B be a set of Boolean functions such that $\mathsf{CTL\text{-}SAT}(T, B)$ is $\mathsf{TC^0}$-complete. Then $\mathsf{CTL\text{-}SAT}_{\mathcal{P}}(T, B)$ is*

(i) *in $\mathsf{TC^0}$ if $T \cap \{\mathsf{AU}, \mathsf{EU}\} \neq \emptyset$ and $[B] \in \{\mathsf{V}, \mathsf{V_0}, \mathsf{E}, \mathsf{E_0}, \mathsf{N}\}$,*

(ii) *$\mathsf{NLOGTIME}$-complete under $\leq^{\mathrm{dlt}}_{\mathrm{proj}}$-reductions if $T \cap \{\mathsf{AU}, \mathsf{EU}\} = \emptyset$ and $[B] \in \{\mathsf{V}, \mathsf{V_0}\}$,*

(iii) *$\mathsf{coNLOGTIME}$-complete under $\leq^{\mathrm{dlt}}_{\mathrm{proj}}$-reductions if $T \cap \{\mathsf{AU}, \mathsf{EU}\} = \emptyset$ and $[B] \in \{\mathsf{E}, \mathsf{E_0}\}$,*

(iv) *equivalent to MOD_2 under $\leq^{\mathrm{dlt}}_{\mathrm{proj}}$-reductions if $T \cap \{\mathsf{AU}, \mathsf{EU}\} = \emptyset$ and $[B] = \mathsf{N}$, and*

(v) *trivial in all other cases.*

Proof (Sketch) For (i), one has to determine the relevant parts of the formula first. This requires counting the parentheses, therefore the problem remains in $\mathsf{TC^0}$.

The cases (ii) and (iii) can be solved analogously to [17, Lemma 3.7], that is, by guessing the position of a satisfying $\top$ (or a falsifying $\bot$, resp.) after substituting all propositions with $\top$. Hardness is obtained via a reduction from the language $\{0, 1\}^\star 1\{0, 1\}$ (or $\{0, 1\}^\star$, resp.).

For (iv), syntactically correct formulae in $\mathsf{CTL}(T, \mathsf{N})$ can be checked for satisfiability by just counting the preceding negations modulo 2, for all temporal operators and Boolean connectives are unary. Hardness for this case arises from a reduction from $\mathrm{PARITY} = \{w \in \{0, 1\}^\star \mid |w|_1 \equiv 1 \mod 2\}$.

Lastly, for any other combination of T and B, all $\mathsf{CTL}(T, B)$-formulae are trivially satisfiable. $\square$

4   Satisfiability in CTL*

Having classified the complexity of the satisfiability problem for CTL, we now turn to $\mathsf{CTL}^\star$, a logic strictly more expressive than CTL: instead of paired, path quantifiers and temporal operators may occur independently of each other. This fact amounts to a jump in the complexity in the general case.

Theorem 4.1 ([19]) $\mathsf{CTL}^\star\text{-}\mathsf{SAT}(\{\mathsf{A}, \mathsf{X}, \mathsf{U}\}, \mathsf{BF})$ *is $\mathsf{2EXPTIME}$-complete.*

We will hence proceed analogously to Section 3 and consider the complexity of the satisfiability problem for restricted sets of path quantifiers and temporal

operators and restricted sets of Boolean functions.

4.1   Restricting the temporal operators and path quantifiers

Theorem 4.2 *Let T be a set of temporal operators. Then* $\mathsf{CTL}^\star\text{-}\mathsf{SAT}(T, \mathsf{BF})$ *is*

(i) NP*-complete under* $\leq_{\mathrm{cd}}$*-reductions if* $T = \emptyset, \{\mathsf{A}\}, \{\mathsf{F}\}, \{\mathsf{X}\}$,

(ii) PSPACE*-complete under* $\leq_{\mathrm{cd}}$*-reductions if* $T = \{\mathsf{U}\}, \{\mathsf{X},\mathsf{F}\}, \{\mathsf{X},\mathsf{U}\}, \{\mathsf{A},\mathsf{X}\}, \{\mathsf{A},\mathsf{F}\}$,

(iii) $\mathsf{2EXPTIME}$*-complete under* $\leq_{\mathrm{cd}}$*-reductions in all other cases.*

Proof (Sketch) NP-completeness for (i) and the first three cases from (ii) follows from [2]—these are LTL-formulae.

The remaining two restricted sets in (ii) can be proven by a similar reduction and algorithm as for the CTL-cases.

For (iii), we modify the hardness part of Vardi's proof showing that $\mathsf{CTL}^\star\text{-}\mathsf{SAT}$ restricted to $\{\mathsf{A}, \mathsf{X}, \mathsf{U}\}$ and BF is $\mathsf{2EXPTIME}$-complete [20]. Vardi gives a reduction from the word problem for exponential-space alternating Turing machines to $\mathsf{CTL}^\star\text{-}\mathsf{SAT}(\{\mathsf{A}, \mathsf{X}, \mathsf{U}\}, \mathsf{BF})$. We restate the formulae in this reduction using either the temporal operators A, X and F, or the temporal operators A and U only.   □

4.2   Restricting the Boolean connectives

Theorem 4.3 *Let T denote a set of temporal operators and let B be a set of Boolean functions such that* $[B] \notin \{\mathsf{L}, \mathsf{L}_0\}$. *Then* $\mathsf{CTL}^\star\text{-}\mathsf{SAT}(T, B)$ *is*

(i) *equivalent to* $\mathsf{CTL}^\star\text{-}\mathsf{SAT}(T, \mathsf{BF})$ *if* $\mathsf{S}_1 \subseteq [B]$,

(ii) NC^1*-complete under* $\leq_{\mathrm{cd}}$*-reductions if* $\mathsf{S}_{11} \subseteq [B] \subseteq \mathsf{M}$, *and*

(iii) TC^0*-complete under* $\leq_{\mathrm{cd}}$*-reductions in all other cases.*

Proof. The results of Section 3 are easily generalized to $\mathsf{CTL}^\star\text{-}\mathsf{SAT}$.   □

5   Conclusion

The complexity of the satisfiability problem for temporal-operator-restricted fragments of CTL and $\mathsf{CTL}^\star$ is a trichotomy: for CTL we classified completeness for $\mathsf{EXPTIME}$, PSPACE and NP, and for $\mathsf{CTL}^\star$ we classified completeness for $\mathsf{2EXPTIME}$, PSPACE and NP. This situation is depicted as a lattice in Figure 1.

Concerning the restrictions on the set of Boolean functions we observe a tetrachotomy: a line from BF down to S_1, whose complexity is determined by the temporal operators we allow, a similar line of NC^1-complete clones from M down to S_{11}, the two clones L and L_0, whose complexity is bounded above by the complexity for the clone BF, and the remaining clones—all of which are TC^0-complete. The complete lattice is shown in Figure 2.

Hence, the complexity of the satisfiability problems increases along the same edges as it does in propositional logic. In particular, if $x \not\to y$ can be implemented then satisfiability is as difficult as if all Boolean connectives were available, whereas

else the complexity of CTL-SAT(T, B) drops to a very low class inside NC^1 and is particularly independent of the temporal operators (except for the clones L and L_0).

For CTL-SAT$_\mathcal{P}$, the satisfiability problem restricted to syntactically correct formulae, the TC^0-complete clones R_1 and D are trivially satisfiable, while the clones N, V, V_0, E, and E_0 yield complexity results depending on the set of CTL-operators allowed: Let B be a set of Boolean functions such that $[B] \in \mathsf{V}, \mathsf{V}_0, \mathsf{E}, \mathsf{E}_0, \mathsf{N}$ and let T be a set of CTL-operators not containing AU and EU ($T \cap \{\mathsf{AU}, \mathsf{EU}\} = \emptyset$). Then CTL-SAT$_\mathcal{P}(T, B)$ is solvable in TC^0. Otherwise, if $T \cap \{\mathsf{AU}, \mathsf{EU}\} = \emptyset$ then CTL-SAT$_\mathcal{P}(T, \mathsf{V})$ and CTL-SAT$_\mathcal{P}(T, \mathsf{V}_0)$ are $\mathsf{NLOGTIME}$-complete; whereas CTL-SAT$_\mathcal{P}(T, \mathsf{E})$ and CTL-SAT$_\mathcal{P}(T, \mathsf{E}_0)$ are $\mathsf{coNLOGTIME}$-complete. Finally, the promise problem CTL-SAT$_\mathcal{P}(T, \mathsf{N})$ is equivalent to MOD_2 under $\leq_{\mathrm{proj}}^{\mathrm{dlt}}$-reductions.

For CTL-SAT$_\mathcal{P}(T, B)$, $[B] \in \{\mathsf{V}, \mathsf{V}_0, \mathsf{E}, \mathsf{E}_0, \mathsf{N}\}$ and $T \cap \{\mathsf{AU}, \mathsf{EU}\} \neq \emptyset$, the gap between membership in TC^0 and hardness for $\mathsf{NLOGTIME}$ (resp. $\mathsf{coNLOGTIME}$ or $\mathsf{AC}^0[2]$) results—intuitively speaking—from the unfortunate circumstance that the given formula is promised to be syntactically correct, but determining the influence of some literal on the satisfyability yet has to be derived from the syntactical structure. On the one hand, determining whether some guessed literal or constant is relevant to the satisfiability of some formula seems to require the capability of counting; on the other hand, it seem unlikely that some TC^0-complete is reducible to CTL-SAT$_\mathcal{P}(T, B)$. Analogous results were obtained for full branching time logic CTL*.

Finally, the complexity of CTL-SAT$(T, [B])$ and CTL*-SAT$(T, [B])$ for $[B] \in \{\mathsf{L}, \mathsf{L}_0\}$ remains unclassified. Though we obtained membership in P for CTL-SAT(T, B) if $T = \{\mathsf{AX}\}$ or $T = \{\mathsf{AG}\}$ (the results will be included in the full version of this paper), the interplay of linearity and temporal operators eluded CTL-SAT(T, B) from a detailed analysis for the remaining cases. Their complexity remains an open question. Note that the result for CTL-SAT$(\{\mathsf{AG}\}, B)$ states the hitherto first subexponential upper bound for a reflexive temporal operator in connection with the $\oplus$-function.

Further work should, besides closing the just mentioned complexity gap, address a detailed analysis of the model checking problem for fragments of CTL*, as begun in [1] for LTL. The fragment CTL is known to generally have an efficient (polynomial time decidable) model checking problem; we consider it very interesting to determine here for which fragments space efficient or parallel algorithms exist.

References

[1] Bauland, M., M. Mundhenk, T. Schneider, H. Schnoor, I. Schnoor and H. Vollmer, *The tractability of model checking for LTL: the good, the bad, and the ugly fragments*, in: *Proceedings Methods for Modalities 5* (2007), pp. 125–140.

[2] Bauland, M., T. Schneider, H. Schnoor, I. Schnoor and H. Vollmer, *The complexity of generalized satisfiability for linear temporal logic*, in: *Proceedings of the Foundations of Software Science and Computation Structures*, Lecture Notes in Computer Science **4423** (2007), pp. 48–62.

[3] Böhler, E., N. Creignou, S. Reith and H. Vollmer, *Playing with Boolean blocks, part I: Post's lattice with applications to complexity theory*, SIGACT News **34** (2003), pp. 38–52.

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

[5] Cook, S. A., *The complexity of theorem proving procedures*, in: *Proceedings 3rd Symposium on Theory of Computing* (1971), pp. 151–158.

[6] Emerson, E. A., "Temporal and Modal Logic," Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics, MIT Press, Cambridge, MA, USA, 1990 pp. 995–1072.

[7] Emerson, E. A. and J. Y. Halpern, *"sometimes" and "not never" revisited: On branching versus linear time*, in: *Proceedings Symposium on Principles of Programming Languages* (1983), pp. 127–140.

[8] Fischer, M. J. and R. E. Ladner, *Propositional modal logic of programs*, Journal of Computer and Systems Sciences **18** (1979), pp. 194–211.

[9] Ladner, R., *The computational complexity of provability in systems of modal propositional logic*, SIAM Journal on Computing **6** (1977), pp. 467–480.

[10] Laroussinie, F., *About the expressive power of CTL combinators*, Information Processing Letters **54** (1995), pp. 343–345.

[11] Lewis, H., *Satisfiability problems for propositional calculi*, Mathematical Systems Theory **13** (1979), pp. 45–53.

[12] Markey, N., *Past is for free: on the complexity of verifying linear temporal properties with past*, Acta Informatica **40** (2004), pp. 431–458.

[13] Meier, A., "Complexity of Temporal Logics," Master's thesis, Gottfried Wilhelm Leibniz Universität Hannover (2007).

[14] Pippenger, N., "Theories of Computability," Cambridge University Press, Cambridge, 1997.

[15] Post, E., *The two-valued iterative systems of mathematical logic*, Annals of Mathematical Studies **5** (1941), pp. 1–122.

[16] Regan, K. and H. Vollmer, *Gap-languages and log-time complexity classes*, Theoretical Computer Science **188** (1997), pp. 101–116.

[17] Schnoor, H., *The complexity of the Boolean formula value problem*, Technical report, Theoretical Computer Science, University of Hannover (2005).

[18] Sistla, A. and E. Clarke, *The complexity of propositional linear temporal logics*, Journal of the ACM **32** (1985), pp. 733–749.

[19] Vardi, M. Y. and L. Stockmeyer, *Improved upper and lower bounds for modal logics of programs: Preliminary report*, in: *STOC '85: Proceedings of the seventeenth annual ACM symposium on Theory of computing*, Lecture Notes in Computer Science, 1985, pp. 240–251.

[20] Vardi, M. Y. and L. Stockmeyer, *Lower bound in full (2EXPTIME-hardness for CTL*-SAT)*, Online, available at http://www.cs.rice.edu/~vardi/papers/ctl_star_lower_bound.pdf (1985).

[21] Vollmer, H., "Introduction to Circuit Complexity – A Uniform Approach," Texts in Theoretical Computer Science, Springer Verlag, Berlin Heidelberg, 1999.