HyperLTL Satisfiability Is Σ^1_1-Complete, HyperCTL* Satisfiability Is Σ^2_1-Complete

Marie Fortin
University of Liverpool, UK

Louwe B. Kuijer
University of Liverpool, UK

Patrick Totzke
University of Liverpool, UK

Martin Zimmermann
University of Liverpool, UK

Abstract

Temporal logics for the specification of information-flow properties are able to express relations between multiple executions of a system. The two most important such logics are HyperLTL and HyperCTL*, which generalise LTL and CTL* by trace quantification. It is known that this expressiveness comes at a price, i.e. satisfiability is undecidable for both logics.

In this paper we settle the exact complexity of these problems, showing that both are in fact highly undecidable: we prove that HyperLTL satisfiability is Σ^1_1-complete and HyperCTL* satisfiability is Σ^2_1-complete. These are significant increases over the previously known lower bounds and the first upper bounds. To prove Σ^2_1-membership for HyperCTL*, we prove that every satisfiable HyperCTL* sentence has a model that is equinumerous to the continuum, the first upper bound of this kind. We prove this bound to be tight. Finally, we show that the membership problem for every level of the HyperLTL quantifier alternation hierarchy is Π^1_1-complete.

2012 ACM Subject Classification Theory of computation → Logic and verification; Theory of computation → Formal languages and automata theory

Keywords and phrases HyperLTL, HyperCTL*, Satisfiability, Analytical Hierarchy

Digital Object Identifier 10.4230/LIPIcs.MFCS.2021.47

Related Version *Extended Version*: `https://arxiv.org/abs/2105.04176` [27]

Funding Partially funded by EPSRC grants EP/S032207/1 and EP/V025848/1.

Acknowledgements We thank Karoliina Lehtinen and Wolfgang Thomas for fruitful discussions.

1 Introduction

Most classical temporal logics like LTL and CTL* refer to a single execution trace at a time while information-flow properties, which are crucial for security-critical systems, require reasoning about multiple executions of a system. Clarkson and Schneider [13] coined the term *hyperproperties* for such properties which, structurally, are sets *of sets* of traces. Just like ordinary trace and branching-time properties, hyperproperties can be specified using temporal logics, e.g. HyperLTL and HyperCTL* [12], expressive, but intuitive specification languages that are able to express typical information-flow properties such as noninterference, noninference, declassification, and input determinism. Due to their practical relevance and theoretical elegance, hyperproperties and their specification languages have received considerable attention during the last decade [1, 2, 5, 6, 7, 10, 12, 13, 14, 16, 26, 28, 31, 32, 39].

HyperLTL is obtained by extending LTL [34], the most influential specification language for linear-time properties, by trace quantifiers to refer to multiple executions of a system. For example, the HyperLTL formula

$$\forall \pi.\ \forall \pi'.\ \mathbf{G}(i_\pi \leftrightarrow i_{\pi'}) \rightarrow \mathbf{G}(o_\pi \leftrightarrow o_{\pi'})$$

expresses input determinism, i.e. every pair of traces that always has the same input (represented by the proposition i) also always has the same output (represented by the proposition o). Similarly, HyperCTL* is the extension of the branching-time logic CTL* [17] by path quantifiers. HyperLTL only allows formulas in prenex normal form while HyperCTL* allows arbitrary quantification, in particular under the scope of temporal operators. Consequently, HyperLTL formulas are evaluated over sets of traces while HyperCTL* formulas are evaluated over transition systems, which yield the underlying branching structure of the traces.

All basic verification problems, e.g. model checking [18, 25], runtime monitoring [3, 8, 11, 24], and synthesis [9, 21, 22], have been studied. Most importantly, HyperCTL* model checking over finite transition systems is decidable and TOWER-complete for a fixed transition system [25, 33]. However, for a small number of alternations, efficient algorithms have been developed and were applied to a wide range of problems, e.g. an information-flow analysis of an I2C bus master [25], the symmetric access to a shared resource in a mutual exclusion protocol [25], and to detect the use of a defeat device to cheat in emission testing [4].

But surprisingly, the exact complexity of the satisfiability problems for HyperLTL and HyperCTL* is still open. Finkbeiner and Hahn proved that HyperLTL satisfiability is undecidable [19], a result which already holds when only considering finite sets of ultimately periodic traces and $\forall\exists$-formulas. In fact, Finkbeiner et al. showed that HyperLTL satisfiability restricted to finite sets of ultimately periodic traces is Σ_1^0-complete [20] (i.e. complete for the set of recursively enumerable problems). Furthermore, Hahn and Finkbeiner proved that the $\exists^*\forall^*$-fragment has decidable satisfiability [19] while Mascle and Zimmermann studied the HyperLTL satisfiability problem restricted to bounded sets of traces [33]. The latter work implies that HyperLTL satisfiability restricted to finite sets of traces (even non ultimately periodic ones) is also Σ_1^0-complete. Finally, Finkbeiner et al. developed tools and heuristics [20, 23].

As every HyperLTL formula can be turned into an equisatisfiable HyperCTL* formula, HyperCTL* satisfiability is also undecidable. Moreover, Rabe has shown that it is even Σ_1^1-hard [35], i.e. it is not even arithmetical. However, both for HyperLTL and for HyperCTL* satisfiability, only lower bounds, but no upper bounds, are known.

Our Contributions. In this paper, we settle the complexity of the satisfiability problems for HyperLTL and HyperCTL* by determining exactly how undecidable they are. That is, we provide matching lower and upper bounds in terms of the analytical hierarchy and beyond, where decision problems (encoded as subsets of $\mathbb{N}$) are classified based on their definability by formulas of higher-order arithmetic, namely by the type of objects one can quantify over and by the number of alternations of such quantifiers. We refer to Roger's textbook [36] for fully formal definitions. For our purposes, it suffices to recall the following classes. Σ_1^0 contains the sets of natural numbers of the form

$$\{x \in \mathbb{N} \mid \exists x_0 \cdots \exists x_k.\ \psi(x, x_0, \ldots, x_k)\}$$

where quantifiers range over natural numbers and ψ is a quantifier-free arithmetic formula. The notation Σ_1^0 signifies that there is a single block of existential quantifiers (the subscript 1) ranging over natural numbers (type 0 objects, explaining the superscript 0). Analogously, Σ_1^1

is induced by arithmetic formulas with existential quantification of type 1 objects (functions mapping natural numbers to natural numbers) and arbitrary (universal and existential) quantification of type 0 objects. Finally, Σ^2_1 is induced by arithmetic formulas with existential quantification of type 2 objects (functions mapping type 1 objects to natural numbers) and arbitrary quantification of type 0 and type 1 objects. So, Σ^0_1 is part of the first level of the arithmetic hierarchy, Σ^1_1 is part of the first level of the analytical hierarchy, while Σ^2_1 is not even analytical.

In terms of this classification, we prove that HyperLTL satisfiability is Σ^1_1-complete while HyperCTL* satisfiability is Σ^2_1-complete, thereby settling the complexity of both problems and showing that they are highly undecidable. In both cases, this is a significant increase of the lower bound and the first upper bound.

First, let us consider HyperLTL satisfiability. The Σ^1_1 lower bound is a reduction from the recurrent tiling problem, a standard Σ^1_1-complete problem asking whether $\mathbb{N} \times \mathbb{N}$ can be tiled by a given finite set of tiles. So, let us consider the upper bound: Σ^1_1 allows to quantify over type 1 objects: functions from natural numbers to natural numbers, or, equivalently, over sets of natural numbers, i.e. countable objects. On the other hand, HyperLTL formulas are evaluated over sets of infinite traces, i.e. uncountable objects. Thus, to show that quantification over type 1 objects is sufficient, we need to apply a result of Finkbeiner and Zimmermann proving that every satisfiable HyperLTL formula has a countable model [26]. Then, we can prove Σ^1_1-membership by expressing the existence of a model and the existence of appropriate Skolem functions for the trace quantifiers by type 1 quantification. We also prove that the satisfiability problem remains Σ^1_1-complete when restricted to ultimately periodic traces, or, equivalently, when restricted to finite traces.

Then, we turn our attention to HyperCTL* satisfiability. Recall that HyperCTL* formulas are evaluated over (possibly infinite) transition systems, which can be much larger than type 2 objects whose cardinality is bounded by $\mathfrak{c}$, the cardinality of the continuum. Hence, to obtain our upper bound on the complexity we need, just like in the case of HyperLTL, an upper bound on the size of minimal models of satisfiable HyperCTL* formulas. To this end, we generalise the proof of Finkbeiner and Zimmermann to HyperCTL*, showing that every satisfiable HyperCTL* formula has a model of size $\mathfrak{c}$. We also exhibit a satisfiable HyperCTL* formula $\varphi_{\mathfrak{c}}$ whose models all have at least cardinality $\mathfrak{c}$, as they have to encode all subsets of $\mathbb{N}$ by disjoint paths. Thus, our upper bound $\mathfrak{c}$ is tight.

With this upper bound on the cardinality of models, we are able to prove Σ^2_1-membership of HyperCTL* satisfiability by expressing with type 2 quantification the existence of a model and the existence of a winning strategy in the induced model checking game. The matching lower bound is proven by directly encoding the arithmetic formulas inducing Σ^2_1 as instances of the HyperCTL* satisfiability problem. To this end, we use the formula $\varphi_{\mathfrak{c}}$ whose models have for each subset $A \subseteq \mathbb{N}$ a path encoding A. Now, quantification over type 0 objects (natural numbers) is simulated by quantification of a path encoding a singleton set, quantification over type 1 objects (which can be assumed to be sets of natural numbers) is simulated by quantification over the paths encoding such subsets, and existential quantification over type 2 objects (which can be assumed to be subsets of $2^{\mathbb{N}}$) is simulated by the choice of the model, i.e. a model encodes k subsets of $2^{\mathbb{N}}$ if there are k existential type 2 quantifiers. Finally, the arithmetic operations can easily be implemented in HyperLTL, and therefore also in HyperCTL*.

After settling the complexity of satisfiability, we turn our attention to the HyperLTL quantifier alternation hierarchy and its relation to satisfiability. Rabe remarks that the hierarchy is strict [35], and Mascle and Zimmermann show that every HyperLTL formula has

a polynomial-time computable equi-satisfiable formula with one quantifier alternation [33]. Here, we present a novel proof of strictness by embedding the FO[<] alternation hierarchy, which is also strict [15, 37]. We use our construction to prove that for every $n > 0$, deciding whether a given formula is equivalent to a formula with at most n quantifier alternations is Π^1_1-complete (i.e. the co-class of Σ^1_1).

All proofs omitted due to space restrictions can be found in the full version [27].

2 Preliminaries

Fix a finite set AP of atomic propositions. A *trace* over AP is a map $t\colon \mathbb{N} \to 2^{\mathrm{AP}}$, denoted by $t(0)t(1)t(2)\cdots$. It is *ultimately periodic*, if $t = x \cdot y^\omega$ for some $x, y \in (2^{\mathrm{AP}})^+$, i.e. there are $s, p > 0$ with $t(n) = t(n+p)$ for all $n \geq s$. The set of all traces over AP is $(2^{\mathrm{AP}})^\omega$.

A transition system $\mathcal{T} = (V, E, v_I, \lambda)$ consists of a set V of vertices, a set $E \subseteq V \times V$ of (directed) edges, an initial vertex $v_I \in V$, and a labelling $\lambda\colon V \to 2^{\mathrm{AP}}$ of the vertices by sets of atomic propositions. A path ρ through $\mathcal{T}$ is an infinite sequence $\rho(0)\rho(1)\rho(2)\cdots$ of vertices with $(\rho(n), \rho(n+1)) \in E$ for every $n \geq 0$.

HyperLTL. The formulas of HyperLTL are given by the grammar

$$\varphi ::= \exists \pi.\ \varphi \mid \forall \pi.\ \varphi \mid \psi \qquad\qquad \psi ::= a_\pi \mid \neg\psi \mid \psi \vee \psi \mid \mathbf{X}\,\psi \mid \psi\,\mathbf{U}\,\psi$$

where a ranges over atomic propositions in AP and where π ranges over a fixed countable set $\mathcal{V}$ of *(trace) variables*. Conjunction, implication, and equivalence are defined as usual, and the temporal operators eventually $\mathbf{F}$ and always $\mathbf{G}$ are derived as $\mathbf{F}\,\psi = \neg\psi\,\mathbf{U}\,\psi$ and $\mathbf{G}\,\psi = \neg\,\mathbf{F}\,\neg\psi$. A *sentence* is a formula without free variables.

The semantics of HyperLTL is defined with respect to a *trace assignment*, a partial mapping $\Pi\colon \mathcal{V} \to (2^{\mathrm{AP}})^\omega$. The assignment with empty domain is denoted by $\Pi_\emptyset$. Given a trace assignment Π, a variable π, and a trace t we denote by $\Pi[\pi \to t]$ the assignment that coincides with Π everywhere but at π, which is mapped to t. Furthermore, $\Pi[j, \infty)$ denotes the trace assignment mapping every π in Π's domain to $\Pi(\pi)(j)\Pi(\pi)(j+1)\Pi(\pi)(j+2)\cdots$, its suffix from position j onwards.

For sets T of traces and trace assignments Π we define

- $(T, \Pi) \models a_\pi$ if $a \in \Pi(\pi)(0)$,
- $(T, \Pi) \models \neg\psi$ if $(T, \Pi) \not\models \psi$,
- $(T, \Pi) \models \psi_1 \vee \psi_2$ if $(T, \Pi) \models \psi_1$ or $(T, \Pi) \models \psi_2$,
- $(T, \Pi) \models \mathbf{X}\,\psi$ if $(T, \Pi[1, \infty)) \models \psi$,
- $(T, \Pi) \models \psi_1\,\mathbf{U}\,\psi_2$ if there is a $j \geq 0$ such that $(T, \Pi[j, \infty)) \models \psi_2$ and for all $0 \leq j' < j$: $(T, \Pi[j', \infty)) \models \psi_1$,
- $(T, \Pi) \models \exists \pi.\ \varphi$ if there exists a trace $t \in T$ such that $(T, \Pi[\pi \to t]) \models \varphi$, and
- $(T, \Pi) \models \forall \pi.\ \varphi$ if for all traces $t \in T$: $(T, \Pi[\pi \to t]) \models \varphi$.

We say that T *satisfies* a sentence φ if $(T, \Pi_\emptyset) \models \varphi$. In this case, we write $T \models \varphi$ and say that T is a *model* of φ. Although HyperLTL sentences are required to be in prenex normal form, they are closed under Boolean combinations, which can easily be seen by transforming such formulas into prenex normal form. Two HyperLTL sentences φ and φ' are equivalent if $T \models \varphi$ if and only if $T \models \varphi'$ for every set T of traces.

HyperCTL*. The formulas of HyperCTL* are given by the grammar

$$\varphi ::= a_\pi \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \exists\pi.\ \varphi \mid \forall\pi.\ \varphi$$

where a ranges over atomic propositions in AP and where π ranges over a fixed countable set $\mathcal{V}$ of *(path) variables*, and where we require that each temporal operator appears in the scope of a path quantifier. Again, other Boolean connectives and temporal operators are derived as usual. Sentences are formulas without free variables.

Let $\mathcal{T}$ be a transition system. The semantics of HyperCTL* is defined with respect to a *path assignment*, a partial mapping Π from variables in $\mathcal{V}$ to paths of $\mathcal{T}$. The assignment with empty domain is denoted by $\Pi_\emptyset$. Given a path assignment Π, a variable π, and a path ρ we denote by $\Pi[\pi \to \rho]$ the assignment that coincides with Π everywhere but at π, which is mapped to ρ. Furthermore, $\Pi[j, \infty)$ denotes the path assignment mapping every π in Π's domain to $\Pi(\pi)(j)\Pi(\pi)(j+1)\Pi(\pi)(j+2)\cdots$, its suffix from position j onwards.

For transition systems $\mathcal{T}$ and path assignments Π we define

- $(\mathcal{T}, \Pi) \models a_\pi$ if $a \in \lambda(\Pi(\pi)(0))$, where λ is the labelling function of $\mathcal{T}$,
- $(\mathcal{T}, \Pi) \models \neg\psi$ if $(\mathcal{T}, \Pi) \not\models \psi$,
- $(\mathcal{T}, \Pi) \models \psi_1 \vee \psi_2$ if $(\mathcal{T}, \Pi) \models \psi_1$ or $(\mathcal{T}, \Pi) \models \psi_2$,
- $(\mathcal{T}, \Pi) \models \mathbf{X}\,\psi$ if $(\mathcal{T}, \Pi[1, \infty)) \models \psi$,
- $(\mathcal{T}, \Pi) \models \psi_1\,\mathbf{U}\,\psi_2$ if there exists a $j \geq 0$ such that $(\mathcal{T}, \Pi[j, \infty)) \models \psi_2$ and for all $0 \leq j' < j$: $(\mathcal{T}, \Pi[j', \infty)) \models \psi_1$,
- $(\mathcal{T}, \Pi) \models \exists\pi.\ \varphi$ if there exists a path ρ of $\mathcal{T}$, starting in $\mathrm{rcnt}(\Pi)$, such that $(\mathcal{T}, \Pi[\pi \to \rho]) \models \varphi$, and
- $(\mathcal{T}, \Pi) \models \forall\pi.\ \varphi$ if for all paths ρ of $\mathcal{T}$ starting in $\mathrm{rcnt}(\Pi)$: $(\mathcal{T}, \Pi[\pi \to \rho]) \models \varphi$.

Here, $\mathrm{rcnt}(\Pi)$ is the initial vertex of $\Pi(\pi)$, where π is the path variable most recently added to Π, and the initial vertex of $\mathcal{T}$ if Π is empty.[1] We say that $\mathcal{T}$ *satisfies* a sentence φ if $(\mathcal{T}, \Pi_\emptyset) \models \varphi$. In this case, we write $\mathcal{T} \models \varphi$ and say that $\mathcal{T}$ is a *model* of φ.

Complexity Classes for Undecidable Problems. A type 0 object is a natural number $n \in \mathbb{N}$, a type 1 object is a function $f\colon \mathbb{N} \to \mathbb{N}$, and a type 2 object is a function $f\colon (\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$. As usual, predicate logic with quantification over type 0 objects (first-order quantifiers) is called first-order logic. Second- and third-order logic are defined similarly.

We consider formulas of arithmetic, i.e. predicate logic with signature $(0, 1, +, \cdot, <)$ evaluated over the natural numbers. With a single free variable of type 0, such formulas define sets of natural numbers (see, e.g. Rogers [36] for more details):

- Σ^0_1 contains the sets of the form $\{x \in \mathbb{N} \mid \exists x_0 \cdots \exists x_k.\ \psi(x, x_0, \ldots, x_k)\}$ where ψ is a quantifier-free arithmetic formula and the x_i are variables of type 0.
- Σ^1_1 contains the sets of the form $\{x \in \mathbb{N} \mid \exists x_0 \cdots \exists x_k.\ \psi(x, x_0, \ldots, x_k)\}$ where ψ is an arithmetic formula with arbitrary (existential and universal) quantification over type 0 objects and the x_i are variables of type 1.
- Σ^2_1 contains the sets of the form $\{x \in \mathbb{N} \mid \exists x_0 \cdots \exists x_k.\ \psi(x, x_0, \ldots, x_k)\}$ where ψ is an arithmetic formula with arbitrary (existential and universal) quantification over type 0 and type 1 objects and the x_i are variables of type 2.

Note that there is a bijection between functions of the form $f\colon \mathbb{N} \to \mathbb{N}$ and subsets of $\mathbb{N}$, which is implementable in arithmetic. Similarly, there is a bijection between functions of the

[1] For the sake of simplicity, we refrain from formalising this notion properly, which would require to keep track of the order in which variables are added to Π's domain.

form $f\colon (\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$ and subsets of $2^{\mathbb{N}}$, which is again implementable in arithmetic. Thus, whenever convenient, we use quantification over sets of natural numbers and over sets of sets of natural numbers, instead of quantification over type 1 and type 2 objects; in particular when proving lower bounds. We then include $\in$ in the signature.

3 HyperLTL satisfiability is Σ^1_1-complete

In this section we settle the complexity of the satisfiability problem for HyperLTL: given a HyperLTL sentence, determine whether it has a model.

▶ **Theorem 1.** *HyperLTL satisfiability is Σ^1_1-complete.*

We should contrast this result with [20, Theorem 1], which shows that HyperLTL satisfiability by *finite* sets of ultimately periodic traces is Σ^0_1-complete. The Σ^1_1-completeness of HyperLTL satisfiability in the general case implies that, in particular, the set of satisfiable HyperLTL sentences is neither recursively enumerable nor co-recursively enumerable. A semi-decision procedure, like the one introduced in [20] for finite sets of ultimately periodic traces, therefore cannot exist in general.

The Σ^1_1 upper bound relies on the fact that every satisfiable HyperLTL formula has a countable model [26]. This allows us to represent these models, and Skolem functions on them, by sets of natural numbers, which are type 1 objects. In this encoding, trace assignments are type 0 objects, as traces in a countable set can be identified by natural numbers. With some more existential type 1 quantification one can then express the existence of a function witnessing that every trace assignment consistent with the Skolem functions satisfies the quantifier-free part of the formula under consideration.

▶ **Lemma 2.** *HyperLTL satisfiability is in Σ^1_1.*

Proof. Let φ be a HyperLTL formula, let Φ denote the set of quantifier-free subformulas of φ, and let Π be a trace assignment whose domain contains the variables of φ. The expansion of φ on Π is the function $e_{\varphi,\Pi}\colon \Phi \times \mathbb{N} \to \{0,1\}$ with

$$e_{\varphi,\Pi}(\psi, j) = \begin{cases} 1 & \text{if } \Pi[j,\infty) \models \psi, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

The expansion is completely characterised by the following consistency conditions:

- $e_{\varphi,\Pi}(a_\pi, j) = 1$ if and only if $a \in \Pi(\pi)(j)$.
- $e_{\varphi,\Pi}(\neg\psi, j) = 1$ if and only if $e_{\varphi,\Pi}(\psi, j) = 0$.
- $e_{\varphi,\Pi}(\psi_1 \vee \psi_2, j) = 1$ if and only if $e_{\varphi,\Pi}(\psi_1, j) = 1$ or $e_{\varphi,\Pi}(\psi_2, j) = 1$.
- $e_{\varphi,\Pi}(\mathbf{X}\,\psi, j) = 1$ if and only if $e_{\varphi,\Pi}(\psi, j+1) = 1$.
- $e_{\varphi,\Pi}(\psi_1 \,\mathbf{U}\, \psi_2, j) = 1$ if and only if there is a $j' \geq j$ such that $e_{\varphi,\Pi}(\psi_2, j') = 1$ and $e_{\varphi,\Pi}(\psi_2, j'') = 1$ for all j'' in the range $j \leq j'' < j'$.

Every satisfiable HyperLTL sentence has a countable model [26]. Hence, to prove that the HyperLTL satisfiability problem is in Σ^1_1, we express, for a given HyperLTL sentence encoded as a natural number, the existence of the following type 1 objects (relying on the fact that there is a bijection between finite sequences over $\mathbb{N}$ and $\mathbb{N}$ itself):

- A countable set of traces over the propositions of φ encoded as a function T from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$, mapping trace names and positions to (encodings of) subsets of the set of propositions appearing in φ.

- A function S from $\mathbb{N} \times \mathbb{N}^*$ to $\mathbb{N}$ to be interpreted as Skolem functions for the existentially quantified variables of φ, i.e. we map a variable (identified by a natural number) and a trace assignment of the variables preceding it (encoded as a sequence of natural numbers) to a trace name.
- A function E from $\mathbb{N} \times \mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$, where, for a fixed $a \in \mathbb{N}$ encoding a trace assignment Π, the function $x, y \mapsto E(a, x, y)$ is interpreted as the expansion of φ on Π, i.e. x encodes a subformula in Φ and y is a position.

Then, we express the following properties using only type 0 quantification: For every trace assignment of the variables in φ, encoded by $a \in \mathbb{N}$, if a is consistent with the Skolem function encoded by S, then the function $x, y \mapsto E(a, x, y)$ satisfies the consistency conditions characterising the expansion, and we have $E(a, x_0, 0) = 1$, where x_0 is the encoding of the maximal quantifier-free subformula of φ. We leave the tedious, but standard, details to the industrious reader. ◀

Now, we prove hardness.

▶ **Lemma 3.** *HyperLTL satisfiability is Σ^1_1-hard.*

Proof. By a reduction from the *recurring tiling problem* which is given as follows. A *tile* is a function $\tau\colon \{east, west, north, south\} \to \mathcal{C}$ that maps directions into a finite set $\mathcal{C}$ of colours. Given a finite set $\mathcal{T}$ of tiles, a *tiling of the positive quadrant* with $\mathcal{T}$ is a function $T : \mathbb{N} \times \mathbb{N} \to \mathcal{T}$ with the property that:

- if $T(i, j) = \tau_1$ and $T(i + 1, j) = \tau_2$, then $\tau_1(east) = \tau_2(west)$ and
- if $T(i, j) = \tau_1$ and $T(i, j + 1) = \tau_2$ then $\tau_1(north) = \tau_2(south)$.

The *recurring tiling problem* is to determine, given a finite set $\mathcal{T}$ of tiles and a designated $\tau_0 \in \mathcal{T}$, whether there is a tiling T of the positive quadrant with $\mathcal{T}$ such that there are infinitely many $j \in \mathbb{N}$ such that $T(0, j) = \tau_0$. This problem is known to be Σ^1_1-complete [29], so if we reduce it to HyperLTL satisfiability this will establish the desired hardness result.

In our reduction, each x-coordinate in the positive quadrant will be represented by a trace, and each y-coordinate by a point in time.[2] In order to keep track of which trace represents which x-coordinate, we use one designated atomic proposition x that holds on exactly one time point in each trace: x holds at time i if and only if the trace represents x-coordinate i.

For this purpose, let $\mathcal{T}$ be given, and define the following formulas over $\mathrm{AP} = \{x\} \cup \mathcal{T}$:

- Every trace has exactly one point where x holds:

$$\varphi_1 = \forall \pi.\ (\neg x_\pi \,\mathbf{U}(x_\pi \wedge \mathbf{X}\,\mathbf{G}\,\neg x_\pi))$$

- For every $i \in \mathbb{N}$, there is a trace with x in the i-th position:

$$\varphi_2 = (\exists \pi.\ x_\pi) \wedge (\forall \pi_1.\ \exists \pi_2.\ \ \mathbf{F}(x_{\pi_1} \wedge \mathbf{X}\, x_{\pi_2}))$$

- If two traces represent the same x-coordinate, then they contain the same tiles:

$$\varphi_3 = \forall \pi_1.\ \forall \pi_2.\ (\mathbf{F}(x_{\pi_1} \wedge x_{\pi_2}) \to \mathbf{G}(\bigwedge_{\tau \in \mathcal{T}} (\tau_{\pi_1} \leftrightarrow \tau_{\pi_2})))$$

[2] Note that this means that if we were to visually represent this construction, traces would be arranged vertically.

- Every time point in every trace contains exactly one tile:

$$\varphi_4 = \forall \pi.\ \mathbf{G} \bigvee_{\tau \in \mathcal{T}} (\tau_\pi \wedge \bigwedge_{\tau' \in T \setminus \{\tau\}} \neg(\tau')_\pi)$$

- Tiles match vertically:

$$\varphi_5 = \forall \pi.\ \mathbf{G} \bigvee_{\tau \in \mathcal{T}} (\tau_\pi \wedge \bigvee\nolimits_{\tau' \in \{\tau' \in \mathcal{T} \mid \tau(north) = \tau'(south)\}} \mathbf{X}(\tau')_\pi)$$

- Tiles match horizontally:

$$\varphi_6 = \forall \pi_1.\ \forall \pi_2.\ (\mathbf{F}(x_{\pi_1} \wedge \mathbf{X}\, x_{\pi_2}) \rightarrow \mathbf{G} \bigvee_{\tau \in \mathcal{T}} (\tau_{\pi_1} \wedge \bigvee\nolimits_{\tau' \in \{\tau' \in \mathcal{T} \mid \tau(east) = \tau'(west)\}} (\tau')_{\pi_2}))$$

- Tile τ_0 occurs infinitely often at x-position 0:

$$\varphi_7 = \exists \pi.\ (x_\pi \wedge \mathbf{G}\, \mathbf{F}\, \tau_0)$$

Finally, take $\varphi_{\mathcal{T}} = \bigwedge_{1 \leq i \leq 7} \varphi_i$. Technically $\varphi_{\mathcal{T}}$ is not a HyperLTL formula, since it is not in prenex normal form, but it can be trivially transformed into one. Collectively, subformulas φ_1–φ_3 are satisfied in exactly those sets of traces that can be interpreted as $\mathbb{N} \times \mathbb{N}$. Subformulas φ_4–φ_6 then hold if and only if the $\mathbb{N} \times \mathbb{N}$ grid is correctly tiled with $\mathcal{T}$. Subformula φ_7, finally, holds if and only if the tiling uses the tile τ_0 infinitely often at x-coordinate 0. Overall, this means $\varphi_{\mathcal{T}}$ is satisfiable if and only if $\mathcal{T}$ can recurrently tile the positive quadrant.

The Σ^1_1-hardness of HyperLTL satisfiability therefore follows from the Σ^1_1-hardness of the recurring tiling problem [29]. ◀

The Σ^1_1-completeness of HyperLTL satisfiability still holds if we restrict to ultimately periodic traces.

▶ **Theorem 4.** *HyperLTL satisfiability restricted to sets of ultimately periodic traces is Σ^1_1-complete.*

Proof. The problem of whether there is a tiling of $\{(i, j) \in \mathbb{N}^2 \mid i \geq j\}$, i.e. the part of $\mathbb{N} \times \mathbb{N}$ below the diagonal, such that a designated tile τ_0 occurs on every row, is also Σ^1_1-complete [29].[3] We reduce this problem to HyperLTL satisfiability on ultimately periodic traces.

The reduction is very similar to the one discussed above, with the necessary changes being: (i) every time point beyond x satisfies the special tile "null", (ii) horizontal and vertical matching are only checked at or before time point x and (iii) for every π_1 there is a π_2 such that π_2 has designated tile τ_0 at the time where π_1 satisfies x (so τ_0 holds at least once in every row).

Membership in Σ^1_1 can be shown similarly to the proof of Lemma 2. So, the problem is Σ^1_1-complete. ◀

[3] The proof in [29] is for the part *above* the diagonal with τ_0 occurring on every column, but that is easily seen to be equivalent.

4 The HyperLTL Quantifier Alternation Hierarchy

The number of quantifier alternations in a formula is a crucial parameter in the complexity of HyperLTL model-checking [25, 35]. A natural question is then to understand *which* properties can be expressed with n quantifier alternations, that is, given a sentence φ, determine if there exists an equivalent one with at most n alternations. In this section, we show that this problem is in fact exactly as hard as the HyperLTL unsatisfiability problem (which asks whether a HyperLTL sentence has no model), and therefore Π^1_1-complete. Here, Π^1_1 is the co-class of Σ^1_1, i.e. it contains the complements of the Σ^1_1 sets.

Formally, the HyperLTL quantifier alternation hierarchy is defined as follows. Let φ be a HyperLTL formula. We say that φ is a Σ_0- or a Π_0-formula if it is quantifier-free. It is a Σ_n-formula if it is of the form $\varphi = \exists \pi_1 \cdots \exists \pi_k.\ \psi$ and ψ is a Π_{n-1}-formula. It is a Π_n-formula if it is of the form $\varphi = \forall \pi_1 \cdots \forall \pi_k.\ \psi$ and ψ is a Σ_{n-1}-formula. We do not require each block of quantifiers to be non-empty, i.e. we may have $k = 0$ and $\varphi = \psi$. By a slight abuse of notation, we also let Σ_n denote the set of hyperproperties definable by a Σ_n-sentence, that is, the set of all $L(\varphi) = \{T \subseteq (2^{\mathrm{AP}})^\omega \mid T \models \varphi\}$ such that φ is a Σ_n-sentence of HyperLTL.

▶ **Theorem 5** ([35, Corollary 5.6.5]). *The quantifier alternation hierarchy of HyperLTL is strict: for all $n > 0$, $\Sigma_n \subsetneq \Sigma_{n+1}$.*

The strictness of the hierarchy also holds if we restrict our attention to sentences whose models consist of finite sets of traces that end in the suffix $\emptyset^\omega$, i.e. that are essentially finite.

▶ **Theorem 6.** *For all $n > 0$, there exists a Σ_{n+1}-sentence φ of HyperLTL that is not equivalent to any Σ_n-sentence, and such that for all $T \subseteq (2^{\mathrm{AP}})^\omega$, if $T \models \varphi$ then T contains finitely many traces and $T \subseteq (2^{\mathrm{AP}})^* \emptyset^\omega$.*

This fact is a necessary ingredient for our argument that membership at some fixed level of the quantifier alternation hierarchy is Π^1_1-hard. It could be derived from a small adaptation of the proof in [35], and we provide an alternative proof in the extended version [27] by exhibiting a connection between the HyperLTL quantifier alternation hierarchy and the quantifier alternation hierarchy for first-order logic over finite words, which is known to be strict [15, 38].

Our goal is to prove the following.

▶ **Theorem 7.** *Fix $n > 0$. The problem of deciding whether a HyperLTL sentence is equivalent to some Σ_n-sentence is Π^1_1-complete.*

The easier part is the upper bound, since a corollary of Theorem 1 is that the problem of deciding whether two HyperLTL formulas are equivalent is Π^1_1-complete. The lower bound is proven by reduction from the HyperLTL unsatisfiability problem. The proof relies on Theorem 6: given a sentence φ, we are going to combine φ with some Σ_{n+1}-sentence φ_{n+1} witnessing the strictness of the hierarchy, to construct a sentence ψ such that φ is unsatisfiable if and only if ψ is equivalent to a Σ_n-sentence. Intuitively, the formula ψ will describe models consisting of the "disjoint union" of a model of φ_{n+1} and a model of φ. Here "disjoint" is to be understood in a strong sense: we split both the set of traces and the time domain into two parts, used respectively to encode the models of φ_{n+1} and those of φ.

To make this more precise, let us introduce some notations. We assume a distinguished symbol $\$ \notin \mathrm{AP}$. We say that a set of traces $T \subseteq (2^{\mathrm{AP} \cup \{\$\}})^\omega$ is *bounded* if there exists $b \in \mathbb{N}$ such that $T \subseteq (2^{\mathrm{AP}})^b \cdot \{\$\}^\omega$.

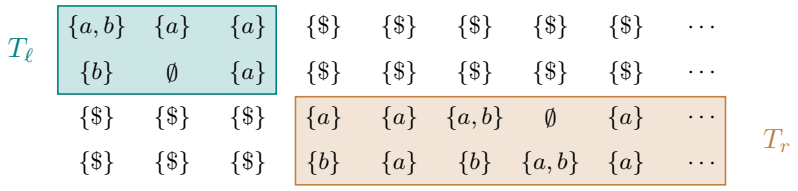

Figure 1 Example of a split set of traces where each row represents a trace and $b = 3$.

▶ **Lemma 8.** *There exists a Π_1-sentence φ_{bd} such that for all $T \subseteq (2^{\mathrm{AP}\cup\{\$\}})^\omega$, we have $T \models \varphi_{bd}$ if and only if T is bounded.*

Proof. We let

$$\varphi_{bd} = \forall\pi.\, \forall\pi'.\, (\neg\$_\pi \,\mathbf{U}\, \mathbf{G}\, \$_\pi) \wedge \bigwedge_{a\in\mathrm{AP}} \mathbf{G}(\neg(a_\pi \wedge \$_\pi)) \wedge \mathbf{F}\,(\neg\$_\pi \wedge \neg\$_{\pi'} \wedge \mathbf{X}\,\$_\pi \wedge \mathbf{X}\,\$_{\pi'})\ .$$

The conjunct $(\neg\$_\pi \,\mathbf{U}\, \mathbf{G}\, \$_\pi) \wedge \bigwedge_{a\in\mathrm{AP}} \mathbf{G}(\neg(a_\pi \wedge \$_\pi))$ ensures that every trace is in $(2^{\mathrm{AP}})^* \cdot \{\$\}^\omega$, while $\mathbf{F}\,(\neg\$_\pi \wedge \neg\$_{\pi'} \wedge \mathbf{X}\,\$_\pi \wedge \mathbf{X}\,\$_{\pi'})$ ensures that the \$'s in any two traces π and π' start at the same position. ◀

We say that T is *split* if there exist $b \in \mathbb{N}$ and T_1, T_2 such that $T = T_1 \uplus T_2$, $T_1 \subseteq (2^{\mathrm{AP}})^b \cdot \{\$\}^\omega$, and $T_2 \subseteq \{\$\}^b \cdot (2^{\mathrm{AP}})^\omega$. Note that b is unique here. Hence, we define the left and right part of T as $T_\ell = T_1$ and $T_r = \{t \in (2^{\mathrm{AP}})^\omega \mid \{\$\}^b \cdot t \in T_2\}$, respectively (see Figure 1).

It is easy to combine HyperLTL specifications for the left and right part of a split model into one global formula.

▶ **Lemma 9.** *For all HyperLTL sentences φ_ℓ, φ_r, one can construct a sentence ψ such that for all split $T \subseteq (2^{\mathrm{AP}\cup\{\$\}})^\omega$, it holds that $T_\ell \models \varphi_\ell$ and $T_r \models \varphi_r$ if and only if $T \models \psi$.*

Proof of Lemma 9. Let $\widehat{\varphi_r}$ denote the formula obtained from φ_r by replacing:

- every existential quantification $\exists\pi.\ \varphi$ with $\exists\pi.\ ((\mathbf{F}\,\mathbf{G}\,\neg\$_\pi) \wedge \varphi)$;
- every universal quantification $\forall\pi.\ \varphi$ with $\forall\pi.\ ((\mathbf{F}\,\mathbf{G}\,\neg\$_\pi) \rightarrow \varphi)$;
- the quantifier-free part φ of φ_r with $\$_\pi\,\mathbf{U}(\neg\$_\pi \wedge \varphi)$, where π is some free variable in φ.

Here, the first two replacements restrict quantification to traces in the right part while the last one requires the formula to hold at the first position of the right part. We define $\widehat{\varphi_\ell}$ by similarly relativizing quantifications in φ_ℓ. The formula $\widehat{\varphi_\ell} \wedge \widehat{\varphi_r}$ can then be put back into prenex normal form to define ψ. ◀

Conversely, any HyperLTL formula that only has split models can be decomposed into a Boolean combination of formulas that only talk about the left or right part of the model. This is formalised in the lemma below.

▶ **Lemma 10.** *For all HyperLTL Σ_n-sentences φ there exists a finite family $(\varphi^i_\ell, \varphi^i_r)_i$ of Σ_n-sentences such that for all split $T \subseteq (2^{\mathrm{AP}\cup\{\$\}})^\omega$: $T \models \varphi$ if and only if there is an i with $T_\ell \models \varphi^i_\ell$ and $T_r \models \varphi^i_r$.*

We are now ready to prove Theorem 7.

Proof of Theorem 7. The upper bound is an easy consequence of Theorem 1: Given a HyperLTL sentence φ, we express the existence of a Σ_n-sentence ψ using first-order quantification and encode equivalence of ψ and φ via the formula $(\neg\varphi \wedge \psi) \vee (\varphi \wedge \neg\psi)$, which is unsatisfiable if and only if φ and ψ are equivalent. Altogether, this shows membership in Π^1_1, as Π^1_1 is closed under existential first-order quantification (see, e.g. [30, Page 82]).

We prove the lower bound by reduction from the unsatisfiability problem for HyperLTL. So given a HyperLTL sentence φ, we want to construct ψ such that φ is unsatisfiable if and only if ψ is equivalent to a Σ_n-sentence.

We first consider the case $n > 1$. Fix a Σ_{n+1}-sentence φ_{n+1} that is in not equivalent to any Σ_n-sentence, and such that every model of φ_{n+1} is bounded. The existence of such a formula is a consequence of Theorem 6. By Lemma 9, there exists a computable ψ such that for all split models T, we have $T \models \psi$ if and only if $T_\ell \models \varphi_{n+1}$ and $T_r \models \varphi$.

First, it is clear that if φ is unsatisfiable, then ψ is unsatisfiable as well, and thus equivalent to $\exists \pi.\ a_\pi \wedge \neg a_\pi$, which is a Σ_n-sentence since $n \geq 1$.

Conversely, suppose towards a contradiction that φ is satisfiable and that ψ is equivalent to some Σ_n-sentence. Let $(\psi_\ell^i, \psi_r^i)_i$ be the finite family of Σ_n-sentences given by Lemma 10 for ψ. Fix a model T_φ of φ. For a bounded T, we let $\overline{T}$ denote the unique split set of traces such that $\overline{T_\ell} = T$ and $\overline{T_r} = T_\varphi$. For all T, we then have $T \models \varphi_{n+1}$ if and only if T is bounded and $\overline{T} \models \psi$. Recall that the set of bounded models can be defined by a Π_1-sentence φ_{bd} (Lemma 8), which is also a Σ_n-sentence since $n > 1$. We then have $T \models \varphi_{n+1}$ if and only if $T \models \varphi_{bd}$ and there exists i such that $T \models \psi_\ell^i$ and $T_\varphi \models \psi_r^i$. So φ_{n+1} is equivalent to

$$\varphi_{bd} \wedge \bigvee_{i \text{ with } T_\varphi \models \psi_r^i} \psi_\ell^i \,,$$

which, since Σ_n-sentences are closed (up to logical equivalence) under conjunction and disjunction, is equivalent to a Σ_n-sentence. This contradicts the definition of φ_{n+1}.

We are left with the case $n = 1$. Similarly, we construct ψ such that φ is unsatisfiable if and only if ψ is unsatisfiable, and if and only if ψ is equivalent to a Σ_1-sentence. However, we do not need to use bounded or split models here. Every satisfiable Σ_1-sentence has a model with finitely many traces. Therefore, a simple way to construct ψ so that it is not equivalent to any Σ_1-sentence (unless it is unsatisfiable) is to ensure that every model of ψ contains infinitely many traces.

Let $x \notin \mathrm{AP}$, and $T_\omega = \{\emptyset^n\{x\}\emptyset^\omega \mid n \in \mathbb{N}\}$. As seen in the proof of Lemma 3, T_ω is definable in HyperLTL: There is a sentence φ_ω such that $T \subseteq (2^{\mathrm{AP} \cup \{x\}})^\omega$ is a model of φ_ω if and only if $T = T_\omega$. By relativising quantifiers in φ_ω and φ to traces with or without the atomic proposition x, one can construct a HyperLTL sentence ψ such that $T \models \psi$ if and only if $T_\omega \subseteq T$ and $T \setminus T_\omega \models \varphi$.

Again, if φ is unsatisfiable then ψ is unsatisfiable and therefore equivalent to $\exists \pi.\ a_\pi \wedge \neg a_\pi$, a Σ_1-sentence. Conversely, all models of ψ contain infinitely many traces and therefore, if ψ is equivalent to a Σ_1-sentence then it is unsatisfiable, and so is φ. ◀

5 HyperCTL* satisfiability is Σ_1^2-complete

Here, we consider the HyperCTL* satisfiability problem: given a HyperLTL sentence, determine whether it has a model $\mathcal{T}$ (of arbitrary size). We prove that it is much harder than HyperLTL satisfiability. As a key step of the proof, we also prove that every satisfiable sentence admits a model of cardinality at most $\mathfrak{c}$ (the cardinality of the continuum), and conversely, we exhibit a satisfiable sentence whose models are all of cardinality at least $\mathfrak{c}$.

▶ **Theorem 11.** *HyperCTL* satisfiability is Σ_1^2-complete.*

On the other hand, HyperCTL* satisfiability restricted to finite transition systems is Σ_1^0-complete. The upper bound follows from HyperCTL* model checking being decidable [12] (therefore, the problem is recursively enumerable and thus in Σ_1^0) while the matching lower bound is inherited from HyperLTL [19].

Upper bound. We begin by proving membership in Σ^2_1. The first step is to obtain a bound on the size of minimal models of satisfiable HyperCTL* sentences. For this, we use an argument based on Skolem functions, which is a transfinite generalisation of the proof that all satisfiable HyperLTL sentences have a countable model [26].

In the following, we use ω and ω_1 to denote the first infinite and the first uncountable ordinal, respectively, and write $\aleph_0$ and $\aleph_1$ for their cardinality.

▶ **Lemma 12.** *Each satisfiable HyperCTL* sentence φ has a model of size at most $\mathfrak{c}$.*

Proof sketch. Suppose φ has a model $\mathcal{T}$ of arbitrary size, and fix Skolem functions witnessing this satisfaction. We then create a transfinite sequence of transition systems $\mathcal{T}_\alpha$. We start by taking $\mathcal{T}_0$ to be any single path from $\mathcal{T}$ starting in the initial vertex, and obtain $\mathcal{T}_{\alpha+1}$ by adding to $\mathcal{T}_\alpha$ all vertices and edges of the paths that are the outputs of the Skolem functions when restricted to inputs from $\mathcal{T}_\alpha$. If α is a limit ordinal we take $\mathcal{T}_\alpha$ to be the union of all previous transition systems.

This sequence does not necessarily stabilise at ω, since $\mathcal{T}_\omega$ may contain a path ρ such that $\rho(i)$ was introduced in $\mathcal{T}_i$. This would result in $\mathcal{T}_\omega$ containing a path that was not present in any earlier model $\mathcal{T}_i$ with $i < \omega$, and therefore we could have $\mathcal{T}_{\omega+1} \neq \mathcal{T}_\omega$.

The sequence does stabilise at ω_1, however. This is because every path ρ contains only countably many vertices, so if every element $\rho(i)$ of ρ is introduced at some countable α_i, then there is a countable α such that all of ρ is included in $\mathcal{T}_\alpha$. It follows that $\mathcal{T}_{\omega_1}$ does not contain any "new" paths that were not already in some $\mathcal{T}_\alpha$ with $\alpha < \omega_1$, and therefore the Skolem function f does not generate any "new" outputs either.

In each step of the construction at most $\mathfrak{c}$ new vertices are added, so $\mathcal{T}_{\omega_1}$ contains at most $\mathfrak{c}$ vertices. Furthermore, because $\mathcal{T}_{\omega_1}$ is closed under the Skolem functions, the satisfaction of φ in $\mathcal{T}$ implies its satisfaction in $\mathcal{T}_{\omega_1}$. ◀

With the upper bound at hand, we can place HyperCTL* satisfiability in Σ^2_1, as the existence of a model of size $\mathfrak{c}$ can be captured by quantification over type 2 objects.

▶ **Lemma 13.** *HyperCTL* satisfiability is in Σ^2_1.*

Proof. As in the proof of Theorem 1. Because every HyperCTL* formula is satisfied in a model of size at most $\mathfrak{c}$, these models can be represented by objects of type 2. Checking whether a formula is satisfied in a transition system is equivalent to the existence of a winning strategy for Verifier in the induced model checking game. Such a strategy is again a type 2 object, which is existentially quantified. Finally, whether it is winning can be expressed by quantification over individual elements and paths, which are objects of types 0 and 1. Checking the satisfiability of a HyperCTL* formula φ therefore amounts to existential third-order quantification (to choose a model and a winning strategy) followed by a second-order formula to verify that φ holds on the model. Hence HyperCTL* satisfiability is in Σ^2_1.

Formally, we encode the existence of a winning strategy for Verifier in the HyperCTL* model checking game $\mathcal{G}(\mathcal{T}, \varphi)$ induced by a transition system $\mathcal{T}$ and a HyperCTL* formula φ. This game is played between Verifier and Falsifier, one of them aiming to prove that $\mathcal{T} \models \varphi$ and the other aiming to prove $\mathcal{T} \not\models \varphi$. It is played in a graph whose positions correspond to subformulas which they want to check (and suitable path assignments of the free variables): each vertex (say, representing a subformula ψ) belongs to one of the players who has to pick a successor, which represents a subformula of ψ. A play ends at an atomic proposition, at which point the winner can be determined.

Formally, a vertex of the game is of the form (Π, ψ, b) where Π is a path assignment, ψ is a subformula of φ, and $b \in \{0, 1\}$ is a flag used to count the number of negations encountered along the play; the initial vertex is $(\Pi_\emptyset, \varphi, 0)$. Furthermore, for until-subformulas ψ, we need auxiliary vertices of the form (Π, ψ, b, j) with $j \in \mathbb{N}$. The vertices of Verifier are

- of the form $(\Pi, \psi, 0)$ with $\psi = \psi_1 \vee \psi_2$, $\psi = \psi_1 \,\mathbf{U}\, \psi_2$, or $\psi = \exists \pi.\ \psi'$,
- of the form $(\Pi, \forall \pi.\ \psi', 1)$, or
- of the form $(\Pi, \psi_1 \,\mathbf{U}\, \psi_2, 1, j)$.

The moves of the game are defined as follows:

- A vertex (Π, a_π, b) is terminal. It is winning for Verifier if $b = 0$ and $a \in \lambda(\Pi(\pi)(0))$ or if $b = 1$ and $a \notin \lambda(\Pi(\pi)(0))$, where λ is the labelling function of $\mathcal{T}$.
- A vertex $(\Pi, \neg\psi, b)$ has a unique successor $(\Pi, \psi, b + 1 \bmod 2)$.
- A vertex $(\Pi, \psi_1 \vee \psi_2, b)$ has two successors of the form (Π, ψ_i, b) for $i \in \{1, 2\}$.
- A vertex $(\Pi, \mathbf{X}\,\psi, b)$ has a unique successor $(\Pi[1, \infty), \psi, b)$.
- A vertex $(\Pi, \psi_1 \,\mathbf{U}\, \psi_2, b)$ has a successor $(\Pi, \psi_1 \,\mathbf{U}\, \psi_2, b, j)$ for every $j \in \mathbb{N}$.
- A vertex $(\Pi, \psi_1 \,\mathbf{U}\, \psi_2, b, j)$ has the successor $(\Pi[j, \infty), \psi_2, b)$ as well as successors $(\Pi[j', \infty), \psi_1, b)$ for every $0 \leq j' < j$.
- A vertex $(\Pi, \exists \pi.\ \psi, b)$ has successors $(\Pi[\pi \mapsto \rho], \psi, b)$ for every path ρ of $\mathcal{T}$ starting in $\mathrm{rcnt}(\Pi)$.
- A vertex $(\Pi, \forall \pi.\ \psi, b)$ has successors $(\Pi[\pi \mapsto \rho], \psi, b)$ for every path ρ of $\mathcal{T}$ starting in $\mathrm{rcnt}(\Pi)$.

A play of the model checking game is a finite path through the graph, starting at the initial vertex and ending at a terminal vertex. It is winning for Verifier if the terminal vertex is winning for her. Note that the length of a play is bounded by $2d$, where d is the depth[4] of φ, as the formula is simplified during each move.

A strategy σ for Verifier is a function mapping each of her vertices v to some successor of v. A play $v_0 \cdots v_k$ is consistent with σ, if $v_{k'+1} = \sigma(v_{k'})$ for every $0 \leq k' < k$ such that $v_{k'}$ is a vertex of Verifier. A straightforward induction shows that Verifier has a winning strategy for $\mathcal{G}(\mathcal{T}, \varphi)$ if and only if $\mathcal{T} \models \varphi$.

Recall that every satisfiable HyperCTL* sentence has a model of cardinality $\mathfrak{c}$ (Lemma 12). Thus, to place HyperCTL* satisfiability in Σ^2_1, we express, for a given natural number encoding a HyperCTL* formula φ, the existence of the following type 2 objects (using suitable encodings):

- A transition system $\mathcal{T}$ of cardinality $\mathfrak{c}$.
- A function σ from V to V, where V is the set of vertices of $\mathcal{G}(\mathcal{T}, \varphi)$. Note that a single vertex of V is a type 1 object.

Then, we express that σ is a strategy for Verifier, which is easily expressible using quantification over type 1 objects. Thus, it remains to express that σ is winning by stating that every play (a sequence of type 1 objects of bounded length) that is consistent with σ ends in a terminal vertex that is winning for Verifier. Again, we leave the tedious, but standard, details to the reader. ◀

Lower bound. We first describe a satisfiable HyperCTL* sentence $\varphi_\mathfrak{c}$ that does not have any model of cardinality less than $\mathfrak{c}$ (more precisely, the initial vertex must have uncountably many successors), thus matching the upper bound from Lemma 12. We construct $\varphi_\mathfrak{c}$ with one particular model $\mathcal{T}_\mathfrak{c}$ in mind, defined below, though it also admits other models.

[4] The depth is the maximal nesting of quantifiers, Boolean connectives, and temporal operators.

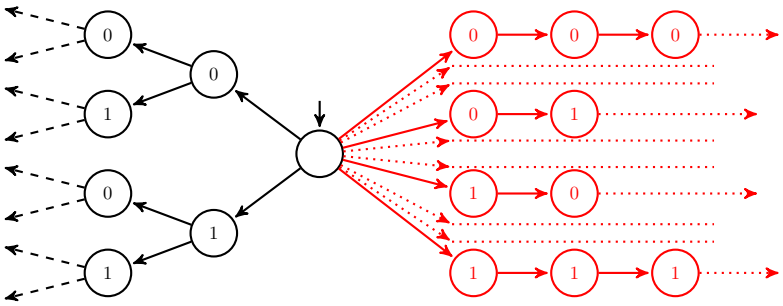

Figure 2 A depiction of $\mathcal{T}_{\mathfrak{c}}$. Vertices in black (on the left including the initial vertex) are labelled by `fbt`, those in red (on the right, excluding the initial vertex) are labelled by `set`.

The idea is that we want all possible subsets of $A \subseteq \mathbb{N}$ to be represented in $\mathcal{T}_{\mathfrak{c}}$ in the form of paths ρ_A such that $\rho_A(i)$ is labelled by 1 if $i \in A$, and by 0 otherwise. By ensuring that the first vertices of these paths are pairwise distinct, we obtain the desired lower bound on the cardinality. We express this in HyperCTL* as follows: First, we express that there is a part of the model (labelled by `fbt`) where every reachable vertex has two successors, one labelled with 0 and one labelled with 1, i.e. the unravelling of this part contains the full binary tree. Thus, this part has a path ρ_A as above for every subset A, but their initial vertices are not necessarily distinct. Hence, we also express that there is another part (labelled by `set`) that contains a copy of each path in the `fbt`-part, and that these paths indeed start at distinct successors of the initial vertex.

We let $\mathcal{T}_{\mathfrak{c}} = (V_{\mathfrak{c}}, E_{\mathfrak{c}}, t_\varepsilon, \lambda_{\mathfrak{c}})$ (see Figure 2), where

$$V_{\mathfrak{c}} = \{t_u \mid u \in \{0,1\}^*\} \cup \{s_A^i \mid i \in \mathbb{N} \wedge A \subseteq \mathbb{N}\}$$

$$\lambda_{\mathfrak{c}}(t_\varepsilon) = \{\mathtt{fbt}\} \quad \lambda_{\mathfrak{c}}(t_{u\cdot 0}) = \{\mathtt{fbt}, 0\} \quad \lambda_{\mathfrak{c}}(t_{u\cdot 1}) = \{\mathtt{fbt}, 1\} \quad \lambda_{\mathfrak{c}}(s_A^i) = \begin{cases} \{\mathtt{set}, 0\} & \text{if } i \notin A \\ \{\mathtt{set}, 1\} & \text{if } i \in A \end{cases}$$

$$E_{\mathfrak{c}} = \{(t_u, t_{u0}), (t_u, t_{u1}) \mid u \in \{0,1\}^*\} \cup \{(t_\varepsilon, s_A^0) \mid A \subseteq \mathbb{N}\} \cup \{(s_A^i, s_A^{i+1}) \mid A \subseteq \mathbb{N}, i \in \mathbb{N}\}.$$

▶ **Lemma 14.** *There is a satisfiable HyperCTL* sentence $\varphi_{\mathfrak{c}}$ that has only models of cardinality at least $\mathfrak{c}$.*

Proof. The formula $\varphi_{\mathfrak{c}}$ is defined as the conjunction of the formulas below:

1. The label of the initial vertex is $\{\mathtt{fbt}\}$ and the labels of non-initial vertices are $\{\mathtt{fbt}, 0\}$, $\{\mathtt{fbt}, 1\}$, $\{\mathtt{set}, 0\}$, or $\{\mathtt{set}, 1\}$:

$$\forall \pi.\ (\mathtt{fbt}_\pi \wedge \neg 0_\pi \wedge \neg 1_\pi \wedge \neg \mathtt{set}_\pi) \wedge \mathbf{X}\,\mathbf{G}\left((\mathtt{set}_\pi \leftrightarrow \neg \mathtt{fbt}_\pi) \wedge (0_\pi \leftrightarrow \neg 1_\pi)\right)$$

2. All `fbt`-labelled vertices have a successor with label $\{\mathtt{fbt}, 0\}$ and one with label $\{\mathtt{fbt}, 1\}$, and all `fbt`-labelled vertices that are additionally labelled by 0 or 1 have no `set`-labelled successor:

$$\forall \pi.\ \mathbf{G}\left(\mathtt{fbt}_\pi \rightarrow ((\exists \pi_0.\ \mathbf{X}(\mathtt{fbt}_{\pi_0} \wedge 0_{\pi_0})) \wedge (\exists \pi_1.\ \mathbf{X}(\mathtt{fbt}_{\pi_1} \wedge 1_{\pi_1})) \wedge ((0_\pi \vee 1_\pi) \rightarrow \forall \pi'.\ \mathbf{X}\,\mathtt{fbt}_{\pi'}))\right)$$

3. For every path of `fbt`-labelled vertices starting at a successor of the initial vertex, there is a path of `set`-labelled vertices (also starting at a successor of the initial vertex) with the same $\{0,1\}$ labelling:

$$\forall \pi.\ \left((\mathbf{X}\,\mathtt{fbt}_\pi) \rightarrow \exists \pi'.\ \mathbf{X}(\mathtt{set}_{\pi'} \wedge \mathbf{G}(0_\pi \leftrightarrow 0_{\pi'}))\right)$$

4. Any two paths starting in the same `set`-labelled vertex have the same sequence of labels:

$$\forall \pi.\ \mathbf{G}\left(\mathtt{set}_\pi \to \forall \pi'.\ \mathbf{G}(0_\pi \leftrightarrow 0_{\pi'})\right).$$

It is easy to check that $\mathcal{T}_\mathfrak{c} \models \varphi_\mathfrak{c}$. Note however that it is not the only model of $\varphi_\mathfrak{c}$: for instance, some paths may be duplicated, or merged after some steps if their label sequences share a common suffix. So, consider an arbitrary transition system $\mathcal{T} = (V, E, v_I, \lambda)$ such that $\mathcal{T} \models \varphi_\mathfrak{c}$. By condition 2, for every set $A \subseteq \mathbb{N}$, there is a path ρ_A starting at a successor of v_I such that $\lambda(\rho_A(i)) = \{\mathtt{fbt}, 1\}$ if $i \in A$ and $\lambda(\rho_A(i)) = \{\mathtt{fbt}, 0\}$ if $i \notin A$. Condition 3 implies that there is also a `set`-labelled path ρ'_A such that ρ'_A starts at a successor of v_I, and has the same $\{0,1\}$ labelling as ρ_A. Finally, by condition 4, if $A \neq B$ then $\rho'_A(0) \neq \rho'_B(0)$. ◀

Before moving to the proof that HyperCTL* satisfiability is Σ^2_1-hard, we introduce one last auxiliary formula that will be used in the reduction, showing that addition and multiplication can be defined in HyperCTL*, and in fact even in HyperLTL, as follows: Let $\mathrm{AP} = \{\mathtt{arg1}, \mathtt{arg2}, \mathtt{res}, \mathtt{add}, \mathtt{mult}\}$ and let $T_{(+,\cdot)}$ be the set of all traces $t \in (2^{\mathrm{AP}})^\omega$ such that

- there are unique $n_1, n_2, n_3 \in \mathbb{N}$ with $\mathtt{arg1} \in t(n_1)$, $\mathtt{arg2} \in t(n_2)$, and $\mathtt{res} \in t(n_3)$, and
- either $\mathtt{add} \in t(n)$ for all n and $n_1 + n_2 = n_3$, or $\mathtt{mult} \in t(n)$ for all n and $n_1 \cdot n_2 = n_3$.

▶ **Lemma 15.** *There is a HyperLTL sentence $\varphi_{(+,\cdot)}$ which has $T_{(+,\cdot)}$ as unique model.*

To establish Σ^2_1-hardness, we give an encoding of formulas of existential third-order arithmetic into HyperCTL*. As explained in Section 2, we can (and do for the remainder of the section) assume that first-order (type 0) variables range over natural numbers, second-order (type 1) variables range over sets of natural numbers, and third-order (type 2) variables range over sets of sets of natural numbers.

▶ **Lemma 16.** *Suppose $\varphi = \exists x_1 \ldots \exists x_n.\ \psi$, where $x_1, \ldots, x_n$ are third-order variables, and ψ is a formula of second-order arithmetic. One can construct a HyperCTL* formula φ' such that $(\mathbb{N}, 0, 1, +, \cdot, <, \in)$ is a model of φ if and only if φ' is satisfiable.*

Proof. The idea of the proof is as follows. We represent sets of natural numbers as infinite paths with labels in $\{0,1\}$, so that quantification over sets of natural numbers in ψ can be replaced by HyperCTL* path quantification. First-order quantification is handled in the same way, but using paths where exactly one vertex is labelled 1. In particular we encode first- and second-order variables x of φ as path variables π_x of φ'. For this to work, we need to make sure that every possible set has a path representative in the transition system (possibly several isomorphic ones). This is where formula $\varphi_\mathfrak{c}$ defined in Lemma 14 is used. For arithmetical operations, we rely on the formula $\varphi_{(+,\cdot)}$ from Lemma 15. Finally, we associate with every existentially quantified third-order variable x_i an atomic proposition a_i, so that for a second-order variable y, the atomic formula $y \in x_i$ is interpreted as the atomic proposition a_i being true on π_y. This is all explained in more details below.

Let $\mathrm{AP} = \{a_1, \ldots, a_n, 0, 1, \mathtt{set}, \mathtt{fbt}, \mathtt{arg1}, \mathtt{arg2}, \mathtt{res}, \mathtt{mult}, \mathtt{add}\}$. Given an interpretation $\nu : \{x_1, \ldots, x_n\} \to 2^{(2^{\mathbb{N}})}$ of the third-order variables of φ, we denote by $\mathcal{T}_\nu$ the transition system over AP obtained as follows: We start from $\mathcal{T}_\mathfrak{c}$, and extend it with an $\{a_1, \ldots, a_n\}$-labelling by setting $a_i \in \lambda(\rho_A(0))$ if $A \in \nu(x_i)$; then, we add to this transition system all traces in $T_{(+,\cdot)}$ as disjoint paths below the initial vertex.

From the formulas $\varphi_\mathfrak{c}$ and $\varphi_{(+,\cdot)}$ defined in Lemmas 14 and 15, it is not difficult to construct a formula $\varphi_{(\mathfrak{c},+,\cdot)}$ such that:

- For all $\nu : \{x_1, \ldots, x_n\} \to 2^{(2^{\mathbb{N}})}$, the transition system $\mathcal{T}_\nu$ is a model of $\varphi_{(\mathfrak{c},+,\cdot)}$.
- Conversely, in any model $\mathcal{T} = (V, E, v_I, \lambda)$ of $\varphi_{(\mathfrak{c},+,\cdot)}$, the following conditions are satisfied:

1. For every path ρ starting at a $\texttt{set}$-labelled successor of the initial vertex v_I, the vertex $\rho(0)$ has a label of the form $\lambda(\rho(0)) = \{\texttt{set}, b\} \cup \ell$ with $b \in \{0,1\}$ and $\ell \subseteq \{a_1, \dots, a_n\}$, and every vertex $\rho(i)$ with $i > 0$ has a label $\lambda(\rho(i)) = \{\texttt{set}, 0\}$ or $\lambda(\rho(i)) = \{\texttt{set}, 1\}$.
2. For every $A \subseteq \mathbb{N}$, there exists a $\texttt{set}$-labelled path ρ_A starting at a successor of v_I such that $1 \in \lambda(\rho_A(i))$ if $i \in A$, and $0 \in \lambda(\rho_A(i))$ if $i \notin A$. Moreover, all such paths have the same $\{a_1, \dots, a_n\}$ labelling; this can be expressed by the formula

$$\forall \pi.\ \forall \pi'.\ \mathbf{X}\left(\mathbf{G}(\texttt{set}_\pi \wedge \texttt{set}_{\pi'} \wedge (1_\pi \leftrightarrow 1_{\pi'})) \rightarrow \bigwedge\nolimits_{a \in \{a_1,\dots,a_n\}} a_\pi \leftrightarrow a_{\pi'}\right).$$

3. For every path ρ starting at an $\texttt{add}$- or $\texttt{mult}$-labelled successor of the initial vertex, the label sequence $\lambda(\rho(0))\lambda(\rho(1))\cdots$ of ρ is in $T_{(+,\cdot)}$.
4. Conversely, for every trace $t \in T_{(+,\cdot)}$, there exists a path ρ starting at a successor of the initial vertex such that $\lambda(\rho(0))\lambda(\rho(1))\cdots = t$.

We then let $\varphi' = \varphi_{(\mathfrak{c},+,\cdot)} \wedge \exists \pi_0.\exists \pi_1.\ \mathbf{X}(1_{\pi_0} \wedge \mathbf{X}\,\mathbf{G}\,0_{\pi_0} \wedge 0_{\pi_1} \wedge \mathbf{X}\,1_{\pi_1} \wedge \mathbf{X}\,\mathbf{X}\,\mathbf{G}\,0_{\pi_1}) \wedge h(\psi)$, where π_0 and π_1 are used to encode the constants 0 and 1, and $h(\psi)$ is defined inductively from the second-order body ψ of φ as follows:

- $h(\psi_1 \vee \psi_2) = h(\psi_1) \vee h(\psi_2)$ and $h(\neg\psi_1) = \neg h(\psi_1)$.
- If x ranges over sets of natural numbers, $h(\exists x.\ \psi_1) = \exists \pi_x.\ ((\mathbf{X}\,\texttt{set}_{\pi_x}) \wedge h(\psi_1))$, and $h(\forall x.\ \psi_1) = \forall \pi_x.\ ((\mathbf{X}\,\texttt{set}_{\pi_x}) \rightarrow h(\psi_1))$.
- If x ranges over natural numbers, $h(\exists x.\ \psi_1) = \exists \pi_x.\ ((\mathbf{X}\,\texttt{set}_{\pi_x}) \wedge \mathbf{X}(0_{\pi_x}\,\mathbf{U}(1_{\pi_x} \wedge \mathbf{X}\,\mathbf{G}\,0_{\pi_x})) \wedge h(\psi_1))$, and $h(\forall x.\ \psi_1) = \forall \pi_x.\ ((\mathbf{X}\,\texttt{set}_{\pi_x}) \wedge \mathbf{X}(0_{\pi_x}\,\mathbf{U}(1_{\pi_x} \wedge \mathbf{X}\,\mathbf{G}\,0_{\pi_x})) \rightarrow h(\psi_1))$.
- If y ranges over sets of natural numbers, $h(y \in x_i) = \mathbf{X}(a_i)_{\pi_y}$.
- If x ranges over natural numbers and y over sets of natural numbers, $h(x \in y) = \mathbf{F}(1_{\pi_x} \wedge 1_{\pi_y})$.
- $h(x < y) = \mathbf{F}(1_{\pi_x} \wedge \mathbf{X}\,\mathbf{F}\,1_{\pi_y})$.
- $h(x \cdot y = z) = \exists \pi.\ (\mathbf{X}\,\texttt{add}_\pi) \wedge \mathbf{F}(\texttt{arg1}_\pi \wedge 1_{\pi_x}) \wedge \mathbf{F}(\texttt{arg2}_\pi \wedge 1_{\pi_y}) \wedge \mathbf{F}(\texttt{res}_\pi \wedge 1_{\pi_z})$, and $h(x + y = z) = \exists \pi.\ (\mathbf{X}\,\texttt{mult}_\pi) \wedge \mathbf{F}(\texttt{arg1}_\pi \wedge 1_{\pi_x}) \wedge \mathbf{F}(\texttt{arg2}_\pi \wedge 1_{\pi_y}) \wedge \mathbf{F}(\texttt{res}_\pi \wedge 1_{\pi_z})$.

If ψ is true under some interpretation ν of $x_1, \dots, x_n$ as sets of sets of natural numbers, then the transition system $\mathcal{T}_\nu$ defined above is a model of φ'. Conversely, if $\mathcal{T} \models \varphi'$ for some transition system $\mathcal{T}$, then for all sets $A \subseteq \mathbb{N}$ there is a path ρ_A matching A in $\mathcal{T}$, and all such paths have the same $\{a_1, \dots, a_n\}$-labelling, so we can define an interpretation ν of $x_1, \dots, x_n$ by taking $A \in \nu(x_i)$ if and only if $a_i \in \lambda(\rho_A(0))$. Under this interpretation ψ holds, and thus φ is true. ◀

▶ **Lemma 17.** *HyperCTL* satisfiability is Σ^2_1-hard.*

Proof. Let N be a Σ^2_1 set, i.e. $N = \{x \in \mathbb{N} \mid \exists x_0 \cdots \exists x_k.\ \psi(x, x_0, \dots, x_k)\}$ for some second-order arithmetic formula ψ with existentially quantified third-order variables x_i. For every $n \in \mathbb{N}$, we define a sentence

$$\varphi_n = \exists x_0 \cdots \exists x_k.\ (\exists x.\ x = 0 \underbrace{+1+1+\cdots+1}_{n \text{ times}} \wedge \psi(x, x_0, \dots, x_k)).$$

Then φ_n is true if and only if $n \in N$. Combining this with Lemma 16, we obtain a computable function that maps any $n \in \mathbb{N}$ to a HyperCTL* formula φ'_n such that $n \in N$ if and only if φ'_n is satisfiable. ◀

6 Conclusion

In this work, we have settled the complexity of the satisfiability problems for HyperLTL and HyperCTL*. In both cases, we significantly increased the lower bounds, i.e. from Σ^0_1 and Σ^1_1 to Σ^1_1 and Σ^2_1, respectively, and presented the first upper bounds, which are tight in both cases. Along the way, we also determined the complexity of restricted variants, e.g. HyperLTL satisfiability restricted to ultimately periodic traces (or, equivalently, to finite traces) is still Σ^1_1-complete while HyperCTL* satisfiability restricted to finite transition systems is Σ^0_1-complete. As a key step in this proof, we showed a tight bound of $\mathfrak{c}$ on the size of minimal models for satisfiable HyperCTL* sentences. Finally, we also show that deciding membership in any level of the HyperLTL quantifier alternation hierarchy is Π^1_1-complete.

References

1 Erika Ábrahám, Ezio Bartocci, Borzoo Bonakdarpour, and Oyendrila Dobe. Probabilistic hyperproperties with nondeterminism. In Dang Van Hung and Oleg Sokolsky, editors, *ATVA 2020*, volume 12302 of *LNCS*, pages 518–534. Springer, 2020. doi:10.1007/978-3-030-59152-6_29.

2 Erika Ábrahám and Borzoo Bonakdarpour. HyperPCTL: A temporal logic for probabilistic hyperproperties. In Annabelle McIver and András Horváth, editors, *QEST 2018*, volume 11024 of *LNCS*, pages 20–35. Springer, 2018. doi:10.1007/978-3-319-99154-2_2.

3 Shreya Agrawal and Borzoo Bonakdarpour. Runtime verification of k-safety hyperproperties in HyperLTL. In *CSF 2016*, pages 239–252. IEEE Computer Society, 2016. doi:10.1109/CSF.2016.24.

4 Gilles Barthe, Pedro R. D'Argenio, Bernd Finkbeiner, and Holger Hermanns. Facets of software doping. In Tiziana Margaria and Bernhard Steffen, editors, *ISoLA 2016, Proceedings, Part II*, volume 9953 of *LNCS*, pages 601–608, 2016. doi:10.1007/978-3-319-47169-3_46.

5 Ezio Bartocci, Thomas Ferrère, Thomas A. Henzinger, Dejan Nickovic, and Ana Oliveira da Costa. Flavours of sequential information flow. *arXiv*, 2021. arXiv:2105.02013.

6 Jan Baumeister, Norine Coenen, Borzoo Bonakdarpour, Bernd Finkbeiner, and César Sánchez. A temporal logic for asynchronous hyperproperties. *arXiv*, 2021. arXiv:2104.14025.

7 Béatrice Bérard, Stefan Haar, and Loïc Hélouët. Hyper partial order logic. In Sumit Ganguly and Paritosh K. Pandya, editors, *FSTTCS 2018*, volume 122 of *LIPIcs*, pages 20:1–20:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.FSTTCS.2018.20.

8 Borzoo Bonakdarpour and Bernd Finkbeiner. Runtime verification for HyperLTL. In Yliès Falcone and César Sánchez, editors, *RV 2016*, volume 10012 of *LNCS*, pages 41–45. Springer, 2016. doi:10.1007/978-3-319-46982-9_4.

9 Borzoo Bonakdarpour and Bernd Finkbeiner. Controller synthesis for hyperproperties. In *CSF 2020*, pages 366–379. IEEE, 2020. doi:10.1109/CSF49147.2020.00033.

10 Laura Bozzelli, Adriano Peron, and Cesar Sanchez. Asynchronous extensions of hyperltl, 2021. arXiv:2104.12886.

11 Noel Brett, Umair Siddique, and Borzoo Bonakdarpour. Rewriting-based runtime verification for alternation-free HyperLTL. In Axel Legay and Tiziana Margaria, editors, *TACAS 2017, Part II*, volume 10206 of *LNCS*, pages 77–93, 2017. doi:10.1007/978-3-662-54580-5_5.

12 Michael R. Clarkson, Bernd Finkbeiner, Masoud Koleini, Kristopher K. Micinski, Markus N. Rabe, and César Sánchez. Temporal logics for hyperproperties. In Martín Abadi and Steve Kremer, editors, *POST 2014*, volume 8414 of *LNCS*, pages 265–284. Springer, 2014. doi:10.1007/978-3-642-54792-8_15.

13 Michael R. Clarkson and Fred B. Schneider. Hyperproperties. *J. Comput. Secur.*, 18(6):1157–1210, 2010. doi:10.3233/JCS-2009-0393.

14 Norine Coenen, Bernd Finkbeiner, Christopher Hahn, and Jana Hofmann. The hierarchy of hyperlogics. In *LICS 2019*, pages 1–13. IEEE, 2019. doi:10.1109/LICS.2019.8785713.

15 Rina S. Cohen and Janusz A. Brzozowski. Dot-depth of star-free events. *J. Comput. Syst. Sci.*, 5(1):1–16, 1971. doi:10.1016/S0022-0000(71)80003-X.

16 Rayna Dimitrova, Bernd Finkbeiner, and Hazem Torfah. Probabilistic hyperproperties of Markov decision processes. In Dang Van Hung and Oleg Sokolsky, editors, *ATVA 2020*, volume 12302 of *LNCS*, pages 484–500. Springer, 2020. doi:10.1007/978-3-030-59152-6_27.

17 E. Allen Emerson and Joseph Y. Halpern. "Sometimes" and "not never" revisited: on branching versus linear time temporal logic. *J. ACM*, 33(1):151–178, 1986. doi:10.1145/4904.4999.

18 Bernd Finkbeiner. Model checking algorithms for hyperproperties (invited paper). In Fritz Henglein, Sharon Shoham, and Yakir Vizel, editors, *VMCAI 2021*, volume 12597 of *LNCS*, pages 3–16. Springer, 2021. doi:10.1007/978-3-030-67067-2_1.

19 Bernd Finkbeiner and Christopher Hahn. Deciding hyperproperties. In Josée Desharnais and Radha Jagadeesan, editors, *CONCUR 2016*, volume 59 of *LIPIcs*, pages 13:1–13:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CONCUR.2016.13.

20 Bernd Finkbeiner, Christopher Hahn, and Tobias Hans. MGHyper: Checking satisfiability of HyperLTL formulas beyond the $\exists^*\forall^*$ fragment. In *ATVA 2018*, volume 11138 of *LNCS*, pages 521–527. Springer, 2018. doi:10.1007/978-3-030-01090-4_31.

21 Bernd Finkbeiner, Christopher Hahn, Jana Hofmann, and Leander Tentrup. Realizing omega-regular hyperproperties. In Shuvendu K. Lahiri and Chao Wang, editors, *CAV 2020, Part II*, volume 12225 of *LNCS*, pages 40–63. Springer, 2020. doi:10.1007/978-3-030-53291-8_4.

22 Bernd Finkbeiner, Christopher Hahn, Philip Lukert, Marvin Stenger, and Leander Tentrup. Synthesis from hyperproperties. *Acta Informatica*, 57(1-2):137–163, 2020. doi:10.1007/s00236-019-00358-2.

23 Bernd Finkbeiner, Christopher Hahn, and Marvin Stenger. EAHyper: Satisfiability, Implication, and Equivalence Checking of Hyperproperties. In Rupak Majumdar and Viktor Kuncak, editors, *CAV 2017, Part II*, volume 10427 of *LNCS*, pages 564–570. Springer, 2017. doi:10.1007/978-3-319-63390-9_29.

24 Bernd Finkbeiner, Christopher Hahn, Marvin Stenger, and Leander Tentrup. RVHyper: A runtime verification tool for temporal hyperproperties. In Dirk Beyer and Marieke Huisman, editors, *TACAS 2018, Part II*, volume 10806 of *LNCS*, pages 194–200. Springer, 2018. doi:10.1007/978-3-319-89963-3_11.

25 Bernd Finkbeiner, Markus N. Rabe, and César Sánchez. Algorithms for Model Checking HyperLTL and HyperCTL*. In Daniel Kroening and Corina S. Pasareanu, editors, *CAV 2015, Part I*, volume 9206 of *LNCS*, pages 30–48. Springer, 2015. doi:10.1007/978-3-319-21690-4_3.

26 Bernd Finkbeiner and Martin Zimmermann. The First-Order Logic of Hyperproperties. In *STACS 2017*, volume 66 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.STACS.2017.30.

27 Marie Fortin, Louwe B. Kuijer, Patrick Totzke, and Martin Zimmermann. HyperLTL satisfiability is Σ^1_1-complete, HyperCTL* satisfiability is Σ^2_1-complete. *arXiv*, 2021. arXiv:2105.04176.

28 Jens Oliver Gutsfeld, Markus Müller-Olm, and Christoph Ohrem. Propositional dynamic logic for hyperproperties. In Igor Konnov and Laura Kovács, editors, *CONCUR 2020*, volume 171 of *LIPIcs*, pages 50:1–50:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CONCUR.2020.50.

29 David Harel. Recurring Dominoes: Making the Highly Undecidable Highly Understandable. *North-Holland Mathematical Studies*, 102:51–71, 1985. doi:10.1016/S0304-0208(08)73075-5.

30 Peter G. Hinman. *Recursion-Theoretic Hierarchies*. Perspectives in Logic. Cambridge University Press, 2017. doi:10.1017/9781316717110.

31 Hsi-Ming Ho, Ruoyu Zhou, and Timothy M. Jones. Timed hyperproperties. *Information and Computation*, page 104639, 2020. doi:10.1016/j.ic.2020.104639.

32 Andreas Krebs, Arne Meier, Jonni Virtema, and Martin Zimmermann. Team semantics for the specification and verification of hyperproperties. In Igor Potapov, Paul G. Spirakis, and James Worrell, editors, *MFCS 2018*, volume 117 of *LIPIcs*, pages 10:1–10:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.MFCS.2018.10.

33 Corto Mascle and Martin Zimmermann. The keys to decidable HyperLTL satisfiability: Small models or very simple formulas. In Maribel Fernández and Anca Muscholl, editors, *CSL 2020*, volume 152 of *LIPIcs*, pages 29:1–29:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CSL.2020.29.

34 Amir Pnueli. The temporal logic of programs. In *FOCS 1977*, pages 46–57. IEEE, October 1977. doi:10.1109/SFCS.1977.32.

35 Markus N. Rabe. *A temporal logic approach to information-flow control*. PhD thesis, Saarland University, 2016. URL: http://scidok.sulb.uni-saarland.de/volltexte/2016/6387/.

36 Hartley Rogers. *Theory of Recursive Functions and Effective Computability*. MIT Press, Cambridge, MA, USA, 1987.

37 Wolfgang Thomas. A combinatorial approach to the theory of omega-automata. *Inf. Control.*, 48(3):261–283, 1981. doi:10.1016/S0019-9958(81)90663-X.

38 Wolfgang Thomas. Classifying regular events in symbolic logic. *J. Comput. Syst. Sci.*, 25(3):360–376, 1982. doi:10.1016/0022-0000(82)90016-2.

39 Jonni Virtema, Jana Hofmann, Bernd Finkbeiner, Juha Kontinen, and Fan Yang. Linear-time temporal logic with team semantics: Expressivity and complexity. *arXiv*, 2020. arXiv:2010.03311.