Inf. Process. Lett. 193 (2026) 106626

Contents lists available at ScienceDirect

Information Processing Letters

journal homepage: www.elsevier.com/locate/ipl

The complexity of HyperQPTL

Gaëtan Regaud [a], Martin Zimmermann [b,*]

[a] ENS Rennes, Rennes, France
[b] Aalborg University, Aalborg, Denmark

ARTICLE INFO

Keywords:
HyperQPTL
Hyperlogics
Satisfiability
Model-checking

ABSTRACT

HyperQPTL and HyperQPTL$^+$ are expressive specification languages for hyperproperties, properties that relate multiple executions of a system. Tight complexity bounds are known for HyperQPTL finite-state satisfiability and model-checking.

Here, we settle the complexity of satisfiability for HyperQPTL as well as satisfiability, finite-state satisfiability, and model-checking for HyperQPTL$^+$: the former is Σ^2_1-complete, the latter are all equivalent to truth in third-order arithmetic, i.e., all four are very undecidable.

1. Introduction

Hyperproperties [1] are properties relating multiple executions of a system and have found applications in security and privacy, epistemic reasoning, and verification. Temporal logics have been introduced to express hyperproperties, e.g., HyperLTL and HyperCTL* [2] (which extend LTL and CTL* with trace quantification), Hyper^{2}LTL [3] (second-order HyperLTL, which extends HyperLTL with quantification over sets of traces), and many more. Here, we focus on the most important verification problems:

- Satisfiability: Given a sentence φ, is there a nonempty set T of traces such that $T \vDash \varphi$?
- Finite-state satisfiability: Given a sentence φ, is there a finite transition system $\mathfrak{T}$ such that $\mathfrak{T} \vDash \varphi$?
- Model-checking: Given a sentence φ and a finite transition system $\mathfrak{T}$, do we have $\mathfrak{T} \vDash \varphi$?

This work is part of a research program [4–10] settling the complexity of these problems for hyperlogics. Most importantly for applications in verification, the model-checking problems for HyperLTL and HyperCTL* are decidable, albeit TOWER-complete [4,5,11]. However, satisfiability is typically much harder. In fact, the satisfiability problems are typically highly undecidable, i.e., we measure their complexity by placing them in the arithmetic or analytic hierarchy, or even beyond: Intuitively, first-order arithmetic is predicate logic over the signature $(+, \cdot, <)$ where quantification ranges over natural numbers. Similarly, second-order arithmetic adds quantification over sets of natural numbers to first-order arithmetic while third-order arithmetic adds quantification over sets of sets of natural numbers to second-order quantification. Fig. 1 depicts the arithmetic, analytic, and "third" hierarchy, each spanned by the classes of languages definable by restricting the number of alternations of the highest-order quantifiers, e.g., Σ^0_n contains languages definable by formulas of first-order arithmetic with $n-1$ quantifier alternations, starting with an existential one.

Here, we study the logics HyperQPTL and HyperQPTL$^+$ which extend HyperLTL by quantification over propositions [5,12], just as QPTL extends LTL with quantification over propositions [13]. QPTL is, unlike LTL, able to express all ω-regular properties while HyperQPTL is, unlike HyperLTL, able to express, e.g., promptness and epistemic properties [12].

The difference between HyperQPTL and HyperQPTL$^+$ manifests itself in the semantics of propositional quantification. Recall that hyperlogics are evaluated over sets T of traces. In HyperQPTL, the quantification of a proposition p reassigns the truth value of p in T in a uniform way over all traces, i.e., after quantifying p all traces coincide on their truth values for p. In HyperQPTL$^+$ on the other hand, the quantification of a proposition p reassigns truth values of p to traces in T individually. Said differently, in HyperQPTL one quantifies over a single sequence of truth values (i.e., a sequence in $\{0,1\}^\omega$) while in HyperQPTL$^+$ one quantifies over a set of sequences of truth values (i.e., a subset of $\{0,1\}^\omega$). Hence, one expects HyperQPTL$^+$ to be more expressive than HyperQPTL.

And indeed, Finkbeiner et al. showed that HyperQPTL$^+$ model-checking is undecidable [12], while it is TOWER-complete for HyperQPTL [5]. It is also known that HyperQPTL finite-state satisfiability is Σ^0_1-complete [5], i.e., complete for the recursively-enumerable languages, but the exact complexity of HyperQPTL satisfiability is open:

* Corresponding author.
E-mail addresses: gaetan.regaud@ens-rennes.fr (G. Regaud), mzi@cs.aau.dk (M. Zimmermann).

https://doi.org/10.1016/j.ipl.2026.106626
Received 10 December 2024; Received in revised form 11 February 2026; Accepted 14 February 2026
Available online 17 February 2026

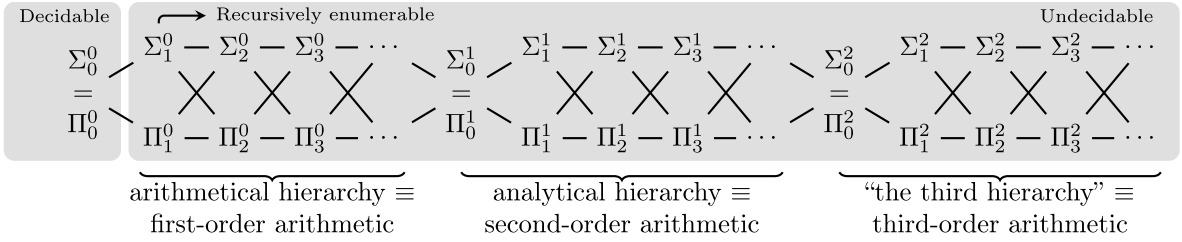

Fig. 1. The arithmetical hierarchy, the analytical hierarchy, and beyond.

it is only known to be Σ^1_1-hard, as satisfiability is already Σ^1_1-complete for its fragment HyperLTL [6,7]. For HyperQPTL$^+$, much less is known: As mentioned above, model-checking HyperQPTL$^+$ is undecidable [12], but its exact complexity is open, as is the complexity of satisfiability and finite-state satisfiability.

Here, we settle the complexity of all four open problems by showing that HyperQPTL satisfiability is Σ^2_1-complete (Section 3) while all three problems for HyperQPTL$^+$ are equivalent to truth in third-order arithmetic (Section 4). These latter results are obtained by showing that HyperQPTL$^+$ and Hyper^{2}LTL have the same expressiveness. This confirms the expectation that HyperQPTL$^+$ is more expressive than HyperQPTL: the non-uniform quantification of propositions allows to simulate quantification over arbitrary sets of traces.

2. Preliminaries

We denote the nonnegative integers by $\mathbb{N}$. An alphabet is a nonempty finite set. The set of infinite words over an alphabet Σ is denoted by Σ^ω. Let AP be a nonempty finite set of atomic propositions. A trace over AP is an infinite word over the alphabet 2^{AP}. Given a subset $\mathrm{AP}' \subseteq \mathrm{AP}$, the AP$'$-projection of a trace $t(0)t(1)t(2)\cdots$ over AP is the trace $(t(0) \cap \mathrm{AP}')(t(1) \cap \mathrm{AP}')(t(2) \cap \mathrm{AP}')\cdots \in (2^{\mathrm{AP}'})^\omega$. The AP$'$-projection of $T \subseteq (2^{\mathrm{AP}})^\omega$ is defined as the set of AP$'$-projections of traces in T. We write $t_0 =_{\mathrm{AP}'} t_1$ ($T_0 =_{\mathrm{AP}'} T_1$) if the AP$'$-projections of t_0 and t_1 (T_0 and T_1) are equal. Now, let AP and AP$'$ be two disjoint sets, let t be a trace over AP, and let t' be a trace over AP$'$. Then, we define $t^\frown t'$ as the pointwise union of t and t', i.e., $t^\frown t'$ is the trace over $\mathrm{AP} \cup \mathrm{AP}'$ defined as $(t(0) \cup t'(0))(t(1) \cup t'(1))(t(2) \cup t'(2))\cdots$.

A transition system $\mathfrak{T} = (V, E, I, \lambda)$ consists of a finite nonempty set V of vertices, a set $E \subseteq V \times V$ of (directed) edges, a nonempty set $I \subseteq V$ of initial vertices, and a labeling $\lambda\colon V \to 2^{\mathrm{AP}}$ of the vertices by sets of atomic propositions. We assume that every vertex has at least one outgoing edge. A path ρ through $\mathfrak{T}$ is an infinite sequence $\rho(0)\rho(1)\rho(2)\cdots$ of vertices with $\rho(0) \in I$ and $(\rho(n), \rho(n+1)) \in E$ for every $n \geq 0$. The trace of ρ is defined as $\lambda(\rho) = \lambda(\rho(0))\lambda(\rho(1))\lambda(\rho(2))\cdots$. The set of traces of $\mathfrak{T}$ is $\mathrm{Tr}(\mathfrak{T}) = \{\lambda(\rho) \mid \rho \text{ is a path through } \mathfrak{T}\}$. Note that this set is always nonempty, as I is nonempty and as there are no terminal vertices.

To capture the complexity of undecidable problems, we consider formulas of arithmetic, i.e., predicate logic with signature $(+, \cdot, <, \in)$, evaluated over the structure $(\mathbb{N}, +, \cdot, <, \in)$ (see, e.g., [14]). A type 0 object is a natural number in $\mathbb{N}$, a type 1 object is a subset of $\mathbb{N}$, and a type 2 object is a set of subsets of $\mathbb{N}$.

First-order arithmetic allows to quantify over type 0 objects, second-order arithmetic allows to quantify over type 0 and type 1 objects, and third-order arithmetic allows to quantify over type 0, type 1, and type 2 objects. Note that every fixed natural number is definable in first-order arithmetic, so we freely use them as syntactic sugar. Similarly, equality can be expressed using $<$, so we use it as syntactic sugar as well.

Truth in third-order arithmetic is the following problem: given a sentence φ of third-order arithmetic, does $(\mathbb{N}, +, \cdot, <, \in)$ satisfy φ (written $(\mathbb{N}, +, \cdot, <, \in) \vDash \varphi$ as usual)? Furthermore, Σ^2_1 contains the sets of the form $\{n \in \mathbb{N} \mid (\mathbb{N}, +, \cdot, <, \in) \vDash \varphi(n)\}$, where φ is a formula of the form $\exists \mathcal{Y}_1 \ldots \exists \mathcal{Y}_k.\, \psi(x, \mathcal{Y}_1, \ldots, \mathcal{Y}_k,)$ where the $\mathcal{Y}_j$ are third-order variables and ψ is a formula of arithmetic containing only first- and second-order quantifiers.

3. HyperQPTL

In this section, we introduce HyperQPTL and then settle the complexity of its satisfiability problem.

3.1. Syntax and semantics

Let $\mathcal{V}$ be a countable set of trace variables. The formulas of HyperQPTL [5,12] are given by the grammar

$$\varphi ::= \exists \pi.\, \varphi \mid \forall \pi.\, \varphi \mid \exists \mathrm{q}.\, \varphi \mid \forall \mathrm{q}.\, \varphi \mid \psi$$
$$\psi ::= \mathrm{p}_\pi \mid \mathrm{q} \mid \neg\psi \mid \psi \vee \psi \mid \mathbf{X}\,\psi \mid \mathbf{F}\,\psi$$

where p and q range over AP and where π ranges over $\mathcal{V}$. Note that there are two types of atomic formulas, i.e., propositions labeled by traces on which they are evaluated (p_π with $\mathrm{p} \in \mathrm{AP}$ and $\pi \in \mathcal{V}$) and unlabeled propositions ($\mathrm{q} \in \mathrm{AP}$). We use different letters for the propositions in these cases, but let us stress again that both p and q are propositions in AP. A formula is a sentence, if every occurrence of an atomic formula p_π is in the scope of a quantifier binding π and every occurrence of an atomic formula q is in the scope of a quantifier binding q. Finally, note that the only temporal operators we have in the syntax are next (**X**) and eventually (**F**), as the other temporal operators like always (**G**) and until (**U**) are syntactic sugar in HyperQPTL [15]. Hence, we use them freely in the following, just as conjunction, implication, and equivalence.

A trace assignment is a partial mapping $\Pi\colon \mathcal{V} \to (2^{\mathrm{AP}})^\omega$. Given a variable $\pi \in \mathcal{V}$, and a trace t, we denote by $\Pi[\pi \mapsto t]$ the trace assignment that coincides with Π on all variables but π, which is mapped to t. Let $\mathrm{q} \in \mathrm{AP}$, let $t \in (2^{\mathrm{AP}})^\omega$ be a trace over AP, and let $t_\mathrm{q} \in (2^{\{\mathrm{q}\}})^\omega$ be a trace over $\{\mathrm{q}\}$. We define the trace $t[\mathrm{q} \mapsto t_\mathrm{q}] = t'^\frown t_\mathrm{q}$, where t' is the $(\mathrm{AP} \setminus \{\mathrm{q}\})$-projection of t: Intuitively, the occurrences of q in t are replaced according to t_q. We lift this to sets T of traces by defining $T[\mathrm{q} \mapsto t_\mathrm{q}] = \{t[\mathrm{q} \mapsto t_\mathrm{q}] \mid t \in T\}$. All traces in $T[\mathrm{q} \mapsto t_\mathrm{q}]$ have the same $\{\mathrm{q}\}$-projection, i.e., t_q.

Now, for a trace assignment Π, a position $i \in \mathbb{N}$, and a set T of traces, we define

- $T, \Pi, i \vDash \mathrm{p}_\pi$ if $\mathrm{p} \in \Pi(\pi)(i)$,
- $T, \Pi, i \vDash \mathrm{q}$ if for all $t \in T$ we have $\mathrm{q} \in t(i)$,
- $T, \Pi, i \vDash \neg\psi$ if $T, \Pi, i \nvDash \psi$,
- $T, \Pi, i \vDash \psi_1 \vee \psi_2$ if $T, \Pi, i \vDash \psi_1$ or $T, \Pi, i \vDash \psi_2$,
- $T, \Pi, i \vDash \mathbf{X}\,\psi$ if $T, \Pi, i+1 \vDash \psi$,
- $T, \Pi, i \vDash \mathbf{F}\,\psi$ if there is an $i' \geq i$ with $T, \Pi, i' \vDash \psi$,
- $T, \Pi, i \vDash \exists \pi.\, \varphi$ if there exists a $t \in T$ such that $T, \Pi[\pi \mapsto t], i \vDash \varphi$,
- $T, \Pi, i \vDash \forall \pi.\, \varphi$ if $T, \Pi[\pi \mapsto t], i \vDash \varphi$ for all $t \in T$,
- $T, \Pi, i \vDash \exists \mathrm{q}.\, \varphi$ if there exists a $t_\mathrm{q} \in (2^{\{\mathrm{q}\}})^\omega$ such that $T[\mathrm{q} \mapsto t_\mathrm{q}], \Pi, i \vDash \varphi$, and
- $T, \Pi, i \vDash \forall \mathrm{q}.\, \varphi$ if $T[\mathrm{q} \mapsto t_\mathrm{q}], \Pi, i \vDash \varphi$ for all $t_\mathrm{q} \in (2^{\{\mathrm{q}\}})^\omega$.

We say that a nonempty set T of traces satisfies a sentence φ, written $T \vDash \varphi$, if $T, \Pi_\emptyset, 0 \vDash \varphi$ where $\Pi_\emptyset$ is the trace assignment with empty domain. We then also say that T is a model of φ. A transition system $\mathfrak{T}$ satisfies φ, written $\mathfrak{T} \vDash \varphi$, if $\mathrm{Tr}(\mathfrak{T}) \vDash \varphi$.

While it is known that HyperQPTL finite-state satisfiability is Σ^0_1-complete [5], i.e., complete for the recursively-enumerable languages, and HyperQPTL model-checking is TOWER-complete [5], the exact complexity of HyperQPTL satisfiability is open: it is only known to be Σ^1_1-hard, as satisfiability is Σ^1_1-complete for its fragment HyperLTL [7].

3.2. HyperQPTL satisfiability

In this subsection, we study the HyperQPTL satisfiability problem, which asks whether, for a given HyperQPTL sentence φ, there is a nonempty T such that $T \vDash \varphi$. We show that it is Σ^2_1-complete: intuitively, quantification over traces is equivalent to quantification over subsets of $\mathbb{N}$: a trace t encodes the set $\{n \in \mathbb{N} \mid \mathtt{p} \in t(n)\}$ and vice versa, where $\mathtt{p}$ is some designated proposition. Furthermore, existential quantification over sets of sets of natural numbers is equivalent to the selection of a set of traces at the meta-level of the satisfiability problem.

For this to be correct, we need to enforce that the model of a HyperQPTL sentence contains enough traces to encode all subsets of $\mathbb{N}$. As usual, we use $\mathfrak{c}$ to denote the cardinality of the continuum, or equivalently, the cardinality of $(2^{\mathrm{AP}})^\omega$ for each finite AP and the cardinality of $2^{\mathbb{N}}$. As models of HyperQPTL sentences are sets of traces, $\mathfrak{c}$ is a trivial upper bound on the size of models. It is straightforward to show that this upper bound is tight, which also implies that there are sentences whose models allow us to encode all subsets of $\mathbb{N}$.

Theorem 1. *There is a satisfiable* HyperQPTL *sentence that has only models of cardinality* $\mathfrak{c}$.

Proof. Fix $\mathrm{AP} = \{\mathtt{x}, \mathtt{q}\}$ and let $\theta_{all} = \forall \mathtt{q}.\ \exists \pi.\ \mathbf{G}(\mathtt{q} \leftrightarrow \mathtt{x}_\pi)$, which requires that the $\{\mathtt{x}\}$-projection of any model of φ_{all} is equal to $(2^{\{\mathtt{x}\}})^\omega$. As $(2^{\{\mathtt{x}\}})^\omega$ has cardinality $\mathfrak{c}$, every model of θ_{all} has cardinality $\mathfrak{c}$. □

This, and the fact that addition and multiplication can be "implemented" in HyperLTL [7] suffice to prove our lower bound on HyperQPTL satisfiability.

Lemma 1. HyperQPTL *satisfiability is* Σ^2_1*-hard.*

Proof. We need to show that every language in Σ^2_1 is polynomial-time reducible to HyperQPTL satisfiability. To this end, we first present a polynomial-time computable function f mapping sentences of the form $\varphi = \exists \mathcal{Y}_1 \dots \exists \mathcal{Y}_k.\ \psi$, with third-order variables $\mathcal{Y}_j$ and with ψ only containing first-order and second-order quantifiers, to HyperQPTL sentences $f(\varphi)$ such that $(\mathbb{N}, +, \cdot, <, \in) \vDash \varphi$ if and only if $f(\varphi)$ is satisfiable. To this end, we fix $\mathrm{AP} = \{\mathtt{x}, \mathtt{q}, \mathtt{m}_1, \dots, \mathtt{m}_k, \mathtt{arg1}, \mathtt{arg2}, \mathtt{res}, \mathtt{add}, \mathtt{mult}\}$. We explain the roles of the different propositions along the way.

Recall that the sentence θ_{all} ensures that the $\{\mathtt{x}\}$-projection of each of its models is equal to $(2^{\{\mathtt{x}\}})^\omega$. Hence, by ignoring all propositions but $\mathtt{x}$, we can use traces to encode sets of natural numbers and natural numbers (as singleton sets) and each model contains the encoding of each set of natural numbers. In our encoding, a trace bound to a trace variable π encodes a singleton set if and only if the formula $(\neg \mathtt{x}_\pi)\,\mathbf{U}(\mathtt{x}_\pi \wedge \mathbf{X}\,\mathbf{G}\,\neg \mathtt{x}_\pi)$ is satisfied.

Thus, a set of sets of natural numbers can be encoded by marking traces (encoding sets) by a marker signifying whether it is in the set (the marker holds at the first position) or not (the marker does not hold at the first position). Note that several traces encode the same set, hence we need to ensure that the marking is consistent among these traces. We use a distinct marker $\mathtt{m}_j$ for each $\mathcal{Y}_j$ appearing in φ and the formula

$$\theta_{cons} = \bigwedge\nolimits_{j=1}^{k} \forall \pi.\ \mathbf{X}\,\mathbf{G}\,\neg(\mathtt{m}_j)_\pi \wedge$$
$$\forall \pi'.\ \Big(\big(\,\mathbf{G}(\mathtt{x}_\pi \leftrightarrow \mathtt{x}_{\pi'})\big) \rightarrow \big((\mathtt{m}_j)_\pi \leftrightarrow (\mathtt{m}_j)_{\pi'}\big)\Big)$$

to express consistency: the first line expresses that only the first position of a trace may (but must not) be labeled with markers and the second line expresses that when two traces have the same truth value of $\mathtt{x}$ everywhere (i.e., they encode the same set), then they have the same marking.

So, every model T of $\theta_{all} \wedge \theta_{cons}$ encodes every subset of $\mathbb{N}$ as a trace in T and also k sets of subsets of $\mathbb{N}$ via the (consistent) marking of traces, i.e., the existential quantification over a model in the satisfiability problem simulates the existential third-order quantification of the $\mathcal{Y}_j$.

Finally, Fortin et al. showed that addition and multiplication can be "implemented" in HyperLTL [7]: 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

- $\mathtt{add} \in t(n)$ and $\mathtt{mult} \notin t(n)$ for all n, and $n_1 + n_2 = n_3$, or
- $\mathtt{mult} \in t(n)$ and $\mathtt{add} \notin t(n)$ for all n, and $n_1 \cdot n_2 = n_3$.

There is a satisfiable HyperLTL sentence $\theta_{(+,\cdot)}$ such that the $\{\mathtt{arg1}, \mathtt{arg2}, \mathtt{res}, \mathtt{add}, \mathtt{mult}\}$-projection of every model of $\theta_{(+,\cdot)}$ is $T_{(+,\cdot)}$ [7, Theorem 5.5]. As HyperLTL is a fragment of HyperQPTL, we can use $\theta_{(+,\cdot)}$ to construct our desired formula.

Now, we define for $\varphi = \exists \mathcal{Y}_1 \dots \exists \mathcal{Y}_k.\ \psi$

$$f(\varphi) = \theta_{all} \wedge \theta_{cons} \wedge \theta_{(+,\cdot)} \wedge hyp(\psi)$$

where $hyp(\psi)$ is defined inductively as follows:

- For second-order variables Y, $hyp(\exists Y.\ \psi) = \exists \pi_Y.\ hyp(\psi)$ and $hyp(\forall Y.\ \psi) = \forall \pi_Y.\ hyp(\psi)$.
- For first-order variables y, $hyp(\exists y.\ \psi) = \exists \pi_y.\ ((\neg \mathtt{x}_{\pi_y})\,\mathbf{U}(\mathtt{x}_{\pi_y} \wedge \mathbf{X}\,\mathbf{G}\,\neg \mathtt{x}_{\pi_y})) \wedge hyp(\psi)$ and $hyp(\forall y.\ \psi) = \forall \pi_y.\ ((\neg \mathtt{x}_{\pi_y})\,\mathbf{U}(\mathtt{x}_{\pi_y} \wedge \mathbf{X}\,\mathbf{G}\,\neg \mathtt{x}_{\pi_y})) \rightarrow hyp(\psi)$,
- $hyp(\psi_1 \vee \varphi_2) = hyp(\psi_1) \vee hyp(\psi_2)$,
- $hyp(\neg\psi) = \neg hyp(\psi)$,
- For third-order variables $\mathcal{Y}_j$ and second-order variables Y, $hyp(Y \in \mathcal{Y}_j) = (\mathtt{m}_j)_{\pi_Y}$,
- For second-order variables Y and first-order variables y, $hyp(y \in Y) = \mathbf{F}(\mathtt{x}_{\pi_y} \wedge \mathtt{x}_{\pi_Y})$,
- For first-order variables y, y', $hyp(y < y') = \mathbf{F}(\mathtt{x}_{\pi_y} \wedge \mathbf{X}\,\mathbf{F}\,\mathtt{x}_{\pi_{y'}})$,
- For first-order variables y_1, y_2, y, $hyp(y_1 + y_2 = y) = \exists \pi.\ \mathtt{add}_\pi \wedge \mathbf{F}(\mathtt{x}_{\pi_{y_1}} \wedge \mathtt{arg1}_\pi) \wedge \mathbf{F}(\mathtt{x}_{\pi_{y_2}} \wedge \mathtt{arg2}_\pi) \wedge \mathbf{F}(\mathtt{x}_{\pi_y} \wedge \mathtt{res}_\pi)$ and $hyp(y_1 \cdot y_2 = y) = \exists \pi.\ \mathtt{mult}_\pi \wedge \mathbf{F}(\mathtt{x}_{\pi_{y_1}} \wedge \mathtt{arg1}_\pi) \wedge \mathbf{F}(\mathtt{x}_{\pi_{y_2}} \wedge \mathtt{arg2}_\pi) \wedge \mathbf{F}(\mathtt{x}_{\pi_y} \wedge \mathtt{res}_\pi)$.

While $f(\varphi)$ is not necessarily in prenex normal form, it can easily be brought into prenex normal form, as there are no quantifiers under the scope of a temporal operator.

An induction shows that we have that $(\mathbb{N}, +, \cdot, <, \in) \vDash \varphi$ if and only if $f(\varphi)$ is satisfiable: Over the structure of ψ, one proves for all variable assignments α of the free variables of a subformula ψ' that $(\mathbb{N}, +, \cdot, <, \in), \alpha \vDash \psi'$ if and only if $T_\alpha, \Pi_\alpha, 0 \vDash hyp(\psi')$, where T_α is a set containing enough traces to satisfy $\theta_{all} \wedge \theta_{(+,\cdot)}$, which are additionally marked according to the values $\alpha(\mathcal{Y}_j)$ such that θ_{cons} is satisfied, and where Π_α mimics the assignment α to first- and second-order variables as explained above. Then, the remaining existential third-order quantifiers are mimicked by the choice of T, as also explained above. We leave the fully formal definition and the actual inductive proof, which follows from the remarks above, to the reader.

Now, consider a language L in Σ^2_1, i.e., it is of the form $\{n \in \mathbb{N} \mid (\mathbb{N}, +, \cdot, <, \in) \vDash \varphi(n)\}$ where $\varphi(x)$ is a formula of the form $\exists \mathcal{Y}_1 \dots \exists \mathcal{Y}_k.\ \psi(x, \mathcal{Y}_1, \dots, \mathcal{Y}_k)$, with first-order variable x and third-order variables $\mathcal{Y}_j$, and with ψ only containing first-order and second-order quantifiers.

There is a function that, given $n \in \mathbb{N}$ returns a first-order formula $\theta_{=n}(x)$ with a single free first-order variable x that is only satisfied in $(\mathbb{N}, +, \cdot, <, \in)$ if x is assigned n. This function can be implemented in polynomial time in $\log n$. So, we have $n \in L$ if and only if $f(\exists x.\ \theta_{=n}(x) \wedge \varphi(x))$ is satisfiable. Thus, we have completed the desired reduction: HyperQPTL satisfiability is indeed Σ^2_1-hard. □

Now, we consider the upper bound. Here, we use the fact that traces can be encoded as sets of natural numbers via an encoding that is "implementable" in arithmetic. This allows us to encode the satisfiability problem in arithmetic.

Lemma 2. HyperQPTL *satisfiability is in* Σ^2_1.

Proof. Formulas φ of HyperQPTL can be encoded via natural numbers $enc(\varphi)$ in a way that the encoding is implementable in first-order arithmetic (e.g., by a Gödel numbering). Relying on such an encoding, we present a formula $\theta(x)$ of arithmetic with a single free (first-order) variable x such that a HyperQPTL sentence φ is satisfiable if and only if $(\mathbb{N}, +, \cdot, <, \in) \models \theta(enc(\varphi))$. To obtain our result, θ may only use existential third-order quantification, but arbitrary second- and first-order quantification.

Intuitively, we express the existence of a nonempty set of traces and the existence of some structures that prove that this set is a model of φ, generalizing the technique used to prove Σ^1_1-membership of HyperLTL satisfiability [7]. These structures include in particular Skolem functions for the existential variables. Then, we express in arithmetic that every trace assignment (which can be encoded by a type 1 object) that is consistent with the Skolem functions satisfies the maximal quantifier-free subformula of φ. To evaluate this subformula, we implement the semantics of quantifier-free HyperQPTL in arithmetic by existentially quantifying a function that assigns to each trace assignment Π and each quantifier-free subformula ψ' of φ the truth value of ψ' with respect to Π. This function is again an existentially quantified type 2 object and its correctness can be expressed in first-order arithmetic. In the following, we introduce the necessary encoding of traces, an equivalent semantics of HyperQPTL that is more convenient for the construction here, Skolem functions, and then the construction of the formula θ.

For the remainder of the proof, let us fix a HyperQPTL sentence φ over AP. To simplify our constructions, we assume w.l.o.g. that each trace variable and each proposition is quantified at most once in φ. Also, we assume w.l.o.g. that AP and the set of trace variables appearing in φ are disjoint finite subsets of $\mathbb{N}$. These properties can be guaranteed by renaming trace variables and propositions.

Let us first explain how we encode traces as sets of natural numbers. Let $pair : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ denote Cantor's pairing function defined as $pair(i, j) = \frac{1}{2}(i + j)(i + j + 1) + j$, which is a bijection and can be implemented in arithmetic. Then, we encode a trace $t \in (2^{\mathrm{AP}})^\omega$ by the set $S_t = \{pair(i, \mathrm{p}) \mid i \in \mathbb{N} \text{ and } \mathrm{p} \in t(i)\} \subseteq \mathbb{N}$. Now, one can write a formula $\varphi_{isTrace}(Y)$ which is satisfied in $(\mathbb{N}, +, \cdot, <, \in)$ if and only if the interpretation of Y encodes a trace over AP [8].

Recall that our plan is to existentially quantify a model (a set of traces), which is encoded by a set of sets of natural numbers, i.e., a type 2 object. Then, we want to evaluate the formula over that model using arithmetic. Furthermore, the semantics of HyperQPTL as defined in Subsection 3.1 updates the model T every time a proposition is quantified, i.e., from T to $T[\mathrm{q} \mapsto t_\mathrm{q}]$ for some trace $t_\mathrm{q} \in (2^{\mathrm{q}})^\omega$. To capture this update naively, in case q is universally quantified, requires universal quantification of a type 2 object that captures the encoding of $T[\mathrm{q} \mapsto t_\mathrm{q}]$. However, this would not yield the desired Σ^2_1 upper bound.

However, we do not need to update the model T as long as we keep track of t_q, as all traces in T have the same truth values w.r.t. q, i.e., those of t_q. Hence, we keep track of the truth values assigned to quantified propositions using an assignment mapping propositions q to sequences in $(2^{\{\mathrm{q}\}})^\omega$. Thus, in this proof we work with two assignments, a trace assignment and a propositional assignment. To clearly distinguish them, we use the symbols Π_t and Π_p, respectively.

Next, we introduce the modified semantics. For a position i and T, Π_t, and Π_p as above, we define

- $T, \Pi_t, \Pi_p, i \models \mathrm{p}_\pi$ if p is in the domain of Π_p and $\mathrm{p} \in \Pi_p(\mathrm{p})(i)$, or if p is not in the domain of Π_p and $\mathrm{p} \in \Pi_t(\pi)(i)$,
- $T, \Pi_t, \Pi_p, i \models \mathrm{q}$ if $\mathrm{q} \in \Pi_p(\mathrm{q})(i)$,
- $T, \Pi_t, \Pi_p, i \models \neg\psi'$ if $T, \Pi_t, \Pi_p, i \not\models \psi'$,
- $T, \Pi_t, \Pi_p, i \models \psi'_1 \vee \psi'_2$ if $T, \Pi_t, \Pi_p, i \models \psi'_1$ or $T, \Pi_t, \Pi_p, i \models \psi'_2$,
- $T, \Pi_t, \Pi_p, i \models \mathbf{X}\psi'$ if $T, \Pi_t, \Pi_p, i + 1 \models \psi'$,
- $T, \Pi_t, \Pi_p, i \models \mathbf{F}\psi'$ if there is an $i' \geq i$ such that $T, \Pi_t, \Pi_p, i' \models \psi'$,
- $T, \Pi_t, \Pi_p, i \models \exists\pi.\ \varphi$ if there exists a trace $t \in T$ such that $T, \Pi_t[\pi \mapsto t], \Pi_p, i \models \varphi$,
- $T, \Pi_t, \Pi_p, i \models \forall\pi.\ \varphi$ if for all traces $t \in T$ we have $T, \Pi_t[\pi \mapsto t], \Pi_p, i \models \varphi$,
- $T, \Pi_t, \Pi_p, i \models \exists\mathrm{q}.\ \varphi$ if there exists a trace $t_\mathrm{q} \in (2^{\{\mathrm{q}\}})^\omega$ such that $T, \Pi_t, \Pi_p[\mathrm{q} \mapsto t_\mathrm{q}], i \models \varphi$, and
- $T, \Pi_t, \Pi_p, i \models \forall\mathrm{q}.\ \varphi$ if for all traces $t_\mathrm{q} \in (2^{\{\mathrm{q}\}})^\omega$ we have $T, \Pi_t, \Pi_p[\mathrm{q} \mapsto t_\mathrm{q}], i \models \varphi$.

We have that $T, \Pi_\emptyset, 0 \models \varphi$ (i.e., in the semantics as in Subsection 3.1) if and only if $T, \Pi_{t,\emptyset}, \Pi_{p,\emptyset}, 0 \models \varphi$ (i.e., in the new semantics) for all nonempty T and all HyperQPTL sentences φ, where $\Pi_{t,\emptyset}$ and $\Pi_{p,\emptyset}$ denote the assignments with empty domain. Also, for quantifier-free formulas ψ', $T, \Pi_t, \Pi_p, i \models \psi'$ is independent of T, i.e., we have $T, \Pi_t, \Pi_p, i \models \psi'$ if and only if $T', \Pi_t, \Pi_p, i \models \psi'$ for all T and T'. This is due to the fact that T is only referred to in the cases of trace quantification in the new semantics. This is useful later when we construct θ.

To simplify the construction of the formula $\theta(x)$, we capture existential quantification of traces and propositions by Skolem functions. This allows us replace all quantifiers by a single universal quantifier that ranges over assignments that are consistent with the Skolem functions.

Fix some nonempty set T of traces over AP. Let π be an existentially quantified trace variable in φ and let $\pi_1, \dots, \pi_k$ be the trace variables that are universally quantified before π and let $\mathrm{q}_1, \dots, \mathrm{q}_\ell$ be the propositional variables that are universally quantified before π. A T-Skolem function for π is a function mapping a k-tuple of traces in T and an ℓ-tuple of traces (the i-th one over $\{\mathrm{q}_i\}$) to a trace in T. Similarly, let q be an existentially quantified propositional variable in φ and let $\pi_1, \dots, \pi_k$ be the trace variables that are universally quantified before π and let $\mathrm{q}_1, \dots, \mathrm{q}_\ell$ be the propositional variables that are universally quantified before π. A T-Skolem function for q is a function mapping a k-tuple of traces in T and an ℓ-tuple of traces (the i-th one over $\{\mathrm{q}_i\}$) to a trace over $\{\mathrm{q}\}$.

Consider a pair (Π_t, Π_p) of a trace assignment and a propositional assignment such that all universally quantified trace variables in φ are mapped to some trace in T by Π_t and all universally quantified propositional variables q are mapped to some trace in $(2^{\{\mathrm{q}\}})^\omega$ by Π_p. We say that (Π_t, Π_p) is consistent with a T-Skolem function f for an existentially quantified trace variable π if $\Pi_t(\pi) = f(\Pi_t(\pi_1), \dots, \Pi_t(\pi_k), \Pi_p(\mathrm{q}_1), \dots, \Pi_p(\mathrm{q}_\ell))$ and it is consistent with a T-Skolem function f for an existentially quantified propositional variable q if $\Pi_t(\mathrm{q}) = f(\Pi_t(\pi_1), \dots, \Pi_t(\pi_k), \Pi_p(\mathrm{q}_1), \dots, \Pi_p(\mathrm{q}_\ell))$. In both cases, the π_j and the q_j are the trace and propositional variables universally quantified before π and q, respectively.

Recall that we have fixed a HyperQPTL sentence φ. Let ψ be the maximal quantifier-free subformula of φ. Now, we have $T \models \varphi$ (in the semantics from Subsection 3.1) if and only if there are T-Skolem functions for all existentially quantified trace variables and for all existentially quantified propositional variables such that

- for every trace assignment Π_t whose domain contains all trace variables of ψ, that maps universally quantified variables to T, and that is consistent with the Skolem functions and
- for every propositional assignment Π_p whose domain contains all propositional variables in ψ, that maps universally quantified q to traces over $\{\mathrm{q}\}$, and that is consistent with the Skolem functions,

we have $\Pi_t, \Pi_p, 0 \models \psi$. Thus, we have related the original semantics with the semantics using Skolem functions and propositional assignments.

Finally, we need to witness the satisfaction of a quantifier-free formula by assignments Π_t and Π_p. Let Ψ be the set of subformulas of ψ, which is the maximal quantifier-free subformula of φ. Now, we define the expansion $e_{\psi,\Pi_t,\Pi_p} : \Psi \times \mathbb{N} \to \{0, 1\}$ of ψ with respect to a trace assignment Π_t and a propositional assignment Π_p via

$$e_{\psi,\Pi_t,\Pi_p}(\psi', i) = \begin{cases} 1 & \text{if } \Pi_t, \Pi_p, i \models \psi', \\ 0 & \text{if } \Pi_t, \Pi_p, i \not\models \psi'. \end{cases}$$

In fact, e_{ψ,Π_t,Π_p} is completely characterized by the following consistency conditions that only depend on Π_t and Π_p:

- $e_{\psi,\Pi_t,\Pi_p}(\mathrm{p}_\pi, i) = 1$ if and only if either p is in the domain of Π_p and $\mathrm{p} \in \Pi_p(\mathrm{p})(i)$ or if p is not in the domain of Π_p and $\mathrm{p} \in \Pi_t(\pi)(i)$,
- $e_{\psi,\Pi_t,\Pi_p}(\mathrm{q}, i) = 1$ if and only if $\mathrm{q} \in \Pi_p(\mathrm{q})(i)$,
- $e_{\psi,\Pi_t,\Pi_p}(\neg\psi', i) = 1$ if and only if $e_{\psi,\Pi_t,\Pi_p}(\psi', i) = 0$,
- $e_{\psi,\Pi_t,\Pi_p}(\psi'_1 \vee \psi'_2, i) = 1$ if and only if $e_{\psi,\Pi_t,\Pi_p}(\psi'_1, i) = 1$ or $e_{\psi,\Pi_t,\Pi_p}(\psi'_2, i) = 1$,
- $e_{\psi,\Pi_t,\Pi_p}(\mathbf{X}\,\psi', i) = 1$ if and only if $e_{\psi,\Pi_t,\Pi_p}(\psi', i+1) = 1$, and
- $e_{\psi,\Pi_t,\Pi_p}(\mathbf{F}\,\psi', i) = 1$ if and only if $e_{\psi,\Pi_t,\Pi_p}(\psi', i') = 1$ for some $i' \geq i$.

Note that all these conditions are expressible using a formula of first-order arithmetic (that depends on ψ).

Now, the formula θ expresses the existence of the following third-order objects:

- A nonempty set $\mathcal{T}$ of sets of natural numbers intended to encode a set of traces over AP.
- A function $\mathcal{S} : \mathbb{N} \times (2^{\mathbb{N}})^* \to 2^{\mathbb{N}}$ to be interpreted as Skolem functions, i.e., it maps variable or proposition names (the first argument) and (encodings of) sequences of traces (the second argument) to traces.
- A function $\mathcal{E} : (2^{\mathbb{N}})^* \times \mathbb{N} \times \mathbb{N} \to \{0,1\}$ to be interpreted as expansions as follows: if the first argument A encodes a trace assignment Π_t and a propositional assignment Π_p whose domains contain exactly the variables quantified in φ, then $x, y \mapsto \mathcal{E}(A, x, y)$ encodes the expansion e_{ψ,Π_t,Π_p}, i.e., x encodes a subformula of ψ and y a position.

Note that $\mathcal{S}$ and $\mathcal{E}$ can also be encoded as sets of sets of natural numbers, using the fact that a sequence $(S_1, \cdots, S_k) \in (2^{\mathbb{N}})^*$ can be encoded by the set $\{pair(n, j) \mid n \in S_j\} \in 2^{\mathbb{N}}$, and that a function $f : \mathbb{N} \to \mathbb{N}$ can be encoded by $\{pair(n, f(n)) \mid n \in \mathbb{N}\} \subseteq \mathbb{N}$. All these encodings are implementable in arithmetic.

Then, we express the following properties using only first- and second-order quantification:

- $\mathcal{T}$ does indeed only contain encodings of traces over AP, i.e., it encodes a set T of traces (using the formula $\varphi_{isTrace}$ from above).
- The image of $\mathcal{S}$ only contains traces over T, if the first argument is a trace variable, and traces over $\{\mathrm{q}\}$, if the first input is a propositional variable q (using a formula $\varphi_{format}(\mathcal{S}, enc(\varphi))$ of second-order arithmetic).
- For every $A \in (2^{\mathbb{N}})^*$ encoding a trace assignment Π_t and a propositional assignment Π_p such that
  - (Π_t, Π_p) is consistent with the Skolem functions encoded by $\mathcal{S}$,
  - all universally quantified trace variables in φ are mapped by Π_t to some trace encoding of a trace in T, and
  - all universally quantified propositional variables q in φ are mapped by Π_p to an encoding of a trace over $\{\mathrm{q}\}$,

  we require that $\mathcal{E}(A, \cdot, \cdot)$ satisfies the consistency conditions (i.e., $\mathcal{E}$ indeed encodes the expansion) and that we have $\mathcal{E}(A, n_0, 0) = 1$, where n_0 is the natural number encoding ψ (using a formula $\varphi_{correct}(\mathcal{S}, enc(\varphi))$ of second-order arithmetic).

Thus, θ has the form

$$\exists \mathcal{T}.\ \exists \mathcal{S}.\ \exists \mathcal{E}.\ \exists Y.\ Y \in \mathcal{T} \wedge \forall Y.\ (Y \in \mathcal{T} \to \varphi_{isTrace}(Y)) \wedge \varphi_{format}(\mathcal{S}, enc(\varphi)) \wedge \varphi_{correct}(\mathcal{S}, enc(\varphi)).$$

We leave the tedious, but standard, construction of $\varphi_{format}(\mathcal{S}, enc(\varphi))$ and $\varphi_{correct}(\mathcal{S}, enc(\varphi))$ to the reader.

Putting all the pieces together yields that φ is satisfiable if and only if $(\mathbb{N}, +, \cdot, <, \in) \models \theta(enc(\varphi))$. □

Now, our main result on HyperQPTL satisfiability is a direct consequence of Lemma 1 and Lemma 2.

Theorem 2. HyperQPTL *satisfiability is* Σ^2_1*-complete.*

4. HyperQPTL$^+$

In this section, we introduce HyperQPTL$^+$ and then settle the complexity of its satisfiability, finite-state satisfiability, and model-checking problems.

4.1. Syntax and semantics

In HyperQPTL, quantification over a proposition q is interpreted as labeling each trace by the same sequence t_{q} of truth values for q, i.e., the assignment of truth values is uniform. However, one can also consider a non-uniform labeling by truth values for q. This results in the logic HyperQPTL$^+$.

The syntax of HyperQPTL$^+$ [12] is very similar to that of HyperQPTL, one just drops the atomic formulas of the form q, i.e., atomic propositions that are not labeled by trace variables:

$$\varphi ::= \exists \pi.\ \varphi \mid \forall \pi.\ \varphi \mid \exists \mathrm{q}.\ \varphi \mid \forall \mathrm{q}.\ \varphi \mid \psi$$
$$\psi ::= \mathrm{p}_\pi \mid \neg\psi \mid \psi \vee \psi \mid \mathbf{X}\,\psi \mid \mathbf{F}\,\psi$$

Here p and q range over AP and π ranges over $\mathcal{V}$. The semantics are also similar, we just change the definition of propositional quantification as follows:

- $T, \Pi, i \models \exists \mathrm{q}.\ \varphi$ if there exists a $T' \subseteq (2^{\mathrm{AP}})^\omega$ such that $T =_{\mathrm{AP}\setminus\{\mathrm{q}\}} T'$ and $T', \Pi, i \models \varphi$, and
- $T, \Pi, i \models \forall \mathrm{q}.\ \varphi$ if for all $T' \subseteq (2^{\mathrm{AP}})^\omega$ such that $T =_{\mathrm{AP}\setminus\{\mathrm{q}\}} T'$ we have $T', \Pi, i \models \varphi$.

It is known that model-checking HyperQPTL$^+$ is undecidable [12], but its exact complexity is open, as is the complexity of satisfiability and finite-state satisfiability.

In the following, we show that HyperQPTL$^+$ is equally expressive as Hyper^{2}LTL, which allows us to transfer the complexity results for Hyper^{2}LTL to HyperQPTL$^+$.

4.2. Second-order HyperLTL

We continue by introducing the syntax and semantics of Hyper^{2}LTL [3]. Let $\mathcal{V}_1$ be a set of first-order trace variables (i.e., ranging over traces) and $\mathcal{V}_2$ be a set of second-order trace variables (i.e., ranging over sets of traces) such that $\mathcal{V}_1 \cap \mathcal{V}_2 = \emptyset$.

The formulas of Hyper^{2}LTL are given by the grammar

$$\varphi ::= \exists X.\ \varphi \mid \forall X.\ \varphi \mid \exists \pi \in X.\ \varphi \mid \forall \pi \in X.\ \varphi \mid \psi$$
$$\psi ::= \mathrm{p}_\pi \mid \neg\psi \mid \psi \vee \psi \mid \mathbf{X}\,\psi \mid \psi\,\mathbf{U}\,\psi$$

where p ranges over AP, π ranges over $\mathcal{V}_1$, X ranges over $\mathcal{V}_2$, and $\mathbf{X}$ (next) and $\mathbf{U}$ (until) are temporal operators. Conjunction, implication, equivalence, and the temporal operators eventually and always are defined as usual.

The semantics of Hyper^{2}LTL is defined with respect to a variable assignment, i.e., a partial mapping $\Pi : \mathcal{V}_1 \cup \mathcal{V}_2 \to (2^{\mathrm{AP}})^\omega \cup 2^{(2^{\mathrm{AP}})^\omega}$ such that

- if $\Pi(\pi)$ for $\pi \in \mathcal{V}_1$ is defined, then $\Pi(\pi) \in (2^{\mathrm{AP}})^\omega$ and
- if $\Pi(X)$ for $X \in \mathcal{V}_2$ is defined, then $\Pi(X) \in 2^{(2^{\mathrm{AP}})^\omega}$.

Given a variable assignment Π, a variable $\pi \in \mathcal{V}_1$, and a trace t, we denote by $\Pi[\pi \mapsto t]$ the assignment that coincides with Π on all variables but π, which is mapped to t. Similarly, for a variable $X \in \mathcal{V}_2$, and a set T of traces, $\Pi[X \mapsto T]$ is the assignment that coincides with Π everywhere but X, which is mapped to T.

For a variable assignment Π and a position i we define

- $\Pi, i \models \mathrm{p}_\pi$ if $\mathrm{p} \in \Pi(\pi)(i)$,
- $\Pi, i \models \neg\psi$ if $\Pi, i \not\models \psi$,
- $\Pi, i \models \psi_1 \vee \psi_2$ if $\Pi, i \models \psi_1$ or $\Pi, i \models \psi_2$,
- $\Pi, i \models \mathbf{X}\,\psi$ if $\Pi, i+1 \models \psi$,

- $\Pi, i \vDash \psi_1 \mathbf{U} \psi_2$ if there is an $i' \geq i$ such that $\Pi, i' \vDash \psi_2$ and for all $i \leq i'' < i'$ we have $\Pi, i'' \vDash \psi_1$,
- $\Pi, i \vDash \exists \pi \in X.\ \varphi$ if there exists a trace $t \in \Pi(X)$ such that $\Pi[\pi \mapsto t], i \vDash \varphi$,
- $\Pi, i \vDash \forall \pi \in X.\ \varphi$ if for all traces $t \in \Pi(X)$ we have $\Pi[\pi \mapsto t], i \vDash \varphi$,
- $\Pi, i \vDash \exists X.\ \varphi$ if there exists a set $T \subseteq (2^{\mathrm{AP}})^\omega$ such that $\Pi[X \mapsto T], i \vDash \varphi$, and
- $\Pi, i \vDash \forall X.\ \varphi$ if for all sets $T \subseteq (2^{\mathrm{AP}})^\omega$ we have $\Pi[X \mapsto T], i \vDash \varphi$.

Note that these are the standard semantics where second-order quantification ranges over arbitrary sets [3], not the closed-world semantics where second-order quantification only ranges over subsets of the model [8].

We assume the existence of two distinguished second-order variables $X_a, X_d \in \mathcal{V}_2$ such that X_a refers to the set $(2^{\mathrm{AP}})^\omega$ of all traces, and X_d refers to the universe of discourse (the set of traces the formula is evaluated over). A sentence is a formula in which only the variables X_a, X_d can be free. Thus, technically, a sentence does have free variables. Alternatively, one could treat the distinguished variables with distinguished syntax so that they are not free. However, for the sake of definitional simplicity, we refrain from doing so.

We say that a nonempty set T of traces satisfies a Hyper^{2}LTL sentence φ, written $T \vDash \varphi$, if $\Pi_\emptyset[X_a \mapsto (2^{\mathrm{AP}})^\omega, X_d \mapsto T], 0 \vDash \varphi$, where $\Pi_\emptyset$ denotes the variable assignment with empty domain. In this case, we say that T is a model of φ. A transition system $\mathfrak{T}$ satisfies φ, written $\mathfrak{T} \vDash \varphi$, if $\mathrm{Tr}(\mathfrak{T}) \vDash \varphi$.

4.3. HyperQPTL$^+$ *"is"* Hyper^{2}LTL

In this subsection, we show that HyperQPTL$^+$ and Hyper^{2}LTL are equally expressive by translating Hyper^{2}LTL into HyperQPTL$^+$ and vice versa.

Lemma 3. *There is a polynomial-time computable function f mapping* Hyper^{2}LTL *sentences φ to* HyperQPTL$^+$ *sentences $f(\varphi)$ such that we have $T \vDash \varphi$ if and only if $T \vDash f(\varphi)$ for all nonempty $T \subseteq (2^{\mathrm{AP}})^\omega$.*

Proof. Let φ be a Hyper^{2}LTL sentence. We assume w.l.o.g. that each (trace and set) variable is quantified at most once in φ. Further, we require that each set variable quantified in φ is different from X_d and X_a. These properties can always be achieved by renaming variables.

Our goal is to mimic quantification over sets of traces in HyperQPTL$^+$ via (non-uniform) quantification over propositions. Let $\mathtt{p}_1, \dots, \mathtt{p}_n$ be the propositions appearing in φ. Note that the truth values of other propositions can be ignored, as satisfaction of φ does not depend on them. We existentially quantify fresh propositions $\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}$ and then express in HyperQPTL$^+$ that all traces over these fresh propositions are available to mimic set quantification. Then, we use further propositional quantification in HyperQPTL$^+$ to mimic quantification over sets X of traces by labeling the traces over $\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}$ with a bit indicating whether they are in X or not. All other quantifiers, connectives, and temporal operators of Hyper^{2}LTL can directly be mimicked in HyperQPTL$^+$.

In the following, we construct θ_{cmplt} and $f'(\varphi)$ such that

$$f(\varphi) = \exists \mathtt{p}_1^{all} \dots \exists \mathtt{p}_n^{all}.\ (\theta_{cmplt} \wedge f'(\varphi))$$

which is satisfied by a nonempty set $T \subseteq (2^{\mathrm{AP}})^\omega$ of traces if and only if there is a set $T' \subseteq (2^{\mathrm{AP}})^\omega$ of traces such that $T' \vDash \theta_{cmplt} \wedge f'(\varphi)$ and $T =_{\mathrm{AP} \setminus \{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}} T'$.

Due to the second property, each trace in T' has the form $t \hat{} t'$ where t is a trace from the $\mathrm{AP} \setminus \{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$-projection of T and t' is a trace over $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$. Also, recall that the propositions $\mathtt{p}_j^{all}$ do not appear in φ. Thus, quantification over T' mimics both quantification of traces over the propositions $\mathtt{p}_j$ relevant for evaluating φ, and quantification over $(2^{\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}})^\omega$.

As a first step, we need to express in HyperQPTL$^+$ that the $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$-projection of a set of traces contains all traces over $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$ using the formula

$$\theta_{cmplt} = \forall \mathtt{p}_1^{tmp} \dots \forall \mathtt{p}_n^{tmp}.\ \forall \pi.\ \exists \pi'. \bigwedge\nolimits_{j=1}^{n} \mathbf{G}((\mathtt{p}_j^{tmp})_\pi \leftrightarrow (\mathtt{p}_j^{all})_{\pi'})$$

with another set $\{\mathtt{p}_1^{tmp}, \dots, \mathtt{p}_n^{tmp}\}$ of fresh propositions used only in this subformula. Intuitively, it expresses that for each trace over $\{\mathtt{p}_1^{tmp}, \dots, \mathtt{p}_n^{tmp}\}$ the trace obtained by replacing each $\mathtt{p}_j^{tmp}$ by $\mathtt{p}_j^{all}$ is in each model of θ_{cmplt}, i.e., the $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$-projection of each model must indeed be the set of all traces over $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$.

Now, let us explain how we mimic the quantification over subsets of traces in HyperQPTL$^+$: Similarly to the construction for the Σ_1^2-lower bound for HyperQPTL satisfiability (see Lemma 1), we use propositional quantification over another fresh proposition to mark each trace in the $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$-projection of a model (say at the first position) to indicate whether it is in the set or not. If $X_1, \dots, X_k$ are the second-order variables quantified in φ, then we employ a marker $\mathtt{m}_j$ for each $j \in \{1, \dots, k\}$.

However, we have to ensure that there are no two different traces that have the same $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$-projection, but different values of $\mathtt{m}_j$. This can happen when quantifying $\mathtt{m}_j$, as we do not mark the $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$-projection of the model, but the model itself. To rule this out, we require that the marking is consistent using the formula

$$\theta_{cons}^j = \forall \pi.\ (\mathbf{X}\,\mathbf{G}\,\neg(\mathtt{m}_j)_\pi) \wedge \forall \pi'. \bigwedge\nolimits_{\ell=1}^{n} \Big(\mathbf{G}\,((\mathtt{p}_\ell^{all})_\pi \leftrightarrow (\mathtt{p}_\ell^{all})_{\pi'}) \Big) \rightarrow \Big((\mathtt{m}_j)_\pi \leftrightarrow (\mathtt{m}_j)_{\pi'} \Big).$$

It expresses that only the first position can be marked (but must not) and that if two traces have the same $\{\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}\}$-projection they have the same marking.

Note that traces in T are still encoded by the "original" propositions $\mathtt{p}_1, \dots, \mathtt{p}_n$ while set quantification is mimicked by traces over the fresh propositions $\mathtt{p}_1^{all}, \dots, \mathtt{p}_n^{all}$. Hence, when mimicking the binding of a trace from some X_j or the distinguished variable X_a to some π, we need to ensure that we use the fresh propositions and not the original ones. However, when mimicking the binding of a trace from X_d, we need to use the original propositions.

We do so by replacing each $(\mathtt{p}_j)_\pi$ by $(\mathtt{p}_j^{all})_\pi$, unless when binding a trace from X_d (which ranges over the set T we are evaluating φ over). Thus, we define f' as

- $f'(\exists X_j.\ \psi) = \exists \mathtt{m}_j.\ \theta_{cons}^j \wedge f'(\psi)$,
- $f'(\forall X_j.\ \psi) = \forall \mathtt{m}_j.\ \theta_{cons}^j \rightarrow f'(\psi)$,
- $f'(\exists \pi \in X_j.\ \psi) = \exists \pi.\ (\mathtt{m}_j)_\pi \wedge f'(repl_\pi(\psi))$, where $repl_\pi(\psi)$ is the formula obtained from ψ by replacing each subformula $(\mathtt{p}_j)_\pi$ by $(\mathtt{p}_j^{all})_\pi$ (note that we only replace propositions labeled by π, the variable quantified here),
- $f'(\forall \pi \in X_j.\ \psi) = \forall \pi.\ (\mathtt{m}_j)_\pi \rightarrow f'(repl_\pi(\psi))$,
- $f'(\exists \pi \in X_a.\ \psi) = \exists \pi.\ f'(repl_\pi(\psi))$,
- $f'(\forall \pi \in X_a.\ \psi) = \forall \pi.\ f'(repl_\pi(\psi))$,
- $f'(\exists \pi \in X_d.\ \psi) = \exists \pi.\ f'(\psi)$,
- $f'(\forall \pi \in X_d.\ \psi) = \forall \pi.\ f'(\psi)$,
- $f'(\neg\psi) = \neg f'(\psi)$,
- $f'(\psi_1 \vee \psi_2) = f'(\psi_1) \vee f'(\psi_2)$,
- $f'(\mathbf{X}\,\psi) = \mathbf{X}\,f'(\psi)$,
- $f'(\mathbf{F}\,\psi) = \mathbf{F}\,f'(\psi)$, and
- $f'(\mathtt{p}_\pi) = \mathtt{p}_\pi$.

While $f'(\varphi)$ is not necessarily in prenex normal form, it can easily be brought into prenex normal form, as there are no quantifiers under the scope of a temporal operator.

An induction shows that we indeed have $T \vDash \varphi$ if and only if $T \vDash f(\varphi)$ for nonempty T: Over the structure of φ, one proves that for all nonempty sets T of traces and for all variable assignments Π of the free variables of a subformula ψ that $T, \Pi, i \vDash \psi$ if and only if $T', \Pi', i \vDash f'(\psi)$, where T' is obtained from T by completely "overwriting" the propositions $\mathtt{p}_j^{all}$ and assigning the marker propositions consistently according to Π, and where Π' is the restriction of Π to trace variables. Again, we leave the tedious, but straightforward, details to the reader. □

Now, we consider the other direction.

Lemma 4. *There is a polynomial-time computable function f mapping* HyperQPTL$^+$ *sentences φ to* Hyper^{2}LTL *sentences $f(\varphi)$ such that we have $T \vDash \varphi$ if and only if $T \vDash f(\varphi)$ for all nonempty $T \subseteq (2^{\mathrm{AP}})^\omega$.*

Proof. The semantics of HyperQPTL$^+$ can directly be expressed in Hyper^{2}LTL. Intuitively, propositional quantification in HyperQPTL$^+$ "updates" the set of traces that the trace quantifiers range over (i.e., the T in $T, \Pi, i \vDash \psi$). All other quantifiers, temporal operators, and connectives in HyperQPTL$^+$ are also available in Hyper^{2}LTL.

To mimic propositional quantification in Hyper^{2}LTL, we use a dedicated second-order variable that explicitly stores the set of traces that the trace quantifiers in the HyperQPTL$^+$ formula range over. This set is initially equal to the set of traces the formula is evaluated over (i.e., the set $\Pi(X_d)$ in the setting of Hyper^{2}LTL), and is updated with each quantification over a proposition q. As this update from T to T' has to satisfy $T =_{\mathrm{AP}\setminus\{\mathrm{q}\}} T'$, we need two set variables X_0 and X_1, one for the old value and one for the new value, to be able to compare these two. As the old value is no longer needed after the update, we can then reuse the variable. To keep track of which of the two variables X_0 and X_1 is currently used to store the set of traces we are evaluating trace quantifiers over, we use a flag $b \in \{0,1\}$, i.e., X_b is the set variable storing the current value and X_{1-b} is the variable storing the old value. To enforce a correct update, we define the formula $\theta_{\mathrm{q}}(X_b, X_{1-b})$ as

$$\forall \pi \in X_b.\ \exists \pi' \in X_{1-b}.\ \bigwedge\nolimits_{\mathrm{p} \in \mathrm{AP}\setminus\{\mathrm{q}\}} \mathbf{G}(\mathrm{p}_\pi \leftrightarrow \mathrm{p}_{\pi'}) \wedge$$

$$\forall \pi \in X_{1-b}.\ \exists \pi' \in X_b.\ \bigwedge\nolimits_{\mathrm{p} \in \mathrm{AP}\setminus\{\mathrm{q}\}} \mathbf{G}(\mathrm{p}_\pi \leftrightarrow \mathrm{p}_{\pi'}),$$

which is satisfied by a variable assignment Π if and only if $\Pi(X_b) =_{\mathrm{AP}\setminus\{\mathrm{q}\}} \Pi(X_{1-b})$.

Now, we express the semantics of HyperQPTL$^+$ in Hyper^{2}LTL. Here, we use $X_0 = X_d$ and let X_1 be some second-order variable other than X_d. Then, we define

- $f'(\exists \pi.\ \psi, b) = \exists \pi \in X_b.\ f'(\psi, b)$,
- $f'(\forall \pi.\ \psi, b) = \forall \pi \in X_b.\ f'(\psi, b)$,
- $f'(\exists \mathrm{q}.\ \psi, b) = \exists X_{1-b}.\ \theta_{\mathrm{q}}(X_{1-b}, X_b) \wedge f'(\psi, 1-b)$,
- $f'(\forall \mathrm{q}.\ \psi, b) = \forall X_{1-b}.\ \theta_{\mathrm{q}}(X_{1-b}, X_b) \rightarrow f'(\psi, 1-b)$,
- $f'(\neg\psi, b) = \neg f'(\psi, b)$,
- $f'(\psi_1 \vee \psi_2, b) = f'(\psi_1, b) \vee f'(\psi_2, b)$,
- $f'(\mathbf{X}\,\psi, b) = \mathbf{X}\, f'(\psi, b)$,
- $f'(\mathbf{F}\,\psi, b) = \mathbf{F}\, f'(\psi, b)$, and
- $f'(\mathrm{p}_\pi, b) = \mathrm{p}_\pi$.

Now, define $f(\varphi) = f'(\varphi, 0)$. While $f(\varphi)$ is not necessarily in prenex normal form, it can easily be brought into prenex normal form, as there are no quantifiers under the scope of a temporal operator.

An induction shows that we indeed have $T \vDash \varphi$ if and only if $T \vDash f(\varphi)$: One shows that for all sets T of traces and all variable valuations Π of the free variables of a subformula ψ that $T, \Pi, i \vDash \psi$ if and only if $T, \Pi', i \vDash f(\psi, b)$, where Π' is obtained from Π by extending it with the appropriate assignment for the variable X_b, which must satisfy $T =_{\mathrm{AP}\setminus\{\mathrm{q}_1,\ldots,\mathrm{q}_k\}} \Pi(X_b)$, where the q_j are the free propositional variables of ψ. One last time, we leave the tedious, but straightforward, details to the reader. □

As Hyper^{2}LTL satisfiability, finite-state satisfiability, and model-checking are equivalent to truth in third-order arithmetic [8], the translations presented in Lemma 3 and Lemma 4 imply that the same is true for HyperQPTL$^+$.

Theorem 3. HyperQPTL$^+$ *satisfiability, finite-state satisfiability, and model-checking are equivalent to truth in third-order arithmetic.*

Let us remark that two second-order variables suffice to translate HyperQPTL$^+$ into Hyper^{2}LTL. Together with the converse translation presented in Lemma 3, we conclude that every Hyper^{2}LTL sentence is equivalent to one with only two second-order variables.

5. Conclusion

We settled the exact complexity of the most important verification problems for HyperQPTL and HyperQPTL$^+$. For HyperQPTL, we proved that satisfiability is Σ^2_1-complete while for HyperQPTL$^+$, we proved that satisfiability, finite-state satisfiability, and model-checking are equivalent to truth in third-order arithmetic. The latter results were obtained by showing that HyperQPTL$^+$ and second-order HyperLTL have the same expressiveness.

Data availability

No data was used for the research described in the article.

CRediT authorship contribution statement

Gaëtan Regaud: Writing – review & editing, Writing – original draft, Investigation, Formal analysis, Conceptualization; **Martin Zimmermann:** Writing – review & editing, Writing – original draft, Supervision, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Martin Zimmermann reports financial support was provided by DFF. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Gaëtan Regaud was supported by the European Union. Martin Zimmermann was supported by DIREC – Digital Research Centre Denmark. We thank the reviewers for their detailed comments, which improved the article considerably.

References

[1] M.R. Clarkson, F.B. Schneider, Hyperproperties, J. Comput. Secur. 18 (6) (2010) 1157–1210. https://doi.org/10.3233/JCS-2009-0393

[2] M.R. Clarkson, B. Finkbeiner, M. Koleini, K.K. Micinski, M.N. Rabe, C. Sánchez, Temporal logics for hyperproperties, in: M. Abadi, S. Kremer (Eds.), POST 2014, 8414 of *LNCS*, Springer, 2014, pp. 265–284. https://doi.org/10.1007/978-3-642-54792-8_15

[3] R. Beutner, B. Finkbeiner, H. Frenkel, N. Metzger, Second-Order hyperproperties, in: C. Enea, A. Lal (Eds.), CAV 2023, Part II, 13965 of *LNCS*, Springer, 2023, pp. 309–332. https://doi.org/10.1007/978-3-031-37703-7_15

[4] B. Finkbeiner, M.N. Rabe, C. Sánchez, Algorithms for model checking HyperLTL and HyperCTL*, in: D. Kroening, C.S. Pasareanu (Eds.), CAV 2015, Part I, 9206 of *LNCS*, Springer, 2015, pp. 30–48. https://doi.org/10.1007/978-3-319-21690-4_3

[5] M.N. Rabe, A temporal logic approach to information-flow control, Ph.D. thesis, Saarland University, 2016. http://scidok.sulb.uni-saarland.de/volltexte/2016/6387/.

[6] B. Finkbeiner, C. Hahn, Deciding hyperproperties, in: J. Desharnais, R. Jagadeesan (Eds.), CONCUR 2016, 59 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, pp. 13:1–13:14. https://doi.org/10.4230/LIPIcs.CONCUR.2016.13

[7] M. Fortin, L.B. Kuijer, P. Totzke, M. Zimmermann, HyperLTL satisfiability is highly undecidable, HyperCTL* is even harder, Log. Methods Comput. Sci. 21 (1) (2025) 3. https://doi.org/10.46298/LMCS-21(1:3)2025

[8] H. Frenkel, M. Zimmermann, The complexity of second-Order HyperLTL, in: J. Endrullis, S. Schmitz (Eds.), CSL 2025, 326 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025, pp. 10:1–10:23. https://doi.org/10.4230/LIPICS.CSL.2025.10

[9] G. Regaud, M. Zimmermann, The complexity of fragments of second-Order HyperLTL, (2025). arXiv:2501.19046, https://doi.org/10.48550/ARXIV.2501.19046

[10] G. Regaud, M. Zimmermann, The complexity of generalized HyperLTL with stuttering and contexts, in: G. Bacci, A. Francalanza (Eds.), GandALF 2025, 428 of *EPTCS*, Open Publishing Association, 2025, pp. 161–176. https://doi.org/10.4204/EPTCS.428.12

[11] C. Mascle, M. Zimmermann, The keys to decidable HyperLTL satisfiability: small models or very simple formulas, in: M. Fernández, A. Muscholl (Eds.), CSL 2020, 152 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, pp. 29:1–29:16. https://doi.org/10.4230/LIPIcs.CSL.2020.29

[12] B. Finkbeiner, C. Hahn, J. Hofmann, L. Tentrup, Realizing omega-regular hyperproperties, in: S.K. Lahiri, C. Wang (Eds.), CAV 2020, Part II, 12225 of *LNCS*, Springer, 2020, pp. 40–63. https://doi.org/10.1007/978-3-030-53291-8_4

[13] A.P. Sistla, Theoretical Issues in the Design and Verification of Distributed Systems, Ph.D. thesis, Harvard University, 1983.
[14] H. Rogers, Theory of Recursive Functions and Effective Computability, MIT Press, Cambridge, MA, USA, Cambridge, MA, USA, 1987.
[15] R. Kaivola, Using Automata to Characterise Fixed Point Temporal Logics, Ph.D. thesis, University of Edinburgh, 1997.