Flavors of Quantifiers in Hyperlogics

Marek Chalupa
IST Austria, Klosterneuburg, Austria

Thomas A. Henzinger
IST Austria, Klosterneuburg, Austria

Ana Oliveira da Costa
IST Austria, Klosterneuburg, Austria

Abstract

Hypertrace logic is a sorted first-order logic with separate sorts for time and execution traces. Its formulas specify hyperproperties, which are properties relating multiple traces. In this work, we extend hypertrace logic by introducing trace quantifiers that range over the set of all possible traces. In this extended logic, formulas can quantify over two kinds of trace variables: *constrained trace variables*, which range over a fixed set of traces defined by the model, and *unconstrained trace variables*, which can be assigned to any trace. In comparison, hyperlogics such as HyperLTL have only constrained trace quantifiers. We use hypertrace logic to study how different quantifier patterns affect the decidability of the satisfiability problem. We prove that hypertrace logic without constrained trace quantifiers is equivalent to monadic second-order logic of one successor (S1S), and therefore satisfiable, and that the trace-prefixed fragment (all trace quantifiers precede all time quantifiers) is equivalent to HyperQPTL. Moreover, we show that all hypertrace formulas where the only alternation between constrained trace quantifiers is from an existential to a universal quantifier are equisatisfiable to formulas without constraints on their trace variables and, therefore, decidable as well. Our framework allows us to study also time-prefixed hyperlogics, for which we provide new decidability and undecidability results.

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases Hyperproperties, Satisfiability, First-order Logic, S1S

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2025.

Acknowledgements This work was supported in part by the Austrian Science Fund (FWF) SFB project SpyCoDe 10.55776/F85 and by the ERC Advanced Grant VAMOS 101020093.

1 Introduction

Temporal logics offer a powerful formalism for specifying *trace properties*, which describe sets of allowed sequences of events (or system states) that a system can exhibit over time. Within the linear-time spectrum, Linear Temporal Logic (LTL) has been particularly successful in system verification, striking a practical balance between expressiveness and decidability. Its extension with propositional quantifiers, known as Quantified Propositional Temporal Logic (QPTL), extends LTL's expressive power to capture all regular trace properties while preserving decidability. The connection between temporal logics and classical first-order logic for specifying linear-time properties was first explored in the seminal work of Kamp [12]. This line of research established that LTL is expressively equivalent to first-order logic of order (FO[<]), where the order relation $<$ is interpreted over the natural numbers. In the case of QPTL, it was proven to be expressively equivalent to monadic second-order logic of order of one successor (S1S) [13]. However, these logics are inherently limited to reasoning about individual execution traces and cannot express properties that involve comparing multiple executions. In particular, they are not capable of capturing *hyperproperties*.

To address this limitation of trace-based formalisms in expressing hyperproperties, nu-

merous extensions have been proposed in the literature. Notably, HyperLTL and HyperQPTL extend LTL and QPTL, respectively, with quantification over traces of the system under verification. An alternative approach involves extending FO[<] with capabilities to compare multiple traces. *Hypertrace logic*, introduced in [1], extends FO[<] into a two-sorted first-order logic, with separate sorts for time and traces of a given system, allowing only binary predicates over traces and time.

In this work, we propose an extension of hypertrace logic that includes both *constrained trace quantifiers*, ranging over all traces in the set given as a model, and *unconstrained trace quantifiers*, ranging over the universe of all traces. Properties with a mix of constrained and unconstrained trace quantification occur naturally in many areas of system verification. For example, in reactive systems, *input enableness* requires that *"the system must produce an output for any possible input"* which can be specified as:

$$\forall \pi \; \exists \pi' :: T \; \forall i \; \big((\mathrm{input}(\pi, i) \leftrightarrow \mathrm{input}(\pi', i)) \wedge \mathrm{outputs}(\pi', i) \big). \tag{1}$$

By using the unconstrained quantifier, $\forall \pi$, we ensure that all possible input values are taken into account, not just those accepted by the reactive system. By linking the universally quantified inputs to those instantiated by the constrained existential quantifier, $\exists \pi' :: T$, we guarantee that every possible input is in at least one system execution.

Another example, can be found in the specification of consistency properties in concurrent systems like *linearizability* [11], which requires all histories of invocations and response events to shared resources to be consistent with some sequential execution of those operations. While verifying for linearizability, it is irrelevant whether the implementation of a concurrent data structure exhibits linear histories because all that matters is that its histories can be successfully related to the "idealized" sequential implementation. In its essence, linearizability can be captured by the following hyperproperty:

$$\forall \pi :: T \; \exists \pi' \; (\varphi_{\mathrm{linear}}(\pi') \wedge \varphi_{\mathrm{equivOrder}}(\pi, \pi')). \tag{2}$$

The formula above universally quantifies over the observed call history of the shared object while requiring the existence of a linear trace ($\varphi_{\mathrm{linear}}(\pi')$) that is equivalent to the observed one and respects the precedence order defined between the observed ($\varphi_{\mathrm{equivOrder}}(\pi, \pi')$).

We prove that the fragment of hypertrace logic without constrained trace quantifiers, named *unconstrained hypertrace logic*, is expressively equivalent to S1S. It follows directly that unconstrained hypertrace logic has a decidable satisfiability problem. We show how to translate hypertrace formulas whose only constrained trace quantifier alternation is from existential to universal quantification into an equisatisfiable unconstrained hypertrace formula. Hence, this fragment also has a decidable satisfiability problem. This fragment places no limitations on the positioning of unconstrained or temporal quantifiers after the constrained quantification. For the fragment where all trace quantifiers occur before time quantifiers, we prove it to be expressively equivalent to HyperQPTL. While the majority of satisfiability results for hyperlogics in the literature focus on the trace-prefix fragment, we extend this line of work by also analyzing the fragment of hypertrace logic where time quantifiers occur first. For the time-prefix fragment, we prove a new undecidability result by presenting a reduction from the non-halting problem for 2-counter Minsky machines. Our main contribution is in highlighting the role of different quantifiers, including unconstrained trace quantification, plays in defining hyperlogics with a decidable satisfiability problem.

2 Logics of Time and Order

In this section, we present the necessary theoretical background, including definitions and prior results, which will form the basis for the results introduced in the next sections.

Let $\mathcal{X}$ be a finite set of boolean variables. A *valuation* is a partial mapping assigning boolean values, $\mathbf{2}=\{0,1\}$, to variables in $\mathcal{X}$, that is, $v:\mathcal{X}\to\mathbf{2}$. We denote by $v[x\mapsto b]$ the valuation resulting from updating the value of x in v to b, and the *set of all valuations of* $\mathcal{X}$ by $\mathbf{2}^{\mathcal{X}}$. We freely treat a valuation as a set of variables where suitable. In particular, we write $x\in v$ when $v(x)=1$. A *trace* τ *over variables* $\mathcal{X}$ is a sequence of valuations in $\mathbf{2}^{\mathcal{X}}$. The set of all *finite* traces over $\mathcal{X}$ is denoted $(\mathbf{2}^{\mathcal{X}})^*$, while the set of all *infinite* traces over $\mathcal{X}$ is denoted by $(\mathbf{2}^{\mathcal{X}})^\omega$. For a trace $\tau=v_0v_1\ldots$ and an index i within its length (i.e., $i<|\tau|$), we adopt the following indexing notations: $\tau[i]=v_i$, $\tau[i\ldots]=v_iv_{i+1}\ldots$, and $\tau[\ldots i]=v_0v_1\ldots v_{i-1}$. When an index falls outside a trace length (i.e., $j\geq|\tau|$), we adopt the convention that $\tau[j\ldots]$ is the empty trace, and $\tau[\ldots j]=\tau$. Two traces, τ and τ', agree on their valuation of a set of propositional variables $\mathcal{Y}$ at position i, denoted $\tau[i]=_{\mathcal{Y}}\tau'[i]$, iff $(\tau[i]\cap\mathcal{Y})=(\tau'[i]\cap\mathcal{Y})$. We generalize it to equality between two traces of equal size with respect to a given set of variables $\mathcal{Y}$, denoted $\tau=_{\mathcal{Y}}\tau'$, when, for all $i<|\tau|$, $\tau[i]=_{\mathcal{Y}}\tau'[i]$.

Trace properties define set of traces satisfying a target specification. Formally, a trace property T over $\mathcal{X}$ is a set of traces over $\mathcal{X}$; for infinite traces, that is, $\mathrm{T}\subseteq(\mathbf{2}^{\mathcal{X}})^\omega$. A trace τ satisfies a trace property T iff it is one of its elements, that is $\tau\in\mathrm{T}$. We refer to the set of all trace properties as $\mathbb{T}=\mathbf{2}^{(\mathbf{2}^{\mathcal{X}})^\omega}$. For systems represented by a set of traces S, where each trace corresponds to one of the system's execution, the system is said to satisfy the trace property T if and only if all its traces are in T; that is, $S\subseteq\mathrm{T}$. A *hyperproperty* $\mathbf{T}$ defines a set of systems (or, equivalently a set of trace properties) satisfying a given specification, $\mathbf{T}\subseteq\mathbb{T}$. Hence, a system S satisfies a hyperproperty $\mathbf{T}$ if and only if $S\in\mathbf{T}$.

2.1 Linear Temporal Logics

A successful formalism to specify trace properties is Linear Temporal Logic (LTL), introduced by Pnueli in [16]. LTL formulas are defined by the grammar: $\varphi ::= a\mid\neg\varphi\mid\varphi\vee\varphi\mid\mathbf{X}\varphi\mid\varphi\,\mathbf{U}\,\varphi$ where $a\in\mathcal{X}$ is a propositional variable, and $\mathbf{X}$ ("next") and $\mathbf{U}$ ("until") are temporal modalities. Given a LTL formula φ and an infinite trace $\tau\in(\mathbf{2}^{\mathcal{X}})^\omega$, the trace τ satisfies φ, denoted $\tau\models_{\mathtt{LTL}}\varphi$, inductively over the structure of φ, as follows:

$\tau\models_{\mathtt{LTL}}a$ iff $a\in\tau[0]$ $\qquad$ $\tau\models_{\mathtt{LTL}}\neg\psi$ iff $\tau\not\models_{\mathtt{LTL}}\psi$

$\tau\models_{\mathtt{LTL}}\psi_1\vee\psi_2$ iff $\tau\models_{\mathtt{LTL}}\psi_1$ or $\tau\models_{\mathtt{LTL}}\psi_2$ $\qquad$ $\tau\models_{\mathtt{LTL}}\mathbf{X}\psi$ iff $\tau[1\ldots]\models_{\mathtt{LTL}}\psi$

$\tau\models_{\mathtt{LTL}}\psi_1\,\mathbf{U}\,\psi_2$ iff exists $j\geq0$ s.t. $\tau[j\ldots]\models_{\mathtt{LTL}}\psi_2$ and for all $0\leq j'<j,\tau[j'\ldots]\models_{\mathtt{LTL}}\psi_1$.

QPTL extends LTL with propositional quantification. Concretely, QPTL formulas may include subformulas with propositional quantifiers (e.g., $\exists q\ \varphi$, where $q\in\mathcal{X}$ and φ is a QPTL formula) interpreted as: $\tau\models_{\mathtt{QPTL}}\exists q\ \psi$ iff exists τ' s.t. $\tau=_{\mathcal{X}\setminus\{q\}}\tau'$ and $\tau'\models_{\mathtt{QPTL}}\psi$. All other subformulas are interpreted as for LTL.

2.2 Monadic Logics of Order

Monadic second-order logic of one successor (S1S) is a widely used formalism for reasoning about regular properties of infinite sequences. S1S formulas φ are defined by the grammar:

$$\varphi ::= \exists X\,\varphi\mid\exists i\,\varphi\mid\neg\varphi\mid\varphi\vee\varphi\mid i=i\mid\mathrm{Succ}(i,i)\mid X(i)\tag{3}$$

where X is a second-order variable from a set of variables $\mathcal{V}_2$, i is a first-order variable over a set of variables $\mathcal{V}_1$ and Succ is a successor function. The set of first and second order variables are disjoint; that is, $\mathcal{V}_1\cap\mathcal{V}_2=\emptyset$. Unless specified otherwise, we use uppercase letters for second-order variables, $\mathcal{V}_2=\{X,Y,\ldots\}$, and lowercase letter for first-order variables, $\mathcal{V}_1=\{i,j,\ldots\}$. The set of free variables of a formula (i.e., not bounded to a quantifier) and closed formulas (no free variables) are defined as usual.

We interpret S1S formulas over the natural numbers, where the $\text{Succ}(n, n')$ is interpreted as usual: $\text{Succ}(n, n')$ iff $n' = n + 1$. We adopt the following abbreviations:

- $X \subseteq Y \stackrel{\text{def}}{=} \forall i\ (X(i) \rightarrow Y(i))$;
- $\text{SuccClosed}(X) \stackrel{\text{def}}{=} \forall i\ \forall i'\ ((X(i) \wedge \text{Succ}(i, i')) \rightarrow X(i'))$;
- $x \leq y \stackrel{\text{def}}{=} \forall Z((Z(x) \wedge \text{SuccClosed}(Z)) \rightarrow Z(y))$;
- $x < y \stackrel{\text{def}}{=} x \leq y \wedge \neg(x = y)$;
- $i = 0 \stackrel{\text{def}}{=} \forall j\ (i = j \vee i < j)$.

The semantics of S1S formulas is defined inductively with respect to two assignments: one for first-order variables, $\Pi_1 : \mathcal{V}_1 \mapsto \mathbb{N}$, and another for second-order variables, $\Pi_2 : \mathcal{V}_2 \mapsto \mathbf{2}^{\mathbb{N}}$. For any assignment Π, $\Pi[x \mapsto v]$ denotes the assignment that is the same as Π except for the value of x that is updated to v. We define that (Π_1, Π_2) satisfies φ inductively, as follows:

$(\Pi_1, \Pi_2) \models_{\text{S1S}} \exists X\ \varphi$ iff exists $S \subseteq \mathbb{N}$ s.t. $(\Pi_1, \Pi_2[X \mapsto S]) \models_{\text{S1S}} \varphi$

$(\Pi_1, \Pi_2) \models_{\text{S1S}} \exists i\ \varphi$ iff exists $k \in \mathbb{N}$ s.t. $(\Pi_1[i \mapsto k], \Pi_2) \models_{\text{S1S}} \varphi$

$(\Pi_1, \Pi_2) \models_{\text{S1S}} \neg\varphi$ iff $(\Pi_1, \Pi_2) \not\models_{\text{S1S}} \varphi$

$(\Pi_1, \Pi_2) \models_{\text{S1S}} \varphi \vee \varphi'$ iff $(\Pi_1, \Pi_2) \models_{\text{S1S}} \varphi$ or $(\Pi_1, \Pi_2) \models_{\text{S1S}} \varphi'$

$(\Pi_1, \Pi_2) \models_{\text{S1S}} i = j$ iff $\Pi_1(i) = \Pi_1(j)$

$(\Pi_1, \Pi_2) \models_{\text{S1S}} \text{Succ}(i, i')$ iff $\Pi_1(i') = \Pi_1(i) + 1$

$(\Pi_1, \Pi_2) \models_{\text{S1S}} X(i)$ iff $\Pi_1(i) \in \Pi_2(X)$

S1S formulas define a set with all their satisfying assignments: $[\![\varphi]\!] = \{(\Pi_1, \Pi_2) \mid (\Pi_1, \Pi_2) \models_{\text{S1S}} \varphi\}$.

▶ **Theorem 1** ([19]). *For all S1S formulas φ, it is decidable to determine whether $[\![\varphi]\!] \neq \emptyset$.*

In his seminal work [12], Kamp studied of relative expressiveness between the classical first-order approach to specify linear-time – the first-order logic of order, FO[<] – and LTL. The first-order logic FO[<] is interpreted over labeled linear orders with all uninterpreted predicates being unary. Formally, FO[<] formulas φ are defined by the grammar:

$$\varphi ::= \exists i\, \varphi \mid \neg\varphi \mid \varphi \vee \varphi \mid i < i \mid i = i \mid X(i) \tag{4}$$

where i is a first-order variable, $=$ is equality, $<$ is the order, and X is predicate from a given set of monadic predicates $\mathcal{X}$.

We work with interpretations of FO[<] over labeled linear-orders defined by a tuple $(\Lambda, <, \mathcal{I})$ where $<$ defines a linear order over the domain Λ, and $\mathcal{I}$ is a function that associates to each predicate symbol a subset of elements of Λ. From a trace τ, defined over the set of propositional variables $\mathcal{X}$, we can derive the labeled linear-order $(\mathbb{N}, <, \mathcal{I}_\tau)$ over the set of unary predicates $\{X_a \mid a \in \mathcal{X}\}$, where $<$ is interpreted as usual for natural numbers and $\mathcal{I}_\tau(X_a) = \{j \mid a \in \tau[j]\}$. Using this interpretation, it is straightforward to translate LTL formulas to equivalent FO[<] formulas interpreted over $(\mathbb{N}, <)$. Hence, FO[<] subsumes LTL. The only question remaining is whether LTL subsumes FO[<] interpreted over $(\mathbb{N}, <)$. Kamp proved in [12] that LTL with both past and future temporal operators is equivalent to FO[<] over Dedekind complete orders. Later, Gabbay et al. [9] proved that when considering only future operators, LTL is complete for FO[<] under the interpretation of $<$ over the natural numbers. For the remaining of the manuscript, we are only interested in the linear order over natural numbers.

▶ **Theorem 2** ([12, 9]). LTL *and FO[<] interpreted over $(\mathbb{N}, <)$ are equally expressive.*

3 Hypertrace Logic

In this section, we extend hypertrace logic, introduced in [1], to support *quantification over unconstrained trace variables*; that is, variables ranging over the set of all possible traces.

3.1 Syntax and Semantics

Hypertrace logic [1], denoted FO[$<, \mathbb{T}$], extends FO[$<$] with a time sort $\mathbb{N}_<$ and a trace sort $\mathbb{T}$. While FO[$<$] allows only unary uninterpreted predicates[1], in FO[$<, \mathbb{T}$] we allow binary uninterpreted predicates defined over pairs of a trace and a time variable. In [1], hypertrace logic is defined only over *constrained trace variables*: trace variables ranging over a fixed set of traces. Here we lift the implicit constraints on trace variables: we quantify instead over the set of all possible traces while enabling trace variables to be constrained by a unary predicate. Formally, in this work, hypertrace formulas φ are defined by the grammar:

$$\varphi ::= \exists \pi \ \varphi \mid \exists \pi{::}T \ \varphi \mid \exists i \ \varphi \mid \neg\varphi \mid \varphi \vee \varphi \mid i < i \mid i = i \mid X(\pi, i) \tag{5}$$

where π is a trace variable from a set $\mathcal{V}_\mathbb{T}$, i is a time variable from a set $\mathcal{V}_\mathbb{N}$ disjoint from $\mathcal{V}_\mathbb{T}$, T is a unary predicate over trace variables and X is a binary predicate over pairs of trace and time variables from a finite set $\mathbb{X}$. We typically use lowercase letters for predicates in $\mathbb{X}$ to reflect their correspondence with propositional variables in traces. Without loss of generality, we assume that all variables are quantified only once. We refer to $\exists \pi{::}T$ as a *(existential) constrained trace quantifier* and we may refer to π as a *constrained trace variable*. We introduce the usual abbreviations: $\forall x \ \varphi \stackrel{\text{def}}{=} \neg(\exists x \ \neg\varphi)$, where $x \in \{i, \pi\}$; $\forall \pi{::}T \varphi \stackrel{\text{def}}{=} \neg(\exists \pi{::}T \ \neg\varphi)$; and $\varphi \wedge \varphi' \stackrel{\text{def}}{=} \neg(\neg\varphi \vee \neg\varphi')$. Finally, constrained trace quantifiers are abbreviations for the following first-order formulas: $\exists \pi{::}T \ \varphi \stackrel{\text{def}}{=} \exists \pi \ (T(\pi) \wedge \varphi)$ and $\forall \pi{::}T \ \varphi \stackrel{\text{def}}{=} \forall \pi \ (T(\pi) \rightarrow \varphi)$.

▶ **Example 3.** We define independence between a secret input and a public output as: $\varphi \stackrel{\text{def}}{=} \forall \pi{::}T \forall \pi'{::}T \exists \pi_\exists{::}T \ \forall i \ ((\text{secret}(\pi, i) \leftrightarrow \text{secret}(\pi_\exists, i)) \wedge (\text{pub}(\pi', i) \leftrightarrow \text{pub}(\pi_\exists, i)))$. We can mix constrained and unconstrained quantifiers to require that a system produces all combinations of visible public outputs with possible input secret values (i.e., a combination of input enableness with independence between secret inputs and public outputs): $\varphi_{\text{mix}} \stackrel{\text{def}}{=} \forall \pi \ \forall \pi'{::}T \ \exists \pi_\exists{::}T \ \forall i \ ((\text{secret}(\pi, i) \leftrightarrow \text{secret}(\pi_\exists, i)) \wedge (\text{pub}(\pi', i) \leftrightarrow \text{pub}(\pi_\exists, i)))$.

Our models of interest are sets of traces. As FO[$<, \mathbb{T}$] is a first-order logic, we start by defining how to translate a set of traces to a first-order structure. We translate each propositional variable $a \in \mathcal{X}$ to the binary predicate $X_a(\tau, k)$, asserting that "variable a is true in trace $\tau \in (\mathbf{2}^\mathcal{X})^\omega$ at time $k \in \mathbb{N}$". Formally, given a set T of traces, we translate T to a structure $\overline{\text{T}}$ with signature: $(\mathbb{N}, (\mathbf{2}^\mathcal{X})^\omega; \ < \subseteq \mathbb{N} \times \mathbb{N}, T \subseteq (\mathbf{2}^\mathcal{X})^\omega, (X_a \subseteq (\mathbf{2}^\mathcal{X})^\omega \times \mathbb{N})_{a \in \mathcal{X}})$ where $\mathbb{N}$ and $(\mathbf{2}^\mathcal{X})^\omega$ are the time and trace sort domains, respectively. The predicate $<$ is interpreted as the usual order on natural numbers, the unary predicate T is true for all traces in the set set of traces used to generated the structure[2], while for all variables $a \in \mathcal{X}$: $X_a = \{(\tau, k) \mid a \in \tau[k]\}$.

We evaluate hypertrace formulas over pairs of assignments for traces and time: $(\Pi_\mathbb{T}, \Pi_\mathbb{N}) : (\mathcal{V}_\mathbb{T} \rightarrow (\mathbf{2}^\mathcal{X})^\omega) \times (\mathcal{V}_\mathbb{N} \rightarrow \mathbb{N})$. A set T of traces is a model of a hypertrace formula $\varphi \in \text{FO}[<, \mathbb{T}]$, denoted $\text{T} \models_\mathbb{T} \varphi$, iff $\overline{\text{T}}$ models φ under the standard first-order semantics. Formally, $\text{T} \models_\mathbb{T} \varphi$,

[1] The binary predicates for equality, $=$, and the linear order, $<$, are interpreted by $(\mathbb{N}, <)$.

[2] We slightly abuse notation by letting T denote both the set of traces generating the structure and the predicate determining which traces constrain the trace quantification. Note that, the set used to define the structure may be renamed, but the predicate T is true only for the traces in that set.

iff there exists a pair of assignments $(\Pi_\mathbb{T}, \Pi_\mathbb{N})$ such that $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models \varphi$, where $\models$ is the standard first-order logic models relation. In particular, trace quantifiers are interpreted as:

$(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \exists \pi\ \varphi$ iff exists $\tau \in (\mathbf{2}^\mathcal{X})^\omega$ s.t. $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}[\pi \mapsto \tau], \Pi_\mathbb{N})) \models_\mathbb{T} \varphi$

$(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \exists \pi :: T\ \varphi$ iff exists $\tau \in \mathrm{T}$ s.t. $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}[\pi \mapsto \tau], \Pi_\mathbb{N})) \models_\mathbb{T} \varphi$.

Hypertrace formulas define a set with all the set of traces it satisfies: $[\![\varphi]\!] = \{ \mathrm{T} \mid \mathrm{T} \models_\mathbb{T} \varphi \}$. From now on, we may refer to X_a just as a, and omit the subscript $\mathbb{T}$ in $\models_\mathbb{T}$, when clear.

▶ **Example 4.** Looking back to our previous example, consider the sets of traces defined over the variables secret and pub, where with each valuation v is represented as a set: $\mathrm{T}_0 = \{\{\}^\omega\}$, $\mathrm{T}_1 = \{\{\mathrm{secret}, \mathrm{pub}\}^\omega\}$, and $\mathrm{T}_{0,1} = \{\{\}^\omega, \{\mathrm{secret}, \mathrm{pub}\}^\omega\}$. For the fully constrained formula in Example 3, $\mathrm{T}_0 \models_\mathbb{T} \varphi$ and $\mathrm{T}_1 \models_\mathbb{T} \varphi$ because in each of these traces we only observe one of the possible values for secret. While, $\mathrm{T}_{0,1} \not\models_\mathbb{T} \varphi$. For the mix formula, φ_{mix} none of sets satisfy its requirements; that is, $\mathrm{T}_0 \not\models_\mathbb{T} \varphi_{\mathrm{mix}}$, $\mathrm{T}_1 \not\models_\mathbb{T} \varphi_{\mathrm{mix}}$ and $\mathrm{T}_{0,1} \not\models_\mathbb{T} \varphi_{\mathrm{mix}}$.

We close this section by defining the function `flatten` that rewrites hypertrace formulas into an equisatisfiable formula, exploiting the independence between variable valuations in traces assigned to unconstrained trace variables. For each trace variable in a set $\mathcal{V}$ and propositional variable, π and $x \in \mathcal{X}$, `flatten` introduces a new trace variable, π_x, which is used exclusively in predicates involving the corresponding variables (e.g., $x(\pi, i)$). For a given set of trace variables $\mathcal{V}$ and propositional variables $\mathcal{X}$, we define $\mathcal{V}_\mathcal{X} = \{\pi_x \mid \pi \in \mathcal{V} \text{ and } x \in \mathcal{X}\}$.

$$
\begin{aligned}
\texttt{flatten}(\exists \pi\ \varphi, \{x_0, \ldots, x_n\}, \mathcal{V}^c) &= \exists \pi_{x_0} \ldots \exists \pi_{x_n} \texttt{flatten}(\varphi, \{x_0, \ldots, x_n\}, \mathcal{V}^c \cup \{\pi\}) \\
\texttt{flatten}(\exists \pi :: T\ \varphi, \mathcal{X}, \mathcal{V}^c) &= \exists \pi :: T\ \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}^c) \\
\texttt{flatten}(\exists i\ \varphi, \mathcal{X}, \mathcal{V}^c) &= \exists i\ \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}^c) \\
\texttt{flatten}(x(\pi, i), \mathcal{X}, \mathcal{V}^c) &= \begin{cases} x(\pi_x, i) & \text{if } \pi \in \mathcal{V}^c \\ x(\pi, i) & \text{otherwise} \end{cases} \\
\texttt{flatten}(\neg \varphi, \mathcal{X}, \mathcal{V}^c) &= \neg \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}^c) \\
\texttt{flatten}(\varphi \vee \varphi', \mathcal{X}, \mathcal{V}^c) &= \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}^c) \vee \texttt{flatten}(\varphi', \mathcal{X}, \mathcal{V}^c)
\end{aligned}
\tag{6}
$$

We prove below that `flatten` returns an equisatisfiable hypertrace formula.

▶ **Lemma 5.** *Let φ be a hypertrace formula over the set of propositional variables $\mathcal{X}$ and T be a set of traces over the same variables. Let $\Pi_\mathbb{N}$ and $\Pi_\mathbb{T}$ be trace and time assignments over the set of free variables in φ. Let $\mathcal{V}^c \subseteq free(\varphi)$ be a set of trace variables that are free in φ. For all trace assignments $\Pi'_\mathbb{T}$ over trace variables in $\mathcal{V}^c_\mathcal{X} \cup free(\varphi)$ agreeing with $\Pi_\mathbb{T}$ in its assignments of propositional variables x for the trace assigned to π (i.e., for all $\pi \in \mathcal{V}^c$ and $x \in \mathcal{X}$, $\Pi_\mathbb{T}(\pi) =_{\{x\}} \Pi'_\mathbb{T}(\pi_x)$) and otherwise having the same trace assignments of $\Pi_\mathbb{T}$ (i.e., for all $\pi \notin \mathcal{V}^c$, $\Pi_\mathbb{T}(\pi) = \Pi'_\mathbb{T}(\pi)$): $(\overline{T}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \varphi$ iff $(\overline{T}, (\Pi'_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}^c)$.*

Proof. Let $\mathcal{X} = \{x_0, \ldots, x_n\}$. We prove the statement by structural induction on the formula.

Let us prove the $\Rightarrow$ direction. In the base case $(\varphi = x(\pi, i))$, $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} x(\pi, i)$ iff $(\Pi_\mathbb{T}(\pi), \Pi_\mathbb{N}(i)) \in X_x$. By definition of X_x (given by $\overline{\mathrm{T}}$), it holds that $(\Pi_\mathbb{T}(\pi), \Pi_\mathbb{N}(i)) \in X_x \Leftrightarrow \Pi_\mathbb{T}(\pi)[\Pi_\mathbb{N}(i)](x) = \mathit{true}$. If $\pi \in \mathcal{V}^c$, this is equivalent to $\Pi'_\mathbb{T}(\pi_x)[\Pi_\mathbb{N}(i)](x) = \mathit{true}$ (by definition of $\Pi'_\mathbb{T}$). Otherwise, it is equivalent to $\Pi'_\mathbb{T}(\pi)[\Pi_\mathbb{N}(i)](x) = \mathit{true}$ (also by definition of $\Pi'_\mathbb{T}$). Therefore, $(\overline{\mathrm{T}}, (\Pi'_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \texttt{flatten}(x(\pi, i), \mathcal{X}, \mathcal{V}^c)$. If φ is $i = j$ or $i < j$, the implication holds as these formulas depend only on $\Pi_\mathbb{N}$ which is not affected by changes from `flatten`.

In the induction step, cases for $\varphi_1 \vee \varphi_2$ and $\neg\varphi$ follow from the definition of $\models_\mathbb{T}$ and induction hypothesis. Assume the formula $\exists i\ \varphi$. Then, $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \exists i\ \varphi \Leftrightarrow \exists c \in \mathbb{N}$

s.t. $(\overline{\mathrm{T}}, (\Pi_{\mathbb{T}}, \Pi_{\mathbb{N}}[i \mapsto c])) \models_{\mathbb{T}} \varphi$. From IH, $(\overline{\mathrm{T}}, (\Pi'_{\mathbb{T}}, \Pi_{\mathbb{N}}[i \mapsto c])) \models_{\mathbb{T}} \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}^c)$ which implies $(\overline{\mathrm{T}}, (\Pi'_{\mathbb{T}}, \Pi_{\mathbb{N}})) \models_{\mathbb{T}} \texttt{flatten}(\exists i\ \varphi, \mathcal{X}, \mathcal{V}^c)$. Analogously for $\exists \pi{::}T$.

Finally, assume formula $\exists \pi\ \varphi$ and a set of trace variables $\mathcal{V}^c \subseteq \text{free}(\exists \pi\ \varphi)$. This formula is satisfied iff there exists t s.t. $(\overline{\mathrm{T}}, (\Pi_{\mathbb{T}}[\pi \mapsto t], \Pi_{\mathbb{N}})) \models_{\mathbb{T}} \varphi$. We observe that $\pi \in \text{free}(\varphi)$, as we assume there are no double binding of variables. Then, from IH, for $\mathcal{V}^{c\prime} = \mathcal{V}^c \cup \{\pi\}$, it follows that $(\overline{\mathrm{T}}, (\widetilde{\Pi_{\mathbb{T}}}, \Pi_{\mathbb{N}})) \models_{\mathbb{T}} \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}^{c\prime})$ where $\widetilde{\Pi_{\mathbb{T}}}$ is such that for all $\pi' \in \mathcal{V}^c \cup \{\pi\}$ and $x \in \mathcal{X}$, $\Pi_{\mathbb{T}}[\pi \mapsto t](\pi') =_{\{x\}} \widetilde{\Pi_{\mathbb{T}}}(\pi_x)$ and otherwise the trace assignments are the same as $\Pi_{\mathbb{T}}$. Observe that $(\overline{\mathrm{T}}, (\Pi'_{\mathbb{T}}[\pi_{x_0} \mapsto \widetilde{\Pi_{\mathbb{T}}}(\pi_{x_0}), ..., \pi_{x_n} \mapsto \widetilde{\Pi_{\mathbb{T}}}(\pi_{x_n})], \Pi_{\mathbb{N}})) \models_{\mathbb{T}} \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V} \cup \{\pi\})$. From the definition of $\models_{\mathbb{T}}$: $(\overline{\mathrm{T}}, (\Pi'_{\mathbb{T}}, \Pi_{\mathbb{N}})) \models_{\mathbb{T}} \exists \pi_{x_0} ... \exists \pi_{x_n} \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}^c)$.

The proof for the $\Leftarrow$ direction is analogous to the $\Rightarrow$ direction, because the relation between $\Pi_{\mathbb{T}}$ and $\Pi'_{\mathbb{T}}$ is equational. ◀

It follows from the above lemma that all hypertrace formulas are equisatisfiable to their rewrite with `flatten`.

▶ **Corollary 6.** *Let φ be a closed hypertrace formula over the set of propositional variables $\mathcal{X}$: $[\![\varphi]\!] \neq \emptyset$ iff $[\![\texttt{flatten}(\varphi, \mathcal{X}, \emptyset)]\!] \neq \emptyset$.*

3.2 Satisfiability of Hypertrace Formulas

We are interested in the decidability of the satisfiability problem of hypertrace logic.

Satisfiability of Hypertrace Formulas

Let φ be a hypertrace formula over trace variables $\mathcal{V}_{\mathbb{T}}$, time variables $\mathcal{V}_{\mathbb{N}}$ and binary predicates defined from $\mathcal{X}$ (i.e., from the set $\{X_a \mid a \in \mathcal{X}\}$).
Is there a set of traces $\mathrm{T} \subseteq (\mathbf{2}^{\mathcal{X}})^\omega$ that is a model of φ; that is, $[\![\varphi]\!] \neq \emptyset$?

We present in Table 1 a summary of decidability results proved in this work for the satisfiability problem of hypertrace logic. We describe fragments by specifying patterns on its quantifiers: we denote $\mathbb{Q} \in \{\forall, \exists\}$, while $\mathbb{Q}_{\mathbb{N}_<}$ denotes time quantifiers, and $\mathbb{Q}_{\mathbb{T}}$ and $\mathbb{Q}_{\mathbb{T}}{::}T$ denotes unconstrained and constrained quantifiers, respectively. We then combine these symbols to define patterns as regular expressions.

Trace-prefixed	
$\exists^*_{\mathbb{T}}(\exists_{\mathbb{T}}{::}T)^*(\forall_{\mathbb{T}}{::}T)^*\mathbb{Q}^*_{\mathbb{T}}\mathbb{Q}^*_{\mathbb{N}_<}$	Decidable [10] [Cor. 16]
$(\forall_{\mathbb{T}}{::}T)^2\exists_{\mathbb{T}}{::}T\ \mathbb{Q}^+_{\mathbb{N}_<}$	Undecidable [6] [Prop. 15]
Time-prefixed	
$\exists^*_{\mathbb{N}_<}\mathbb{E}^*_{\mathbb{T}}(\exists_{\mathbb{T}}{::}T)^*(\forall_{\mathbb{T}}{::}T)^*\mathbb{Q}^*_{\mathbb{T}}$	Decidable [Cor. 21]
$\exists_{\mathbb{N}_<}\forall_{\mathbb{N}_<}\exists^2_{\mathbb{N}_<}\forall_{\mathbb{N}_<}\forall_{\mathbb{T}}{::}T(\exists_{\mathbb{T}}{::}T)^2\exists_{\mathbb{T}}$	Undecidable [Thm. 22]

Table 1 Summary of results on the decidability of hypertrace formulas satisfiability.

4 Unconstrained Hypertrace Logic

We look into the fragment of hypertrace formulas without constrained quantifiers (i.e., no subformulas with shape $\mathbb{Q}\pi{:}T\ \varphi$), which we refer to as *unconstrained hypertrace logic*. We prove that unconstrained hypertrace logic is expressively equivalent to monadic second-order logic of order of one successor (`S1S`). The translation follows naturally from Lemma 5. Specifically, we show that quantification over variables in each unconstrained trace is equivalent to second-order quantification over the sets of time points at which the corresponding propositional variable holds in that trace.

4.1 Equivalence to S1S

To establish the equivalence between unconstrained hypertrace logic and S1S, we define a translation between unconstrained hypertrace formulas and models to their counterpart in S1S, and vice-versa. The translation from unconstrained hypertrace formulas to S1S rewrites the formulas using $\texttt{flatten}$, while reinterpreting each π_x as a second-order variable.

To translate from unconstrained formulas to S1S formulas, we define the substitution $\sigma = \{x(\pi_x, i) \mapsto \pi_x(i) \mid x \in \mathcal{X}, \pi \in \mathcal{V}_\mathbb{T}, i \in \mathcal{V}_\mathbb{N}\}$ and the rewriting function $\texttt{toS1S}(\varphi, \mathcal{X}) = \texttt{flatten}(\varphi, \mathcal{X}, \mathcal{V}_\mathbb{T} \cap \text{free}(\varphi))[\sigma]$ where φ is an unconstrained trace formula. This translation does not work for formulas with constrained trace quantifiers because, in general, we cannot assume T to be S1S-definable. The next step is to define a translation for trace assignments. A trace assignment $\Pi_\mathbb{T} : \mathcal{V}_\mathbb{T} \to (\mathbf{2}^\mathcal{X})^\omega$ is translated to an assignment over second-order variables where each variable is mapped to its support set. Formally, $\texttt{toSuppSet}(\Pi_\mathbb{T}) : \mathcal{V}_\mathcal{X} \to \mathbf{2}^\mathbb{N}$ where $\texttt{toSuppSet}(\Pi_\mathbb{T})(\pi_x) = \{i \mid x \in \Pi_\mathbb{T}(\pi)[i]\}$.

For the translation from S1S to unconstrained hypertrace formulas, each second-order variable X becomes the trace variable τ_X. Additionally, we use the standard translation of Succ using $\leq$.

$$
\begin{aligned}
\texttt{toHyper}(\exists X\ \varphi') &= \exists \pi_X\ \texttt{toHyper}(\varphi') & \qquad \texttt{toHyper}(X(i)) &= X(\pi_X, i) \\
\texttt{toHyper}(\exists i\ \varphi') &= \exists i\ \texttt{toHyper}(\varphi') & \texttt{toHyper}(i = j) &= (i = j) \\
\texttt{toHyper}(\neg\varphi') &= \neg\texttt{toHyper}(\varphi') & & \\
\texttt{toHyper}(\varphi_1 \vee \varphi_2) &= \texttt{toHyper}(\varphi_1) \vee \texttt{toHyper}(\varphi_2) & & \\
\texttt{toHyper}(\text{Succ}(i, i')) &= i < i' \wedge (\forall j\ (i < j \to i' \leq j)) & &
\end{aligned}
\tag{7}
$$

We translate second-order assignments, Π_2, to trace assignments as: $\texttt{toBool}(\Pi_2) : \mathcal{V}_\mathbb{T} \to (\mathbf{2}^\mathcal{X})^\omega$ where for all $i \in \mathbb{N}$ and $X \in \mathcal{X}$, $X \in (\texttt{toBool}(\Pi_2)(\pi_X))[i]$ iff $i \in \Pi_2(X)$.

▶ **Theorem 7.** *Let T be a set of traces over $\mathcal{X}$. For all unconstrained hypertrace formulas φ over $\mathcal{X}$: $(\overline{T}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \varphi$ iff $(\Pi_\mathbb{N}, \texttt{toSuppSet}(\Pi_\mathbb{T})) \models_{S1S} \texttt{toS1S}(\varphi, \mathcal{X})$. For all S1S formulas: $(\Pi_1, \Pi_2) \models_{S1S} \varphi$ iff $(\overline{T}, (\texttt{toBool}(\Pi_2), \Pi_1)) \models_\mathbb{T} \texttt{toHyper}(\varphi)$.*

Proof. We start by proving, by structural induction, the translation from unconstrained hypertrace formulas and trace assignments to S1S formulas and second-order assignments. For the base case, we need to prove $(\overline{\text{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} x(\pi, i)$ iff $(\Pi_\mathbb{N}, \texttt{toSuppSet}(\Pi_\mathbb{T})) \models_{\text{S1S}} \pi_x(i)$. Given $k = \Pi_\mathbb{N}(i)$, we observe that, by definition, $x \in \Pi_\mathbb{T}(\pi)[k]$ iff $k \in (\texttt{toSuppSet}(\Pi_\mathbb{T}))(\pi_x)$. Hence, the base case holds. For the induction cases, we use the induction hypothesis and Lemma 5.

We consider now the translation from S1S formulas and second-order assignments to unconstrained hypertrace formulas and trace assignments. We have two base cases. We start by proving that $(\Pi_1, \Pi_2) \models_{\text{S1S}} X(i)$ iff $(\overline{\text{T}}, (\texttt{toBool}(\Pi_2), \Pi_1) \models_\mathbb{T} X(\pi_X, i)$. This holds, because for $k = \Pi_\mathbb{N}(i)$, $X \in (\texttt{toBool}(\Pi_2)(\pi_X))[k]$ iff $k \in \Pi_2(X)$. The second base-case is $(\Pi_1, \Pi_2) \models_{\text{S1S}} \text{Succ}(i,i')$ iff $(\overline{\text{T}}, (\texttt{toBool}(\Pi_2), \Pi_1)) \models_\mathbb{T} i < i' \wedge (\forall j\ (i < j \to i' \leq j))$, holding by definition of $\leq$. For the induction cases, we use the induction hypothesis and Lemma 5. ◀

From the previous theorem and S1S having a decidable satisfiability checking problem [19], checking whether an unconstrained hypertrace formula is satisfiable is also decidable.

▶ **Corollary 8.** *The satisfiability problem for unconstrained hypertrace logic is decidable.*

4.2 Relation to Full Hypertrace Logic

In this section, we show that any hypertrace formula with a single alternation from an existential to a universal constrained quantifier can be rewritten into an equisatisfiable formula without constrained quantifiers. We prove our results by adapting techniques introduced in [6, 5] to rewrite HyperQPTL formulas into QPTL. We start by showing how to eliminate the universal constrained trace quantifiers.

▶ **Lemma 9.** *Let* $\varphi = \overrightarrow{\mathbb{E}}\ \exists \pi_1{::}T \ldots \exists \pi_n{::}T\ \forall \pi'_1{::}T \ldots \forall \pi'_m{::}T\ \overrightarrow{\mathbb{Q}}\ \varphi_{qf}$ *be a hypertrace formula in prenex normal form s.t.* $\mathbb{E}$ *is any combination of existential time or unconstrained trace quantifiers,* $\overrightarrow{\mathbb{Q}}$ *is any combination of time or unconstrained trace quantifiers and* φ_{qf} *is a quantifier-free hypertrace formula. Let*

$$\texttt{removeForAll}(\varphi) = \overrightarrow{\mathbb{E}}\ \exists \pi_1{::}T \ldots \exists \pi_n{::}T \bigwedge_{j_1=1}^{n} \ldots \bigwedge_{j_m=1}^{n} \overrightarrow{\mathbb{Q}}\ \varphi_{qf}[\pi'_1 \mapsto \pi_{j_1}, \ldots, \pi'_m \mapsto \pi_{j_m}]$$

where $\varphi_{qf}[\pi'_1 \mapsto \pi_{j_1}, \ldots, \pi'_m \mapsto \pi_{j_m}]$ *is the formula obtained by substituting* π'_i *with* π_{j_i}*, for all* $1 \leq i \leq m$*, in* φ_{qf}*. Then,* $[\![\varphi]\!] \neq \emptyset$ *iff* $[\![\texttt{removeForAll}(\varphi)]\!] \neq \emptyset$*.*

Proof. A set of trace variables $\mathcal{V}$ and a trace assignment $\Pi_\mathbb{T}$ that includes assignments for all variables in $\mathcal{V}$ (i.e., $\mathcal{V} \subseteq \mathrm{Dom}(\Pi_\mathbb{T})$) define the set of traces $\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})} = \{\Pi_\mathbb{T}(\pi) \mid \pi \in \mathcal{V}\}$. For a set of free variables and assignment, $\mathcal{V}$ and $\Pi_\mathbb{T}$, when we evaluate the subformula starting with the universal constrained quantifier against the set of traces $\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}$, we can remove the universal quantifiers by considering all possible combinations of assignments to the traces in $\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}$. Formally, for any hypertrace formula $\forall \pi'_1{::}T \ldots \forall \pi'_m{::}T\ \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}}$ where $\overrightarrow{\mathbb{Q}}$ is any combination of time or unconstrained trace quantifiers, φ_{qf} is a quantifier-free hypertrace formula, and $\mathcal{V} = \{\pi_1, \ldots, \pi_n\}$ s.t. $\mathcal{V} \subseteq \mathrm{free}(\forall \pi'_1{::}T \ldots \forall \pi'_m{::}T\ \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}})$. Formally:

$$(\overline{\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \forall \pi'_1{::}T \ldots \forall \pi'_m{::}T\ \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \text{ iff} \tag{8}$$
$$(\overline{\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \bigwedge_{j_1=1}^{n} \ldots \bigwedge_{j_m=1}^{n} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}}[\pi'_1 \mapsto \pi_{j_1}, \ldots, \pi'_m \mapsto \pi_{j_m}].$$

We prove this equivalence below:

$$(\overline{\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \forall \pi'_1{::}T \ldots \forall \pi'_m{::}T\ \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \overset{\text{def.}}{\Leftrightarrow} \models_\mathbb{T}$$
$$\text{for all } \tau_1 \in \mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}, \ldots, \tau_m \in \mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})} :$$
$$(\overline{\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}, (\Pi_\mathbb{T}[\pi_1 \mapsto \tau_1, \ldots, \pi_m \mapsto \tau_m], \Pi_\mathbb{N})) \models_\mathbb{T} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \overset{\text{def. } \mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}{\Leftrightarrow}$$
$$\text{for all } \tau_1 \in \{\Pi_\mathbb{T}[\pi_1], \ldots, \Pi_\mathbb{T}[\pi_n]\}, \ldots, \tau_m \in \{\Pi_\mathbb{T}[\pi_1], \ldots, \Pi_\mathbb{T}[\pi_n]\} :$$
$$(\overline{\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}, (\Pi_\mathbb{T}[\pi_1 \mapsto \tau_1, \ldots, \pi_m \mapsto \tau_m], \Pi_\mathbb{N})) \models_\mathbb{T} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \Leftrightarrow$$
$$\text{for all } \pi_{j_1} \in \{1, \ldots, n\}, \ldots, \pi_{j_m} \in \{1, \ldots, n\} :$$
$$(\overline{\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}, (\Pi_\mathbb{T}[\pi_1 \mapsto \Pi_\mathbb{T}[\pi_{j_1}], \ldots, \pi_m \mapsto \Pi_\mathbb{T}[\pi_{j_m}]], \Pi_\mathbb{N})) \models_\mathbb{T} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \overset{\text{finite domains}}{\Leftrightarrow}$$
$$\bigwedge_{j_1=1}^{n} \ldots \bigwedge_{j_m=1}^{n} : (\overline{\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}, (\Pi_\mathbb{T}[\pi_1 \mapsto \Pi_\mathbb{T}[\pi_{j_1}], \ldots, \pi_m \mapsto \Pi_\mathbb{T}[\pi_{j_m}]], \Pi_\mathbb{N})) \models_\mathbb{T} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \overset{\text{def.}}{\Leftrightarrow} \models_\mathbb{T}$$
$$(\overline{\mathrm{T}_{(\Pi_\mathbb{T},\mathcal{V})}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \bigwedge_{j_1=1}^{n} \ldots \bigwedge_{j_m=1}^{n} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}}[\pi'_1 \mapsto \pi_{j_1}, \ldots, \pi'_m \mapsto \pi_{j_m}].$$

Before we prove the main claim, we need to prove that the set of models of a hypertrace formulas without existential constrained quantifiers is subset-closed.

Claim 1: *Let φ be a hypertrace formula in prenex and negation normal form (i.e., all quantifiers at the beginning of the formula and negation only at the atomic level) without existential constrained trace quantifiers. For all sets of traces T and its subsets $T' \subseteq T$, and assignments $\Pi_\mathbb{T}$ and $\Pi_\mathbb{N}$: if $(\overline{T}, (\Pi_\mathbb{T}, \Pi_\mathbb{N}) \models_\mathbb{T} \varphi$, then $(\overline{T'}, (\Pi_\mathbb{T}, \Pi_\mathbb{N}) \models_\mathbb{T} \varphi$.*

Proof of claim 1: We prove by structural induction on φ. For the bases cases, we observe that $\models_\mathbb{T}$ is independent of the set of traces, hence, as the assignments are preserved, the property holds. The induction cases $\varphi \vee \varphi'$, $\varphi \wedge \varphi'$, $\exists i\ \varphi$ and $\forall i\ \varphi$ follow directly from definitions and induction hypothesis. The only case remaining is $\forall \pi :: T\ \varphi$. Consider arbitrary set of traces T and $\mathrm{T}' \subseteq \mathrm{T}$ and assignments $\Pi_\mathbb{T}$ and $\Pi_\mathbb{N}$. We assume that $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N}) \models_\mathbb{T} \forall \pi :: T\ \varphi$, which, by definition of $\models_\mathbb{T}$, is equivalent to: for all traces $\tau \in \mathrm{T}$, $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}[\pi \mapsto \tau], \Pi_\mathbb{N}) \models_\mathbb{T} \varphi$. By induction hypothesis and $\mathrm{T}' \subseteq \mathrm{T}$, for all traces $\tau \in \mathrm{T}'$, $(\overline{\mathrm{T}'}, (\Pi_\mathbb{T}[\pi \mapsto \tau], \Pi_\mathbb{N}) \models_\mathbb{T} \varphi$. Hence, by definition of $\models_\mathbb{T}$, $(\overline{\mathrm{T}'}, (\Pi_\mathbb{T}, \Pi_\mathbb{N}) \models_\mathbb{T} \forall \pi :: T\ \varphi$.

We prove now each side of the implication. We assume without loss of generality that the formula is in prenex and negation normal form. In what follows, $\mathcal{V}_\exists = \{\pi_1, \ldots, \pi_n\}$. We start by proving that $[\![\varphi]\!] \neq \emptyset \Rightarrow [\![\texttt{removeForAll}(\varphi)]\!] \neq \emptyset$:

$[\![\varphi]\!] \neq \emptyset \Leftrightarrow \text{exists } \mathrm{T} \in [\![\overrightarrow{\mathbb{E}}\ \exists \pi_1 :: T \ldots \exists \pi_n :: T\ \forall \pi'_1 :: T \ldots \forall \pi'_m :: T\ \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}}]\!] \overset{\text{def.} \models_\mathbb{T}}{\Leftrightarrow}$

exists $\mathrm{T}, \Pi_\mathbb{T}$ and $\Pi_\mathbb{N}$ s.t. $\Pi_\mathbb{T}(\pi) \in \mathrm{T}$ with $\pi \in \mathcal{V}_\exists$:

$$(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T}\ \forall \pi'_1 :: T \ldots \forall \pi'_m :: T\ \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \overset{\mathrm{T}_{(\Pi_\mathbb{T}, \mathcal{V}_\exists)} \subseteq \mathrm{T}, \text{Claim 1}}{\Rightarrow}$$

exists $\Pi_\mathbb{T}$ and $\Pi_\mathbb{N}$: $(\overline{\mathrm{T}_{(\Pi_\mathbb{T}, \mathcal{V}_\exists)}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \forall \pi'_1 :: T \ldots \forall \pi'_m :: T\ \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \overset{(8)}{\Leftrightarrow}$

exists $\Pi_\mathbb{T}$ and $\Pi_\mathbb{N}$:

$$(\overline{\mathrm{T}_{(\Pi_\mathbb{T}, \mathcal{V}_\exists)}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \bigwedge_{j_1=1}^{n} \ldots \bigwedge_{j_m=1}^{n} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}}[\pi'_1 \mapsto \pi_{j_1}, \ldots, \pi'_m \mapsto \pi_{j_m}] \Leftrightarrow$$

exists $\Pi_\mathbb{T}$ and $\Pi_\mathbb{N}$: $\mathrm{T}_{(\Pi_\mathbb{T}, \mathcal{V}_\exists)} \models_\mathbb{T} \texttt{removeForAll}(\varphi) \quad \Rightarrow \quad [\![\texttt{removeForAll}(\varphi)]\!] \neq \emptyset$

And now, we prove $[\![\texttt{removeForAll}(\varphi)]\!] \neq \emptyset \Rightarrow [\![\varphi]\!] \neq \emptyset$:

$[\![\texttt{removeForAll}(\varphi)]\!] \neq \emptyset \Leftrightarrow$

$$\text{exists } \mathrm{T} \in [\![\overrightarrow{\mathbb{E}}\ \exists \pi_1 :: T \ldots \exists \pi_n :: T \bigwedge_{j_1=1}^{n} \ldots \bigwedge_{j_m=1}^{n} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}}[\pi'_1 \mapsto \pi_{j_1}, \ldots, \pi'_m \mapsto \pi_{j_m}]]\!] \overset{\text{def.} \models_\mathbb{T}}{\Leftrightarrow}$$

$$\text{exists } \mathrm{T}, \Pi_\mathbb{T} \text{ and } \Pi_\mathbb{N} : (\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \bigwedge_{j_1=1}^{n} \ldots \bigwedge_{j_m=1}^{n} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}}[\pi'_1 \mapsto \pi_{j_1}, \ldots, \pi'_m \mapsto \pi_{j_m}] \overset{\substack{\mathrm{T}_{(\Pi_\mathbb{T}, \mathcal{V}_\exists)} \subseteq \mathrm{T} \\ \text{Claim 1, (8)}}}{\Rightarrow}$$

$$\text{exists } \Pi_\mathbb{T} \text{ and } \Pi_\mathbb{N} : (\overline{\mathrm{T}_{(\Pi_\mathbb{T}, \mathcal{V}_\exists)}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \bigwedge_{j_1=1}^{n} \ldots \bigwedge_{j_m=1}^{n} \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}}[\pi'_1 \mapsto \pi_{j_1}, \ldots, \pi'_m \mapsto \pi_{j_m}] \Leftrightarrow$$

$$\text{exists } \Pi_\mathbb{T} \text{ and } \Pi_\mathbb{N} : (\overline{\mathrm{T}_{(\Pi_\mathbb{T}, \mathcal{V}_\exists)}}, (\Pi_\mathbb{T}, \Pi_\mathbb{N})) \models_\mathbb{T} \forall \pi'_1 :: T \ldots \forall \pi'_m :: T\ \overrightarrow{\mathbb{Q}}\ \varphi_{\mathrm{qf}} \Rightarrow \text{exists } \mathrm{T} \in [\![\varphi]\!] \qquad \blacktriangleleft$$

▶ **Example 10.** for the formula: $\varphi = \exists \pi_1 \exists \pi_2 \forall \pi_3 \exists i \exists j\ a(\pi_1, i) \wedge \neg a(\pi_2, i) \wedge b(\pi_3, j)$, we have

$$\texttt{removeForAll}(\varphi) = \exists \pi_1 \exists \pi_2\ (\exists i \exists j\ a(\pi_1, i) \wedge \neg a(\pi_2, i) \wedge b(\pi_1, j)) \wedge (\exists i \exists j\ a(\pi_1, i) \wedge \neg a(\pi_2, i) \wedge b(\pi_2, j))$$

which can be simplified to $\exists \pi_1 \exists \pi_2 \exists i\ a(\pi_1, i) \wedge \neg a(\pi_2, i) \wedge (\exists j\ b(\pi_1, j)) \wedge (\exists j\ b(\pi_2, j))$.

We observe that hypertrace formulas with only existential constrained trace quantifiers are equisatisfiable to an unconstrained formula. Intuitively, each existential trace quantifier can be instantiated independently of the others.

▶ **Lemma 11.** *Let* $\varphi = \overrightarrow{\mathbb{E}}\ \exists\pi_1{::}T \ldots \exists\pi_n{::}T\ \overrightarrow{\mathbb{Q}}\ \varphi_{qf}$ *be a hypertrace formula in prenex normal form s.t.* $\overrightarrow{\mathbb{E}}$ *is a any combination of existential time or unconstrained trace quantifiers,* $\overrightarrow{\mathbb{Q}}$ *is any combination of time or unconstrained trace quantifiers and* φ_{qf} *is a quantifier-free hypertrace formula.* $[\![\varphi]\!] \neq \emptyset$ *iff* $[\![\overrightarrow{\mathbb{E}}\ \exists\pi_1 \ldots \exists\pi_n\ \overrightarrow{\mathbb{Q}}\ \varphi_{qf}]\!] \neq \emptyset$.

Finally, for all hypertrace formulas where all universal constrained trace quantifiers are only followed by existential constrained quantifiers, we can apply the rewrite described in Lemma 9 and Lemma 11 to get an equisatisfiable unconstrained hypertrace formula.

▶ **Theorem 12.** *Let* $\varphi = \overrightarrow{\mathbb{E}}\ \exists\pi_1{::}T \ldots \exists\pi_n{::}T\ \forall\pi'_1{::}T \ldots \forall\pi'_m{::}T\ \overrightarrow{\mathbb{Q}}\ \varphi_{qf}$ *be a hypertrace formula in prenex normal form s.t.* $\mathbb{E}$ *is any combination of existential time or unconstrained trace quantifiers,* $\overrightarrow{\mathbb{Q}}$ *is any combination of time or unconstrained trace quantifiers and* φ_{qf} *is a quantifier-free hypertrace formula. There exists an unconstrained hypertrace formula* φ_u *s.t.:* $[\![\varphi]\!] \neq \emptyset$ *iff* $[\![\varphi_u]\!] \neq \emptyset$.

5 Trace-prefixed Hypertrace Logic

In this section, we consider the fragment of FO$[<, \mathbb{T}]$ where trace quantifiers come before time quantifiers. We call this the *trace-prefixed hypertrace logic*, denoted $\mathbf{T}$-FO$[<, \mathbb{T}]$. Formally, trace-prefixed formulas $\varphi \in \mathbf{T}$-FO$[<, \mathbb{T}]$ are defined by the grammar:

$$\varphi ::= \exists\pi\, \varphi \mid \exists\pi{::}T\, \varphi \mid \neg\varphi \mid \psi \qquad \psi ::= \exists i\ \psi \mid \psi \vee \psi \mid \neg\psi \mid i < i \mid i = i \mid X(\pi, i)$$

where π is a trace variable, i is a time variable and X is a binary predicate.

We observe that $\mathbf{T}$-FO$[<, \mathbb{T}]$ is orthogonal to unconstrained hypertrace logic. While the unconstrained fragment allows any mix of (unconstrained) trace and time quantifiers, $\mathbf{T}$-FO$[<, \mathbb{T}]$ requires trace quantifiers to come first. On the other hand, $\mathbf{T}$-FO$[<, \mathbb{T}]$ introduces constrained trace quantifiers, which the unconstrained version does not support.

▶ **Example 13.** We can express *bounded promptness* in $\mathbf{T}$-FO$[<, \mathbb{T}]$:

$$\exists\pi\ \forall\pi'{::}T\ \exists i\ \forall j\ \big((j < i \rightarrow \neg p(\pi, j)) \wedge p(\pi, i) \wedge q(\pi', i)\big). \tag{9}$$

In this property, the unconstrained trace is used to guess a synchronization point (i.e., the time when p becomes true in π) where all traces agree with the value of proposition q.

In [1], the authors study the trace-prefixed hypertrace logic for the fragment of FO$[<, \mathbb{T}]$ with only constrained trace quantifiers, which we refer to as $\mathbf{T}$-FO$[<, \mathbb{T}{::}T]$. They prove that $\mathbf{T}$-FO$[<, \mathbb{T}{::}T]$ is expressively equivalent to HyperLTL. From this result, we can prove that adding unconstrained quantifiers to $\mathbf{T}$-FO$[<, \mathbb{T}{::}T]$ extends its expressive power.

▶ **Proposition 14.** $\boldsymbol{T}$*-FO*$[<, \mathbb{T}]$ *is strictly more expressive than* $\boldsymbol{T}$*-FO*$[<, \mathbb{T}{::}T]$.

Proof. Clearly, $\mathbf{T}$-FO$[<, \mathbb{T}{::}T]$ is a fragment of $\mathbf{T}$-FO$[<, \mathbb{T}]$. To prove that $\mathbf{T}$-FO$[<, \mathbb{T}{::}T]$ is not equally expressive to $\mathbf{T}$-FO$[<, \mathbb{T}]$, we observe that bounded promptness (9) is not expressible in HyperLTL [4] and, from $\mathbf{T}$-FO$[<, \mathbb{T}{::}T]$ being equally expressive to HyperLTL [1], it is also not expressible in $\mathbf{T}$-FO$[<, \mathbb{T}{::}T]$. ◀

We can also use $\mathbf{T}$-FO$[<, \mathbb{T}{::}T]$ to get our first undecidability result for $\mathbf{T}$-FO$[<, \mathbb{T}]$.

▶ **Proposition 15.** *Let* $\varphi = \forall\pi_1{::}T\ \forall\pi_2{::}T\ \exists\pi'_1{::}T\ \psi$ *where* ψ *has only time quantifiers. It is undecidable to check whether* $[\![\varphi]\!] \neq \emptyset$.

Proof. We prove by reduction of the HyperLTL satisfiability problem for formulas with quantifier pattern $\forall\forall\exists$ to the problem of checking for satisfiability of hypertrace formulas with quantifier pattern $\forall{::}T\ \forall{::}T\ \exists{::}T$. Consider an arbitrary HyperLTL formula $\varphi = \forall\pi_1\forall\pi_2\exists\pi_1'\psi$ where ψ has only temporal operators. By $\mathbf{T}$-FO$[<,\mathbb{T}{::}T]$ being expressively equivalent to HyperLTL, there exists $\varphi' \in \mathbf{T}\text{-FO}[<,\mathbb{T}{::}T]$ s.t. $[\![\varphi]\!] = [\![\varphi']\!]$. Using the computable translation in [1], $\varphi' = \forall\pi_1{::}T\ \forall\pi_2{::}T\ \exists\pi_1'{::}T\psi'$ where ψ' has only temporal quantifiers. From [6, 14], it is undecidable to check for satisfiability of HyperLTL formulas with quantifier pattern $\forall\forall\exists$. Hence, it is also not possible to decide $[\![\varphi]\!] \neq \emptyset$. ◀

5.1 Satisfiability

From Theorem 12 and Corollary 8, we get our first fragment of $\mathbf{T}$-FO$[<,\mathbb{T}]$ with a decidable satisfiability problem: the fragment where constrained trace quantifiers is of shape $\exists^*\forall^*$. This matches the known decidable fragment for HyperQPTL [10].

▶ **Corollary 16.** *Let $\varphi = \overrightarrow{\mathbb{E}_\mathbb{T}}\ \exists\pi_1{::}T \ldots \exists\pi_n{::}T\ \forall\pi_1'{::}T \ldots \forall\pi_m'{::}T\ \overrightarrow{\mathbb{Q}_\mathbb{T}}\ \overrightarrow{\mathbb{Q}_{\mathbb{N}_<}}\ \varphi_{qf}$ be a hypertrace formula in prenex normal form s.t. $\overrightarrow{\mathbb{E}_\mathbb{T}}$ is a sequence of existential unconstrained trace quantifiers, $\overrightarrow{\mathbb{Q}_\mathbb{T}}$ and $\overrightarrow{\mathbb{Q}_{\mathbb{N}_<}}$ are any combination of time and unconstrained trace quantifiers, respectively, and φ_{qf} is a quantifier-free. It is decidable to check whether $[\![\varphi]\!] \neq \emptyset$.*

Unconstrained trace quantifiers not only extend the expressivity of $\mathbf{T}$-FO$[<,\mathbb{T}]$ compared to $\mathbf{T}$-FO$[<,\mathbb{T}{::}T]$ but can also be used to seamlessly define a semi-decision procedure to check for the unsatisfiability of $\mathbf{T}$-FO$[<,\mathbb{T}{::}T]$ formulas. In the proposition below, we prove that removing constraints from existentially quantified trace variables preserves the models of the constrained formula. We can use the contrapositive of this proposition to determine the unsatisfiability of constrained hypertrace formulas.

▶ **Proposition 17.** *Let $\varphi = \forall\pi_0{::}T\ \exists\pi_0'{::}T \ldots \forall\pi_k{::}T\ \exists\pi_k'{::}T\ \psi$ be a hypertrace formula where ψ is quantifier-free, and T be a set of traces. If $T \models_\mathbb{T} \varphi$, then $T \models_\mathbb{T} \forall\pi_0{::}T\ \exists\pi_0' \ldots \forall\pi_k{::}T\ \exists\pi_k'\ \psi$.*

Proof. The statement follows directly from T being a subset of all possible traces and, thus, any assignment over T is also an assignment over the set of all traces. ◀

5.2 Equivalence to HyperQPTL

We prove that, for sets of infinite traces, trace-prefixed hypertrace logic is expressively equivalent to HyperQPTL [17]. HyperQPTL formulas φ are defined by the grammar:

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

where $\pi \in \mathcal{V}_\mathbb{T}$ is a trace variable and $q, a \in \mathcal{X}$ are propositional variables, where $\mathcal{V}_\mathbb{T} \cap \mathcal{X} = \emptyset$. We evaluate HyperQPTL formulas over a set of traces and a trace assignment $\Pi : \mathcal{V}_\mathbb{T} \to (\mathbf{2}^\mathcal{X})^\omega$, relative to a time point $i \in \mathbb{N}$:

$(\Pi, \mathrm{T}, i) \models_{\mathtt{HQ}} \exists q\ \psi$ iff there exists $\tau \in (\mathbf{2}^{\{q\}})^\omega : (\Pi[\pi_q \mapsto \tau], \mathrm{T}, i) \models_{\mathtt{HQ}} \psi$;
$(\Pi, \mathrm{T}, i) \models_{\mathtt{HQ}} \forall q\ \psi$ iff for all $\tau \in (\mathbf{2}^{\{q\}})^\omega : (\Pi[\pi_q \mapsto \tau], \mathrm{T}, i) \models_{\mathtt{HQ}} \psi$;
$(\Pi, \mathrm{T}, i) \models_{\mathtt{HQ}} \exists\pi\ \psi$ iff there exists $\tau \in \mathrm{T} : (\Pi[\pi \mapsto \tau], \mathrm{T}, i) \models_{\mathtt{HQ}} \psi$;
$(\Pi, \mathrm{T}, i) \models_{\mathtt{HQ}} \forall\pi\ \psi$ iff for all $\tau \in \mathrm{T} : (\Pi[\pi \mapsto \tau], \mathrm{T}, i) \models_{\mathtt{HQ}} \psi$;
$(\Pi, \mathrm{T}, i) \models_{\mathtt{HQ}} q$ iff $q \in \Pi(\pi_q)[i]$; $(\Pi, \mathrm{T}, i) \models_{\mathtt{HQ}} a_\pi$ iff $a \in \Pi(\pi)[i]$;
$(\Pi, \mathrm{T}, i) \models_{\mathtt{HQ}} \neg\psi$ iff $(\Pi, \mathrm{T}, i) \not\models_{\mathtt{HQ}} \psi$;

$(\Pi, \mathrm{T}, i) \models_{\texttt{HQ}} \psi_1 \vee \psi_2$ iff $(\Pi, \mathrm{T}, i) \models_{\texttt{HQ}} \psi_1$ or $(\Pi, \mathrm{T}, i) \models_{\texttt{HQ}} \psi_2$;

$(\Pi, \mathrm{T}, i) \models_{\texttt{HQ}} \mathbf{X}\psi$ iff $(\Pi, \mathrm{T}, i+1) \models_{\texttt{HQ}} \psi$;

$(\Pi, \mathrm{T}, i) \models_{\texttt{HQ}} \psi_1 \,\mathbf{U}\, \psi_2$ iff exists $j \geq i : (\Pi, \mathrm{T}, j) \models_{\texttt{HQ}} \psi_2$ and all $i \leq j' < j$ $(\Pi, \mathrm{T}, j') \models_{\texttt{HQ}} \psi_1$.

A set T of traces is a model of a HyperQPTL formula φ, denoted $\mathrm{T} \models_{\texttt{HQ}} \varphi$, iff there exists an assignment Π such that $(\Pi, \mathrm{T}, 0) \models_{\texttt{HQ}} \varphi$. For all closed formulas φ, $\mathrm{T} \models_{\texttt{HQ}} \varphi$ iff $(\Pi^{\emptyset}, \mathrm{T}, 0) \models_{\texttt{HQ}} \varphi$, where $\Pi^{\emptyset}$ is the empty assignment.

▶ Remark 18. In this work, we interpret HyperQPTL formulas as introduced in [17, 5]. An alternative definition found in the literature uniformly instantiates quantified propositions across the set of traces used as the model [10, 7, 20, 18]. The semantics we adopt in this work subsumes the uniform interpretation; that is, we can simulate the uniform interpretation by rewriting the HyperQPTL formula interpreted under the uniform semantics.

For a set of traces T and an assignment Π for variables in $\mathcal{V}$, we define its *flattening* to a trace as: $a_\pi \in \langle\Pi\rangle[i]$ iff $a \in \Pi(\pi)[i]$ and $q \in \langle\Pi\rangle[i]$ iff $q \in \Pi(\pi_q)[i]$ for all $i \in \mathbb{N}$, trace variables $\pi \in \mathcal{V}$ and propositional variables $a, q \in \mathcal{X}$. A trace assignment satisfies a quantifier-free HyperQPTL formula iff its flattening satisfies the same formula under the LTL semantics.

▶ **Proposition 19.** *Let φ a quantifier-free* HyperQPTL *formula. For all $i \in \mathbb{N}$, all trace sets T, and all trace assignments* Π, $(\Pi, T, i) \models_{\texttt{HQ}} \varphi$ *iff* $\langle\Pi\rangle[i \ldots] \models_{\texttt{LTL}} \varphi$.

▶ **Theorem 20.** *For all* HyperQPTL *sentences φ_H there exists a trace-prefixed hypertrace sentence φ such that for all sets of infinite traces $T \subseteq (\mathbf{2}^{\mathcal{X}})^\omega$, $T \models_{\texttt{HQ}} \varphi_H$ iff $T \models_{\mathbb{T}} \varphi$. For all trace-prefixed hypertrace sentences φ there exists a* HyperQPTL *sentence φ_H such that for all sets of infinite traces $T \subseteq (\mathbf{2}^{\mathcal{X}})^\omega$, $T \models_{\texttt{HQ}} \varphi_H$ iff $T \models_{\mathbb{T}} \varphi$.*

Proof. We denote by $\texttt{LTLtoFO}(\psi)$ and $\texttt{FOtoLTL}(\psi)$ the translation from LTL formulas to FO[<] and vice-versa given by LTL and FO[<] being equivalent [9].

We start with the translation from HyperQPTL formulas to an equivalent $\mathbf{T}$-FO[$<, \mathbb{T}$] formula. We first define the translation of $\varphi \in$ HyperQPTL for its different quantifiers:

$$\texttt{tr}_H^{\text{quant}}(\varphi) = \begin{cases} \varphi & \text{if } \varphi \text{ is quantifier-free} \\ \mathbb{Q}\pi{::}T\ \texttt{tr}_H^{\text{quant}}(\varphi') & \text{if } \varphi = \mathbb{Q}\pi\ \varphi', \mathbb{Q} \in \{\forall, \exists\} \text{ and } \pi \in \mathcal{V}_{\mathbb{T}} \\ \mathbb{Q}\pi_q\ \texttt{tr}_H^{\text{quant}}(\varphi') & \text{if } \text{ if } \varphi = \mathbb{Q}q\ \varphi', \mathbb{Q} \in \{\forall, \exists\} \text{ and } q \in \mathcal{X} \end{cases} \tag{10}$$

With $\texttt{LTLtoFO}(\psi)$ all propositional variables a_π in ψ are mapped to $P_{a_\pi}(i)$, while all q are mapped to $P_q(i)$. We define the substitution from these predicates to the binary predicates of FO[$<, \mathbb{T}$]: $\sigma = \{P_{a_\pi}(i) \mapsto X_a(\pi, i) \mid a \in \mathcal{X}, \pi \in \mathcal{V}_{\mathbb{T}}, i \in \mathcal{V}_{\mathbb{N}}\} \cup \{P_q(i) \mapsto X_q(\pi_q, i) \mid q \in \mathcal{X}, \pi \in \mathcal{V}_{\mathbb{T}}, i \in \mathcal{V}_{\mathbb{N}}\}$. The translation from HyperQPTL formulas to $\mathbf{T}$-FO[$<, \mathbb{T}$] is defined below, where $\psi \in$ LTL, $x_i \in \mathcal{V}_{\mathbb{T}} \cup \mathcal{X}$, for $1 \leq i \leq n$, and $\mathbb{Q} \in \{\forall, \exists\}$:

$$\texttt{tr}_H(\mathbb{Q}x_1 \ldots \mathbb{Q}x_n\ \psi) = \texttt{tr}_H^{\text{quant}}(\mathbb{Q}x_1 \ldots \mathbb{Q}x_n\ ((\texttt{LTLtoFO}(\psi))[\sigma])). \tag{11}$$

Given a trace assignment, Π, and $k \in \mathbb{N}$ we denote Π_k any assignment satisfying $q \in \Pi_k(\pi_q)[i]$ iff $q \in \Pi(\pi_q)[i+k]$, and otherwise, $\Pi_k(\pi) = (\Pi(\pi))[k \ldots]$. We prove in the extended version, using the equivalence between LTL and FO[<] [9] and Prop. 19, that for all HyperQPTL formula $\varphi = \mathbb{Q}x_1 \ldots \mathbb{Q}x_n\ \psi$ where $x_i \in \mathcal{V}_{\mathbb{T}} \cup \mathcal{X}$, for $1 \leq i \leq n$, and $\mathbb{Q} \in \{\forall, \exists\}$: $(\Pi, \mathrm{T}, k) \models_{\texttt{HQ}} \varphi$ iff $(\overline{\mathrm{T}[k \ldots]}, (\Pi_k, \Pi^{\emptyset}_{\mathbb{N}})) \models_{\mathbb{T}} \texttt{tr}_H(\varphi)$, where $\Pi^{\emptyset}_{\mathbb{N}}$ is the empty time assignment.

For the translation from trace-prefixed hypertrace formulas to HyperQPTL, we use Lemma 5, and give a translation from flattened hypertrace formulas to HyperQPTL. As before we define a substitution $\sigma'' = \{X_a(\pi, i) \mapsto P_{a_\pi}(i) \mid a \in \mathcal{X}, \pi \in \mathcal{V}_{\mathbb{T}}, i \in \mathcal{V}_{\mathbb{N}}\} \cup \{X_q(\pi_q, i) \mapsto$

$P_q(i) \mid q \in \mathcal{X}, \pi \in \mathcal{V}_\mathbb{T}, i \in \mathcal{V}_\mathbb{N}\}$. The translation is defined as: $\mathtt{tr}_\mathbb{T}(\varphi) = \overrightarrow{\mathbb{Q}}(\mathtt{FOtoLTL}(\psi'[\sigma'']))$ with $\overrightarrow{\mathbb{Q}}\psi' = \mathtt{flatten}(\varphi, \mathcal{X}, \emptyset)$ where $\overrightarrow{\mathbb{Q}}$ is a sequence of constrained and unconstrained trace quantifiers and ψ' is a hypertrace formula with no trace quantifiers.

We prove that for all hypertrace formulas φ with only free trace variables, for all sets of traces T and assignments $\Pi_\mathbb{T}$: $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi^\emptyset_\mathbb{N})) \models_\mathbb{T} \varphi$ iff $(\Pi'_\mathbb{T}, \mathrm{T}, 0) \models_{\mathtt{HQ}} \mathtt{tr}_\mathbb{T}(\varphi)$, where $\Pi'_\mathbb{T}(\pi_q) = (\Pi_\mathbb{T}(\pi_q)[0] \cap \{q\})(\Pi_\mathbb{T}(\pi_q)[1] \cap \{q\}) \ldots$. For the base case, where ψ' has no trace quantifiers and all time quantifiers are bounded: $(\overline{\mathrm{T}}, (\Pi_\mathbb{T}, \Pi^\emptyset_\mathbb{N})) \models_\mathbb{T} \psi'$ iff $(\overline{\langle \Pi_\mathbb{T} \rangle}, \Pi^\emptyset_\mathbb{N}) \models \psi'[\sigma']$, follows from an analogous proof from the previous case. As the translation from $\Pi_\mathbb{T}$ to $\Pi'_\mathbb{T}$ does not change the valuation of q in the trace assigned to π_q and by [9], $(\overline{\langle \Pi_\mathbb{T} \rangle}, \Pi^\emptyset_\mathbb{N}) \models \psi'[\sigma']$ iff $\langle \Pi'_\mathbb{T} \rangle \models_{\mathtt{LTL}} \mathtt{FOtoLTL}(\psi'[\sigma'])$. By Proposition 19, it is equivalent to $(\Pi'_\mathbb{T}, \mathrm{T}, 0) \models_{\mathtt{HQ}} \mathtt{FOtoLTL}(\psi'[\sigma'])$. The induction cases (i.e., constrained and unconstrained quantifiers) follow from induction hypothesis and definitions. ◀

6 Time-prefixed Hypertrace Logic

We consider now time-prefixed hypertrace logic, with formulas defined by the grammar:

$$\varphi ::= \exists i\ \varphi \mid \neg\varphi \mid \psi \qquad \psi ::= \exists \pi\ \psi \mid \exists \pi :: T\ \psi \mid \psi \vee \psi \mid \neg\psi \mid i < i \mid i = i \mid X(\pi, i)$$

where π is a trace variable, i is a time variable and X is a binary predicate. To the best of our knowledge, there is no formalism in the literature allowing to specify hyperproperties by quantifying over time before traces, while supporting arbitrary time and trace quantifiers. The relative expressiveness between the trace-prefix and time-prefix fragments, however, remains an open problem.

As for the previously studied fragments, we use Theorem 12, and Corollary 8 to identify a fragment of the time-prefixed hypertrace logic with decidable satisfiability problem.

▶ **Corollary 21.** *Let* $\varphi = \overrightarrow{\mathbb{E}_{\mathbb{N}_<}}\ \exists \pi_1 :: T \ldots \exists \pi_n :: T\ \forall \pi'_1 :: T \ldots \forall \pi'_m :: T\ \overrightarrow{\mathbb{Q}_\mathbb{T}}\ \varphi_{qf}$ *be a hypertrace formula in prenex normal form s.t.* $\overrightarrow{\mathbb{E}_{\mathbb{N}_<}}$ *is a sequence of existential time quantifiers,* $\overrightarrow{\mathbb{Q}_\mathbb{T}}$ *is any combination of time and unconstrained trace quantifiers, respectively, and* φ_{qf} *is a quantifier-free. It is decidable to check whether* $[\![\varphi]\!] \neq \emptyset$.

In our theorem below, we prove that the satisfiability problem for arbitrary time-prefixed formulas is undecidable. We prove our result with a reduction from the (non)-halting problem for 2-counter Minsky machines.

A *2-counter Minsky machine* is defined by a tuple $\mathcal{M} = (Q, \Delta, \hat{q})$ where Q is a finite set of states with $\hat{q} \in Q$ being the initial state, and $\Delta \subseteq Q \times \{1, 2\} \times \{\mathtt{inc}, \mathtt{dec}, \mathtt{isZero}\} \times Q$ being the transition relation. For all $n, n' \in \mathbb{N}$, we write: $n \xrightarrow{\mathtt{isZero}} n'$ iff $n = n' = 0$; $n \xrightarrow{\mathtt{inc}} n'$ iff $n' = n + 1$; and $n \xrightarrow{\mathtt{dec}} n'$ iff $n' = n - 1$. We observe that `dec` is only allowed when $n > 0$. A configuration is a tuple with a state and the current value of the two counters. Given one of the counters $c \in \{1, 2\}$ we refer to the other counter as $\bar{c} = 3 - c$. We say that there is a transition between two configurations (q, n_1, n_2) and (q', n'_1, n'_2) iff there exists $(q, c, op, q') \in \Delta$ s.t. $n_c \xrightarrow{op} n'_c$ and, for the other counter, $n_{\bar{c}} = n'_{\bar{c}}$. A computation is a sequence of configurations connected by transitions. It is undecidable to check whether an arbitrary 2-counter Minsky machine has an infinite computation [15].

▶ **Theorem 22.** *Let* $\mathbb{Q} \in \exists_{\mathbb{N}_<} \forall_{\mathbb{N}_<} \exists^2_{\mathbb{N}_<} \forall_{\mathbb{N}_<} \forall_\mathbb{T} :: T (\exists_\mathbb{T} :: T)^2 \exists_\mathbb{T}$ *and* $\varphi = \mathbb{Q}\varphi'$ *be a time-prefixed hypertrace formula where* φ' *is quantifier-free. It is undecidable to check whether* $[\![\varphi]\!] \neq \emptyset$.

Proof. In our reduction from the (non)-halting problem for 2-counter Minsky machines, each trace encodes one configuration and the transition relation for the next step of the computation. Traces include the following propositional variables to capture configurations, for a given set of states Q: $\mathcal{X}_{\text{config}} = Q \cup \{\texttt{mem}_1, \texttt{mem}_2\}$ with $\texttt{mem}_1$ and $\texttt{mem}_2$ encoding the current value in their respective counters. We use $\mathcal{X}_{\text{transition}} = \{q_{\text{to}}, q_{\text{from}} \mid q \in Q\} \cup \{\texttt{inc}, \texttt{dec}, \texttt{isZero}, \texttt{to}_1, \texttt{to}_2\}$ to specify transitions where $\texttt{to}_1$ and $\texttt{to}_2$ represent the counters to be updated. The values in each counter are represented in unary and so incrementing the counter c amounts to making $\texttt{mem}_i$ true for one more step, while decrementing removes the last true valuation. We use an unconstrained trace quantifier to guess the time point where this increment or decrement takes place. This guess is guided by two propositional variables: $\mathcal{X}_{\text{guess}} = \{\texttt{guess}, \texttt{guessed}\}$. Hence, our traces are defined over $\mathcal{X} = \mathcal{X}_{\text{config}} \cup \mathcal{X}_{\text{transition}} \cup \mathcal{X}_{\text{guess}}$.

We start by defining useful formulas to encode requirements of well-formed traces, where $\texttt{exactlyOne}(\mathcal{Y})$ is true when only one of the propositions in $\mathcal{Y}$ holds:

- At each time i the trace π defines an unique transition:

$$\texttt{singleTr}(Q, \pi, i) \stackrel{\text{def}}{=} \texttt{exactlyOne}(\{q_{\text{to}}(\pi, i) \mid q \in Q\}) \wedge \texttt{exactlyOne}(\{q_{\text{from}}(\pi, i) \mid q \in Q\}) \wedge \texttt{exactlyOne}(\{\texttt{to}_1(\pi, i), \texttt{to}_2(\pi, i)\}) \wedge \texttt{exactlyOne}(\{\texttt{inc}(\pi, i), \texttt{dec}(\pi, i), \texttt{isZero}(\pi, i)\}) \tag{12}$$

- for times i and j the trace π defines the same transition:

$$\texttt{sameTr}(Q, \pi, i, j) \stackrel{\text{def}}{=} \bigwedge_{q \in Q} (q_{\text{to}}(\pi, i) \leftrightarrow q_{\text{to}}(\pi, j)) \wedge \bigwedge_{q \in Q} (q_{\text{from}}(\pi, i) \leftrightarrow q_{\text{from}}(\pi, j)) \wedge \bigwedge_{c \in \{1,2\}} (\texttt{to}_c(\pi, i) \leftrightarrow \texttt{to}_c(\pi, j)) \wedge \bigwedge_{op \in \{\texttt{inc}, \texttt{dec}, \texttt{isZero}\}} (op(\pi, i) \leftrightarrow op(\pi, j)) \tag{13}$$

- transition matches a transition from Δ:

$$\texttt{validTr}(\Delta, \pi, i) \stackrel{\text{def}}{=} \bigvee_{(q, c, op, q') \in \Delta} (q_{\text{from}}(\pi, i)) \wedge \texttt{to}_c(\pi, i)) \wedge op(\pi, i)) \wedge q'_{\text{to}}(\pi, i))) \tag{14}$$

- for times i and j the trace π defines the same unique state:

$$\texttt{sameState}(Q, \pi, i, j) \stackrel{\text{def}}{=} (\bigwedge_{q \in Q} q(\pi, i) \leftrightarrow q(\pi, j)) \wedge \texttt{exactlyOne}(\{q(\pi, i) \mid q \in Q\}) \tag{15}$$

- states in π and π' match the configuration:

$$\texttt{goodStates}(Q, \pi, \pi', i) \stackrel{\text{def}}{=} \bigvee_{q \in Q} (q_{\text{from}}(\pi, i) \leftrightarrow q(\pi, i)) \wedge \bigvee_{q \in Q} (q_{\text{to}}(\pi, i) \leftrightarrow q(\pi', i)) \tag{16}$$

- Only one (the first) guess can affect the evaluation:

$$\texttt{stopMulitpleGuess}(\pi_g, i, j) \stackrel{\text{def}}{=} (\texttt{guess}(\pi_g, i) \vee \texttt{guessed}(\pi_g, i)) \rightarrow \texttt{guessed}(\pi_g, j) \tag{17}$$

We recall that operations update at most one of the counters. To guess for the time point where this change occurs, our encoding uses propositional variables guess and guessed to be instantiated by an unconstrained trace quantifier. In particular, if either the guess is false or guessed is true, we require the two traces being compared to have equal values in their counters. We define now how to encode the effect of an operation, where π encodes the transition to be applied as well as the incoming configuration, π' encodes the outgoing configuration and π_g the unconstrained trace with the guess:

$$\texttt{op}(\pi, c, \pi', \pi_g, i, i_+, i_-) \stackrel{\text{def}}{=} (\texttt{mem}_{\overline{c}}(\pi, i) \leftrightarrow \texttt{mem}_{\overline{c}}(\pi', i) \wedge \tag{18}$$

$$(\texttt{isZero}(\pi, i) \rightarrow (\neg\texttt{mem}_c(\pi, i) \wedge \neg\texttt{mem}_c(\pi', i))) \wedge \tag{19}$$

$$((\neg\texttt{guess}(\pi_g, i) \vee \texttt{guessed}(\pi_g, i)) \rightarrow (\texttt{mem}_c(\pi, i) \leftrightarrow \texttt{mem}_c(\pi', i))) \wedge \tag{20}$$

$$((\neg \texttt{guessed}(\pi_g, i) \wedge \texttt{guess}(\pi_g, i) \wedge \texttt{inc}(\pi, i)) \rightarrow (\neg \texttt{mem}_c(\pi, i) \wedge \texttt{mem}_c(\pi', i) \wedge \neg \texttt{mem}_c(\pi', i_+)) \wedge \quad (21)$$

$$((\neg \texttt{guessed}(\pi_g, i) \wedge \texttt{guess}(\pi_g, i) \wedge \texttt{dec}(\pi, i)) \rightarrow (\texttt{mem}_c(\pi, i) \wedge \neg \texttt{mem}_c(\pi', i) \wedge \texttt{mem}_c(\pi', i_-)) \quad (22)$$

We include the time just before i, i_-, and just after, i_+, because time quantification will precede the quantification of π' and, so, we may get a different π' at each time point. In the time-prefixed formula defined next, we encode the requirement that all true valuations of $\texttt{mem}_1$ and $\texttt{mem}_2$ must be consecutive, starting from the beginning of the trace. Hence, by using i_- and i_+ we ensure that the change around time i is consistent with the operation leading to it. We combine all requirements above in the following time-prefixed hypertrace logic formula $\varphi_{\mathcal{M}}$ for a given machine $\mathcal{M} = (Q, \Delta, \hat{q})$:

$$\exists i_0 \forall i \; \exists i_+ \; \exists i_- \; \forall j \; \forall \pi :: T \; \exists \pi' :: T \; \exists \pi_{\hat{q}} :: T \; \exists \pi_g \big(\quad (23)$$

$$(i_0 \leq j \wedge i_- \leq i \wedge i < i_+ \wedge (i < j \rightarrow i_+ \leq j) \wedge (i_- = i \rightarrow i = i_0) \wedge (j < i \rightarrow j \leq i_-)) \rightarrow \quad (24)$$

$$(\texttt{mem}_1(\pi, i_+) \rightarrow \texttt{mem}_1(\pi, i)) \wedge (\texttt{mem}_2(\pi, i_+) \rightarrow \texttt{mem}_2(\pi, i)) \; \wedge \quad (25)$$

$$\hat{q}(\pi_{\hat{q}}, i) \wedge (\hat{q}(\pi, i) \rightarrow (\neg \texttt{mem}_1(\pi, i) \wedge \neg \texttt{mem}_2(\pi, i))) \; \wedge \quad (26)$$

$$\texttt{stopMulitpleGuess}(\pi_g, i, i_+) \; \wedge \quad (27)$$

$$\texttt{singleTr}(Q, \pi, i) \wedge \texttt{sameTr}(Q, \pi, i, i_+) \wedge \texttt{validTr}(\Delta, \pi, i) \; \wedge \quad (28)$$

$$\texttt{sameState}(Q, \pi, i, i_+) \wedge \texttt{goodStates}(Q, \pi, \pi', i) \; \wedge \quad (29)$$

$$(\texttt{to}_1(\pi, i) \rightarrow \texttt{op}(\pi, 1, \pi', \pi_g, i, i_+, i_-)) \wedge (\texttt{to}_2(\pi, i) \rightarrow \texttt{op}(\pi, 2, \pi', \pi_g, i, i_+, i_-))\big) \quad (30)$$

A set of traces $\mathrm{T} \models_{\mathbb{T}} \varphi_{\mathcal{M}}$ satisfies the following properties, where for $\mathcal{Y} \subseteq \mathcal{X}$ we define its projection over a trace as $\tau|_{\mathcal{Y}} = (\tau[0] \cap \mathcal{Y})(\tau[1] \cap \mathcal{Y}) \ldots$:

- exists $\tau \in \mathrm{T}$ and $(\hat{q}, c, op, q') \in \Delta$ s.t. $\tau|_{\{\hat{q}, \hat{q}_{\text{from}}, \texttt{to}_c, op, q'_{\text{to}}, \texttt{mem}_1, \texttt{mem}_2\}} = \{\hat{q}, \hat{q}_{\text{from}}, \texttt{to}_c, op, q'_{\text{to}}\}^{\omega}$ encoding the initial state where both counters are 0;
- for all $\tau \in \mathrm{T}$, $\tau|_{\{\texttt{mem}_1\}} = \{\texttt{mem}_1\}^{n_1}\{\}^{\omega}$, $\tau|_{\{\texttt{mem}_2\}} = \{\texttt{mem}_2\}^{n_2}\{\}^{\omega}$ and there exists $q \in Q$ and $(q, c, op, q') \in \Delta$ s.t. $\tau|_{\{q, q_{\text{from}}, \texttt{to}_c, op, q'_{\text{to}}\}} = \{q, q_{\text{from}}, \texttt{to}_c, op, q'_{\text{to}}\}^{\omega}$, which are unique and denoted by $\text{config}(\tau) = (q, n_1, n_2)$ and $\text{trans}(\tau) = (q, c, op, q')$;
- for each trace assigned to π, there can be only one time point where for the unconstrained trace assigned to π_g the value of `guess` is true and `guessed` is false;
- the trace assigned to π' encodes a correct next state from the transition defined by π.

$\mathrm{T} \models_{\mathbb{T}} \varphi_{\mathcal{M}}$, iff T has a trace encoding the initial state of $\mathcal{M}$ and, starting from that trace we can define a valid infinite computation of $\mathcal{M}$ using only traces in T. ◀

7 Related Work

The first work to explore the connection between hyperlogics (specifically HyperLTL) and classical first-order reasoning is by Finkbeiner and Zimmermann [8]. They extend FO[<] with the *equal-level* predicate E, which allows comparisons between positions at the same time point across different traces. They also use the bounded-promptness (also referred to as bounded-termination) property from [4] to show that $FO[<, E]$ is strictly more expressive than HyperLTL. Later, Bartocci et al. introduce hypertrace logic [1], which takes a different approach by extending FO[<] to a two-sorted logic, explicitly distinguishing between sorts for time and (constrained) traces. They prove that $FO[<, E]$ and hypertrace logic (with only constrained trace quantifiers) are equivalent. More recently, Beutner and Finkbeiner, in [2], presented a tool to check the satisfiability of hyperproperties specified in HyperLTL by translating them to first-order logic. Their translation follows the same approach as hypertrace logic: it considers separate sorts for traces and time. Other works have explore reasoning about hyperproperties using second-order quantification. For example, Hyper^{2}LTL [3] extends HyperLTL by introducing second-order quantification over the set of all possible traces, in

addition to first-order quantification over a given trace set. These approaches fall outside the scope of our work, as we focus on first-order definable languages.

The boundaries for the decidability of the satisfiability problem of HyperLTL were first investigated in [6]. These results were extended, in [14], to understand further which HyperLTL fragments have a decidable satisfiability checking. In [5], Coenen et al. give a complete picture of the hierarchy of hyperlogics based on extensions of temporal logics. In particular, they compare and summarize results for HyperLTL and HyperQPTL. A different line of work looks into LTL interpretation with team semantics, called TeamLTL. Team semantics generalizes classical first-order logic by evaluating formulas over sets of assignments, called teams, rather than single assignments. TeamLTL, based on this framework, does not use explicit trace quantifiers. Instead, it adopts the modular approach of team semantics, where reasoning over multiple traces is introduced implicitly through the use of generalized atoms. In [20], the authors establish the relation between different extensions of TeamLTL to fragments of HyperQPTL$^+$, which generalizes HyperQPTL to include non-uniform quantification over propositional variables.

8 Conclusion

In this work, we introduced an extension of hypertrace logic with unconstrained trace quantifiers, effectively extending its expressive power. We established expressiveness results connecting fragments of hypertrace logic to well-known temporal logics: unconstrained hypertrace logic is equivalent to S1S, while trace-prefixed hypertrace logic corresponds to HyperQPTL. We also identified a decidable fragment of the logic, where constrained trace quantifier alternation is restricted to a single existential-to-universal switch. This generalizes a known decidability result for HyperQPTL by allowing arbitrary placements of time quantifiers. Finally, we identified a fragment of time-prefixed hypertrace logic with undecidable satisfiability checking. For future work, it remains open how the trace- and time-prefixed fragments relate in terms of expressiveness. Additionally, we plan to explore the connection between hyperlogics and classical first-order reasoning to enable the transfer of techniques and results across these areas.

References

1 Ezio Bartocci, Thomas Ferrère, Thomas A. Henzinger, Dejan Nickovic, and Ana Oliveira da Costa. Flavors of sequential information flow. In Bernd Finkbeiner and Thomas Wies, editors, *Verification, Model Checking, and Abstract Interpretation (VMCAI)*, pages 1–19. Springer International Publishing, 2022. doi:10.1007/978-3-030-94583-1_1.

2 Raven Beutner and Bernd Finkbeiner. Checking satisfiability of hyperproperties using first-order logic. In S. Akshay, Aina Niemetz, and Sriram Sankaranarayanan, editors, *Automated Technology for Verification and Analysis (ATVA)*, pages 198–211, Cham, 2025. Springer Nature Switzerland. doi:10.1007/978-3-031-78750-8_10.

3 Raven Beutner, Bernd Finkbeiner, Hadar Frenkel, and Niklas Metzger. Second-order hyperproperties. In Constantin Enea and Akash Lal, editors, *Computer Aided Verification (CAV)*, pages 309–332, Cham, 2023. Springer Nature Switzerland. doi:10.1007/978-3-031-37703-7_15.

4 Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Unifying hyper and epistemic temporal logics. In *International Conference on Foundations of Software Science and Computation Structures*, pages 167–182. Springer, 2015.

5 Norine Coenen, Bernd Finkbeiner, Christopher Hahn, and Jana Hofmann. The hierarchy of hyperlogics. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, 2019. doi:10.1109/LICS.2019.8785713.

6 Bernd Finkbeiner and Christopher Hahn. Deciding Hyperproperties. In Josée Desharnais and Radha Jagadeesan, editors, *27th International Conference on Concurrency Theory (CONCUR 2016)*, volume 59 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:14, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CONCUR.2016.13.

7 Bernd Finkbeiner, Christopher Hahn, Jana Hofmann, and Leander Tentrup. Realizing ω-regular hyperproperties. In Shuvendu K. Lahiri and Chao Wang, editors, *Computer Aided Verification (CAV)*, pages 40–63, Cham, 2020. Springer International Publishing.

8 Bernd Finkbeiner and Martin Zimmermann. The First-Order Logic of Hyperproperties. In Heribert Vollmer and Brigitte Vallée, editors, *34th Symposium on Theoretical Aspects of Computer Science (STACS 2017)*, volume 66 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 30:1–30:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2017.30.

9 Dov Gabbay, Amir Pnueli, Saharon Shelah, and Jonathan Stavi. On the temporal analysis of fairness. In *Proceedings of the 7th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 163–173, 1980.

10 Christopher Hahn. *Logical and deep learning methods for temporal reasoning*. 2021. doi:http://dx.doi.org/10.22028/D291-35192.

11 Maurice P. Herlihy and Jeannette M. Wing. Linearizability: a correctness condition for concurrent objects. *ACM Trans. Program. Lang. Syst.*, 12(3):463–492, July 1990. doi:10.1145/78969.78972.

12 Hans Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, UCLA, 1968.

13 Yonit Kesten and Amir Pnueli. Complete proof system for qptl. *Journal of Logic and Computation*, 12(5):701–745, 10 2002. doi:10.1093/logcom/12.5.701.

14 Corto Mascle and Martin Zimmermann. The Keys to Decidable HyperLTL Satisfiability: Small Models or Very Simple Formulas. In Maribel Fernández and Anca Muscholl, editors, *28th EACSL Annual Conference on Computer Science Logic (CSL 2020)*, volume 152 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 29:1–29:16, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CSL.2020.29.

15 Marvin L. Minsky. *Computation: finite and infinite machines*. Prentice-Hall, Inc., USA, 1967.

16 Amir Pnueli. The temporal logic of programs. In *Proc. of FOCS77: the 18th Annual Symposium on Foundations of Computer Science*, pages 46–57. IEEE Computer Society, 1977. doi:10.1109/SFCS.1977.32.

17 Markus N. Rabe. *A temporal logic approach to information-flow control*. 2016. doi:http://dx.doi.org/10.22028/D291-26650.

18 Gaëtan Regaud and Martin Zimmermann. The complexity of hyperqptl. *arXiv preprint arXiv:2412.07341*, 2024.

19 J. Richard Büchi. Symposium on decision problems: On a decision method in restricted second order arithmetic. In Ernest Nagel, Patrick Suppes, and Alfred Tarski, editors, *Logic, Methodology and Philosophy of Science*, volume 44 of *Studies in Logic and the Foundations of Mathematics*, pages 1–11. Elsevier, 1960. doi:https://doi.org/10.1016/S0049-237X(09)70564-6.

20 Jonni Virtema, Jana Hofmann, Bernd Finkbeiner, Juha Kontinen, and Fan Yang. Linear-Time Temporal Logic with Team Semantics: Expressivity and Complexity. In Mikołaj Bojańczyk and Chandra Chekuri, editors, *41st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2021)*, volume 213 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 52:1–52:17, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.FSTTCS.2021.52.

9 Full Proofs

Theorem 20. *For all* HyperQPTL *sentences* φ_H *there exists a trace-prefixed hypertrace sentence* φ *such that for all sets of infinite traces* $T \subseteq (\mathbf{2}^{\mathcal{X}})^\omega$, $T \models_{\texttt{HQ}} \varphi_H$ *iff* $T \models_{\mathbb{T}} \varphi$. *For all trace-prefixed hypertrace sentences* φ *there exists a* HyperQPTL *sentence* φ_H *such that for all sets of infinite traces* $T \subseteq (\mathbf{2}^{\mathcal{X}})^\omega$, $T \models_{\texttt{HQ}} \varphi_H$ *iff* $T \models_{\texttt{QPTL}} \varphi$.

Proof. We denote by $\texttt{LTLtoFO}(\psi)$ and $\texttt{FOtoLTL}(\psi)$ the translation from LTL formulas to FO[<] and vice-versa given by LTL and FO[<] being equivalent [9].

We start with the translation from HyperQPTL formulas to an equivalent trace-prefixed hypertrace formula. We first define the translation of $\varphi \in$ HyperQPTL for its different quantifiers:

$$\texttt{tr}_H^{\text{quant}}(\varphi) = \begin{cases} \varphi & \text{if } \varphi \text{ is quantifier-free} \\ \mathbb{Q}\pi{::}T\ \texttt{tr}_H^{\text{quant}}(\varphi') & \text{if } \varphi = \mathbb{Q}\pi\ \varphi', \mathbb{Q} \in \{\forall, \exists\} \text{ and } \pi \in \mathcal{V}_{\mathbb{T}} \\ \mathbb{Q}\pi_q\ \texttt{tr}_H^{\text{quant}}(\varphi') & \text{if } \text{ if } \varphi = \mathbb{Q}q\ \varphi', \mathbb{Q} \in \{\forall, \exists\} \text{ and } q \in \mathcal{X} \end{cases} \tag{31}$$

With $\texttt{LTLtoFO}(\psi)$ all propositional variables a_π in ψ are mapped to $P_{a_\pi}(i)$, while all q are mapped to $P_q(i)$. We define the substitution from these predicates to the binary predicates of FO[$<, \mathbb{T}$]: $\sigma = \{P_{a_\pi}(i) \mapsto X_a(\pi, i) \mid a \in \mathcal{X}, \pi \in \mathcal{V}_{\mathbb{T}}, i \in \mathcal{V}_{\mathbb{N}}\} \cup \{P_q(i) \mapsto X_q(\pi_q, i) \mid q \in \mathcal{X}, \pi \in \mathcal{V}_{\mathbb{T}}, i \in \mathcal{V}_{\mathbb{N}}\}$. The translation from HyperQPTL formulas to $\mathbf{T}$-FO[$<, \mathbb{T}$] is defined below, where $\psi \in$ LTL, $x_i \in \mathcal{V}_{\mathbb{T}} \cup \mathcal{X}$, for $1 \leq i \leq n$, and $\mathbb{Q} \in \{\forall, \exists\}$:

$$\texttt{tr}_H(\mathbb{Q}x_1 \ldots \mathbb{Q}x_n\ \psi) = \texttt{tr}_H^{\text{quant}}(\mathbb{Q}x_1 \ldots \mathbb{Q}x_n\ ((\texttt{LTLtoFO}(\psi))[\sigma])). \tag{32}$$

Given a trace assignment, Π, and $k \in \mathbb{N}$ we denote Π_k any assignment satisfying $q \in \Pi_k(\pi_q)[i]$ iff $q \in \Pi(\pi_q)[i + k]$, and otherwise, $\Pi_k(\pi) = (\Pi(\pi))[k \ldots]$. We want to prove that given a HyperQPTL formula $\varphi = \mathbb{Q}x_1 \ldots \mathbb{Q}x_n\ \psi$ where $x_i \in \mathcal{V}_{\mathbb{T}} \cup \mathcal{X}$, for $1 \leq i \leq n$, and $\mathbb{Q} \in \{\forall, \exists\}$: $(\Pi, \mathrm{T}, k) \models_{\texttt{HQ}} \varphi$ iff $(\overline{\mathrm{T}[k \ldots]}, (\Pi_k, \Pi_{\mathbb{N}}^{\emptyset})) \models_{\mathbb{T}} \texttt{tr}_H(\varphi)$, where $\Pi_{\mathbb{N}}^{\emptyset}$ is the empty time assignment.

We proceed by induction on the structure of the formula, with the base case being the (longest) quantifier-free formula, $\psi \in$ LTL. By Proposition 19 and equivalence of LTL to FO[<] [9], $(\Pi, \mathrm{T}, k) \models_{\texttt{HQ}} \psi$ iff $\langle \Pi \rangle[k \ldots] \models \texttt{LTLtoFO}(\psi)$. Let Π'_k be the trace assignment defined from $\langle \Pi \rangle[k \ldots]$ as follows: $a \in \Pi'_k(\pi)[i]$ iff $a_\pi \in \langle \Pi \rangle[i + k]$ and $q \in \Pi'_k(\pi_q)[i]$ iff $q \in \langle \Pi \rangle[i + k]$ for all $i \in \mathbb{N}$, trace variables $\pi \in \mathcal{V}$ and propositional variables $a, q \in \mathcal{X}$. We prove by induction on FO[<] formulas that $(\overline{\langle \Pi \rangle[k \ldots]}, \Pi_{\mathbb{N}}) \models \texttt{LTLtoFO}(\psi)$ iff $(\overline{\mathrm{T}[k \ldots]}, (\Pi'_k, \Pi_{\mathbb{N}})) \models_{\mathbb{T}} (\texttt{LTLtoFO}(\psi))[\sigma']$, for any trace assignments Π, time assignments $\Pi_{\mathbb{N}}$, set of traces T and LTL formula ψ. We observe that σ' changes only predicates, thus, we need only to prove the base cases, as the induction cases follow from definitions and induction hypothesis. For the first base case, $(\overline{\langle \Pi \rangle[k \ldots]}, \Pi_{\mathbb{N}}) \models X_{a_\pi}(i)$ iff $a_\pi \in (\langle \Pi \rangle[k \ldots])[\Pi_{\mathbb{N}}(i)]$ iff $a_\pi \in \langle \Pi \rangle[k + \Pi_{\mathbb{N}}(i)]$. By definition of Π'_k, this is equivalent to $a \in \Pi'_k(\pi)[\Pi_{\mathbb{N}}(i)]$, and so, $(\overline{\mathrm{T}[k \ldots]}, (\Pi'_k, \Pi_{\mathbb{N}})) \models_{\mathbb{T}} X_a(\pi, i)$. The other base case, $(\overline{\langle \Pi \rangle[k \ldots]}, \Pi_{\mathbb{N}}) \models X_q(i)$ is analogous.

Let's consider now the induction case, $\exists \pi\ \varphi$. We assume that $(\Pi, \mathrm{T}, k) \models_{\texttt{HQ}} \exists \pi\ \varphi$, that is, exists $\tau \in \mathrm{T}$ s.t. $(\Pi[\pi \mapsto \tau]\mathrm{T}, k) \models_{\texttt{HQ}} \varphi$. By induction hypothesis, $(\overline{\mathrm{T}[k \ldots]}, (\Pi[\pi \mapsto \tau]_k, \Pi_{\mathbb{N}}^{\emptyset})) \models_{\mathbb{T}} \varphi$ and so $(\overline{\mathrm{T}[k \ldots]}, (\Pi_k, \Pi_{\mathbb{N}}^{\emptyset})) \models_{\mathbb{T}} \exists \varphi{::}T\varphi$. The other quantifier is analogous. Hence, for all HyperQPTL formulas φ_H there exists the trace-prefixed hypertrace formula $\mathrm{tr}_H(\varphi_H)$ s.t. $\mathrm{T} \models_{\texttt{HQ}} \varphi_H$ iff $\mathrm{T} \models_{\mathbb{T}} \mathrm{tr}_H(\varphi_H)$.

For the translation from trace-prefixed hypertrace formulas to HyperQPTL, we use Lemma 5, and give a translation from flattened hypertrace formulas to HyperQPTL. As before we define a substitution $\sigma'' = \{X_a(\pi, i) \mapsto P_{a_\pi}(i) \mid a \in \mathcal{X}, \pi \in \mathcal{V}_{\mathbb{T}}, i \in \mathcal{V}_{\mathbb{N}}\} \cup \{X_q(\pi_q, i) \mapsto P_q(i) \mid q \in \mathcal{X}, \pi \in \mathcal{V}_{\mathbb{T}}, i \in \mathcal{V}_{\mathbb{N}}\}$. The translation is defined as: $\texttt{tr}_{\mathbb{T}}(\varphi) = \overrightarrow{\mathbb{Q}}(\texttt{FOtoLTL}(\psi'[\sigma'']))$

with $\overrightarrow{\mathbb{Q}}\psi' = \texttt{flatten}(\varphi, \mathcal{X}, \emptyset)$ where $\overrightarrow{\mathbb{Q}}$ is a sequence of constrained and unconstrained trace quantifiers and ψ' is a hypertrace formula with no trace quantifiers.

We prove that for all hypertrace formulas φ with only free trace variables, for all sets of traces T and assignments $\Pi_{\mathbb{T}}$: $(\overline{\mathrm{T}}, (\Pi_{\mathbb{T}}, \Pi^{\emptyset}_{\mathbb{N}})) \models_{\mathbb{T}} \varphi$ iff $(\Pi'_{\mathbb{T}}, \mathrm{T}, 0) \models_{\texttt{HQ}} \texttt{tr}_{\mathbb{T}}(\varphi)$, where $\Pi'_{\mathbb{T}}(\pi_q) = (\Pi_{\mathbb{T}}(\pi_q)[0] \cap \{q\})(\Pi_{\mathbb{T}}(\pi_q)[1] \cap \{q\}) \ldots$. Base-case, where ψ' has no trace quantifiers and all time quantifiers are bounded: $(\overline{\mathrm{T}}, (\Pi_{\mathbb{T}}, \Pi^{\emptyset}_{\mathbb{N}})) \models_{\mathbb{T}} \psi'$ iff $(\overline{\langle \Pi_{\mathbb{T}} \rangle}, \Pi^{\emptyset}_{\mathbb{N}}) \models \psi'[\sigma']$, follows from an analogous proof from the previous case. As the translation from $\Pi_{\mathbb{T}}$ to $\Pi'_{\mathbb{T}}$ does not change the valuation of q in the trace assigned to π_q and by [9], $(\overline{\langle \Pi_{\mathbb{T}} \rangle}, \Pi^{\emptyset}_{\mathbb{N}}) \models \psi'[\sigma']$ iff $\langle \Pi'_{\mathbb{T}} \rangle \models_{\texttt{LTL}} \texttt{FOtoLTL}(\psi'[\sigma'])$. By Proposition 19, it is equivalent to $(\Pi'_{\mathbb{T}}, \mathrm{T}, 0) \models_{\texttt{HQ}} \texttt{FOtoLTL}(\psi'[\sigma'])$. The induction cases (i.e., constrained and unconstrained quantifiers) follow from induction hypothesis and definitions. ◀