Team Semantics for the Specification and Verification of Hyperproperties

Andreas Krebs
Universität Tübingen, Wilhelm-Schickard-Institut für Informatik, Tübingen, Germany
krebs@informatik.uni-tuebingen.de

Arne Meier[1]
Leibniz Universität Hannover, Institut für Theoretische Informatik, Hannover, Germany
meier@thi.uni-hannover.de
https://orcid.org/0000-0002-8061-5376

Jonni Virtema
Hasselt University, Databases and Theoretical Computer Science Group, Diepenbeek, Belgium
jonni.virtema@uhasselt.be
https://orcid.org/0000-0002-1582-3718

Martin Zimmermann[2]
Saarland University, Reactive Systems Group, Saarbrücken, Germany
zimmermann@react.uni-saarland.de

Abstract

We develop team semantics for Linear Temporal Logic (LTL) to express hyperproperties, which have recently been identified as a key concept in the verification of information flow properties. Conceptually, we consider an asynchronous and a synchronous variant of team semantics. We study basic properties of this new logic and classify the computational complexity of its satisfiability, path, and model checking problem. Further, we examine how extensions of these basic logics react on adding other atomic operators. Finally, we compare its expressivity to the one of HyperLTL, another recently introduced logic for hyperproperties. Our results show that LTL under team semantics is a viable alternative to HyperLTL, which complements the expressivity of HyperLTL and has partially better algorithmic properties.

2012 ACM Subject Classification Theory of computation → Complexity theory and logic

Keywords and phrases LTL, Hyperproperties, Team Semantics, Model Checking, Satisfiability

Digital Object Identifier 10.4230/LIPIcs.MFCS.2018.10

Related Version [21], `https://arxiv.org/abs/1709.08510`

Acknowledgements We thank Alexander Weinert for numerous fruitful discussions.

1 Introduction

Guaranteeing security and privacy of user information is a key requirement in software development. However, it is also one of the hardest goals to accomplish. One reason for this difficulty is that such requirements typically amount to reasoning about the flow of information and relating different execution traces of the system. In particular, these

[1] Funded by the German Research Foundation DFG, project ME 4279/1-2.
[2] Funded by the German Research Foundation DFG, project "TriCS" (ZI 1516/1-1).

requirements are no longer trace properties, i.e., properties whose satisfaction can be verified by considering each trace in isolation. For example, the property "the system terminates eventually" is satisfied if every trace eventually reaches a final state. Formally, a trace property φ is a set of traces and a system satisfies φ if each of its traces is in φ.

In contrast, the property "the system terminates within a bounded amount of time" is no longer a trace property; consider a system that has a trace t_n for every n, so that t_n only reaches a final state after n steps. This system does not satisfy the bounded termination property, but each individual trace t_n could also stem from a system that does satisfy it. Thus, satisfaction of the property cannot be verified by considering each trace in isolation.

Properties with this characteristic were termed *hyperproperties* by Clarkson and Schneider [6]. Formally, a hyperproperty φ is a set of sets of traces and a system satisfies φ if its set of traces is contained in φ. The conceptual difference to trace properties allows specifying a much richer landscape of properties including information flow and trace properties. Further, one can also express specifications for symmetric access to critical resources in distributed protocols and Hamming distances between code words in coding theory [29]. However, the increase in expressiveness requires novel approaches to specification and verification.

HyperLTL. Trace properties are typically specified in temporal logics, most prominently in Linear Temporal Logic (LTL) [28]. Verification of LTL specifications is routinely employed in industrial settings and marks one of the most successful applications of formal methods to real-life problems. Recently, this work has been extended to hyperproperties: HyperLTL, LTL equipped with trace quantifiers, has been introduced to specify hyperproperties [5]. Accordingly, a model of a HyperLTL formula is a set of traces and the quantifiers range over these traces. This logic is able to express the majority of the information flow properties found in the literature (we refer to Section 3 of [5] for a full list). The satisfiability problem for HyperLTL is undecidable [10] while the model checking problem is decidable, albeit of non-elementary complexity [5, 13]. In view of this, the full logic is too strong. Fortunately most information flow properties found in the literature can be expressed with at most one quantifier alternation and consequently belong to decidable (and tractable) fragments. Further works have studied runtime verification [2, 11], connections to first-order logic [14], provided tool support [13, 10], and presented applications to "software doping" [7] and the verification of web-based workflows [12]. In contrast, there are natural properties, e.g., bounded termination, which are not expressible in HyperLTL (which is an easy consequence of a much stronger non-expressibility result [3]).

Team Semantics. Intriguingly, there exists another modern family of logics, *Dependence Logics* [32, 9], which operate as well on sets of objects instead of objects alone. Informally, these logics extend first-order logic (FO) by atoms expressing, e.g., that "the value of a variable x functionally determines the value of a variable y" or that "the value of a variable x is informationally independent of the value of a variable y". Obviously, such statements only make sense when being evaluated over a set of assignments. In the language of dependence logic, such sets are called *teams* and the semantics is termed *team semantics*.

In 1997, Hodges introduced compositional semantics for Hintikka's Independence-friendly logic [19]. This can be seen as the cornerstone of the mathematical framework of dependence logics. Intuitively, this semantics allows for interpreting a team as a database table. In this approach, variables of the table correspond to attributes and assignments to rows or records. In 2007, Väänänen [32] introduced his modern approach to such logics and adopted team semantics as a core notion, as dependence atoms are meaningless under Tarskian semantics.

After the introduction of dependence logic, a whole family of logics with different atomic statements have been introduced in this framework: *independence logic* [17] and *inclusion logic* [15] being the most prominent. Interest in these logics is rapidly growing and the research community aims to connect their area to a plethora of disciplines, e.g., linguistics [16], biology [16], game [4] and social choice theory [30], philosophy [30], and computer science [16]. We are the first to exhibit connections to formal languages via application of Büchi automata (see Theorem 4.3). Team semantics has also found their way into modal [33] and temporal logic [20], as well as statistics [8].

Recently, Krebs et al. [20] proposed team semantics for Computation Tree Logic (CTL), where a team consists of worlds of the transition system under consideration. They considered synchronous and asynchronous team semantics, which differ in how time evolves in the semantics of the temporal operators. They proved that satisfiability is EXPTIME-complete under both semantics while model checking is PSPACE-complete under synchronous semantics and P-complete under asynchronous semantics.

Our Contribution. The conceptual similarities between HyperLTL and team semantics raise the question how an LTL variant under team semantics relates to HyperLTL. For this reason, we develop team semantics for LTL, analyse the complexity of its satisfiability and model checking problems, and subsequently compare the novel logic to HyperLTL.

When defining the logic, we follow the approach of Krebs et al. [20] for defining team semantics for CTL: we introduce synchronous and asynchronous team semantics for LTL, where teams are now sets of traces. In particular, as a result, we have to consider potentially uncountable teams, while all previous work on model checking problems for logics under team semantics has been restricted to the realm of finite teams.

We prove that the satisfiability problem for team LTL is PSPACE-complete under both semantics, by showing that the problems are equivalent to LTL satisfiability under classical semantics. Generally, we observe that for the basic asynchronous variant all of our investigated problems trivially reduce to and from classical LTL semantics. However, for the synchronous semantics this is not the case for two variants of the model checking problem. As there are uncountably many traces, we have to represent teams, i.e., sets of traces, in a finitary manner. The path checking problem asks to check whether a finite team of ultimately periodic traces satisfies a given formula. As our main result, we establish this problem to be PSPACE-complete for synchronous semantics. In the (general) model checking problem, a team is represented by a finite transition system. Formally, given a transition system and a formula, the model checking problem asks to determine whether the set of traces of the system satisfies the formula. For the synchronous case we give a polynomial space algorithm for the model checking problem for the disjunction-free fragment, while we leave open the complexity of the general problem. Disjunction plays a special role in team semantics, as it splits a team into two. As a result, this operator is commonly called *splitjunction* instead of disjunction. In our setting, the splitjunction requires us to deal with possibly infinitely many splits of uncountable teams, if a splitjunction is under the scope of a G-operator, which raises interesting language-theoretic questions.

Further, we study the effects for complexity that follow when our logics are extended by dependence atoms and the contradictory negation. Finally, we show that LTL under team semantics is able to specify properties which are not expressible in HyperLTL and *vice versa*.

Recall that satisfiability for HyperLTL is undecidable and model checking of non-elementary complexity. Our results show that similar problems for LTL under team semantics have a much simpler complexity while some hyperproperties are still expressible (e.g., input

determinism, see page 11, or bounded termination). This proposes LTL under team semantics to be a significant alternative for the specification and verification of hyperproperties that complements HyperLTL.

2 Preliminaries

The non-negative integers are denoted by $\mathbb{N}$ and the power set of a set S is denoted by 2^S. Throughout the paper, we fix a finite set AP of atomic propositions.

Computational Complexity. We will make use of standard notions in complexity theory. In particular, we will use the complexity classes P and PSPACE. Most reductions used in the paper are $\leq_{\mathrm{m}}^{\mathrm{p}}$-reductions, that is, polynomial time, many-to-one reductions.

Traces. A *trace* over AP is an infinite sequence from $(2^{\mathrm{AP}})^\omega$; a finite trace is a finite sequence from $(2^{\mathrm{AP}})^*$. The length of a finite trace t is denoted by $|t|$. The empty trace is denoted by ε and the concatenation of two finite traces t_0 and t_1 by t_0t_1. Unless stated otherwise, a trace is always assumed to be infinite. A *team* is a (potentially infinite) set of traces.

Given a trace $t = t(0)t(1)t(2)\cdots$ and $i \geq 0$, we define $t[i,\infty) := t(i)t(i+1)t(i+2)\cdots$, which we lift to teams $T \subseteq (2^{\mathrm{AP}})^\omega$ by defining $T[i,\infty) := \{t[i,\infty) \mid t \in T\}$. A trace t is *ultimately periodic*, if it is of the form $t = t_0 \cdot t_1^\omega = t_0t_1t_1t_1\cdots$ for two finite traces t_0 and t_1 with $|t_1| > 0$. As a result, an ultimately periodic trace t is finitely represented by the pair (t_0, t_1); we define $[\![(t_0,t_1)]\!] = t_0t_1^\omega$. Given a set $\mathcal{T}$ of such pairs, we define $[\![\mathcal{T}]\!] = \{[\![(t_0,t_1)]\!] \mid (t_0,t_1) \in \mathcal{T}\}$, which is a team of ultimately periodic traces. We call $\mathcal{T}$ a team encoding of $[\![\mathcal{T}]\!]$.

Linear Temporal Logic. The formulas of Linear Temporal Logic (LTL) [28] are defined via the grammar $\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\varphi \mid \mathsf{F}\varphi \mid \mathsf{G}\varphi \mid \varphi\mathsf{U}\varphi \mid \varphi\mathsf{R}\varphi$, where p ranges over the atomic propositions in AP. The length of a formula is defined to be the number of Boolean and temporal connectives occurring in it. The length of an LTL formula is often defined to be the number of syntactically different subformulas, which might be exponentially smaller. Here, we need to distinguish syntactically equal subformulas which becomes clearer after defining the semantics (see also Example 2.1 afterwards on this). As we only consider formulas in negation normal form, we use the full set of temporal operators.

Next, we recall the classical semantics of LTL before we introduce team semantics. For traces $t \in (2^{\mathrm{AP}})^\omega$ we define

$$
\begin{aligned}
t &\models^{\mathrm{c}} p && \text{if } p \in t(0),\\
t &\models^{\mathrm{c}} \neg p && \text{if } p \notin t(0),\\
t &\models^{\mathrm{c}} \psi \wedge \varphi && \text{if } t \models^{\mathrm{c}} \psi \text{ and } t \models^{\mathrm{c}} \varphi,\\
t &\models^{\mathrm{c}} \psi \vee \varphi && \text{if } t \models^{\mathrm{c}} \psi \text{ or } t \models^{\mathrm{c}} \varphi,\\
t &\models^{\mathrm{c}} \mathsf{X}\varphi && \text{if } t[1,\infty) \models^{\mathrm{c}} \varphi,\\
t &\models^{\mathrm{c}} \mathsf{F}\varphi && \text{if } \exists k \geq 0 : t[k,\infty) \models^{\mathrm{c}} \varphi,\\
t &\models^{\mathrm{c}} \mathsf{G}\varphi && \text{if } \forall k \geq 0 : t[k,\infty) \models^{\mathrm{c}} \varphi,\\
t &\models^{\mathrm{c}} \psi\mathsf{U}\varphi && \text{if } \exists k \geq 0 : t[k,\infty) \models^{\mathrm{c}} \varphi \text{ and}\\
& && \quad \forall k' < k : t[k',\infty) \models^{\mathrm{c}} \psi,\\
t &\models^{\mathrm{c}} \psi\mathsf{R}\varphi && \text{if } \forall k \geq 0 : t[k,\infty) \models^{\mathrm{c}} \varphi \text{ or}\\
& && \quad \exists k' < k : t[k',\infty) \models^{\mathrm{c}} \psi.
\end{aligned}
$$

Team Semantics for LTL. Next, we introduce two variants of team semantics for LTL, which differ in their interpretation of the temporal operators: a synchronous semantics ($\models^{\mathrm{s}}$), where time proceeds in lockstep along all traces of the team, and an asynchronous semantics ($\models^{\mathrm{a}}$) in which, on each trace of the team, time proceeds independently. We write $\models^{\star}$ whenever

property	definition	$\models^{a}$	$\models^{s}$
empty team property	$\emptyset \models^{\star} \varphi$	✓	✓
downwards closure	$T \models^{\star} \varphi$ implies $\forall T' \subseteq T$: $T' \models^{\star} \varphi$	✓	✓
union closure	$T \models^{\star} \varphi$, $T' \models^{\star} \varphi$ implies $T \cup T' \models^{\star} \varphi$	✓	×
flatness	$T \models^{\star} \varphi$ if and only if $\forall t \in T$: $\{t\} \models^{\star} \varphi$	✓	×
singleton equivalence	$\{t\} \models^{\star} \varphi$ if and only if $t \models^{c} \varphi$	✓	✓

Figure 1 Structural properties overview.

a definition coincides for both semantics. For teams $T \subseteq (2^{\mathrm{AP}})^{\omega}$ let

$T \models^{\star} p$ if $\forall t \in T : p \in t(0)$,
$T \models^{\star} \neg p$ if $\forall t \in T : p \notin t(0)$,
$T \models^{\star} \psi \wedge \varphi$ if $T \models^{\star} \psi$ and $T \models^{\star} \varphi$,
$T \models^{\star} \psi \vee \varphi$ if $\exists T_1 \cup T_2 = T : T_1 \models^{\star} \psi$ and $T_2 \models^{\star} \varphi$,
$T \models^{\star} \mathsf{X}\varphi$ if $T[1, \infty) \models^{\star} \varphi$.

This concludes the cases where both semantics coincide. Next, we present the remaining cases for the synchronous semantics, which are inherited from the classical semantics of LTL.

$T \models^{s} \mathsf{F}\varphi$ if $\exists k \geq 0 : T[k, \infty) \models^{s} \varphi$,
$T \models^{s} \mathsf{G}\varphi$ if $\forall k \geq 0 : T[k, \infty) \models^{s} \varphi$,
$T \models^{s} \psi \mathsf{U} \varphi$ if $\exists k \geq 0 : T[k, \infty) \models^{s} \varphi$ and $\forall k' < k : T[k', \infty) \models^{s} \psi$, and
$T \models^{s} \psi \mathsf{R} \varphi$ if $\forall k \geq 0 : T[k, \infty) \models^{s} \varphi$ or $\exists k' < k : T[k', \infty) \models^{s} \psi$.

Finally, we present the remaining cases for the asynchronous semantics. Note that, here there is no unique timepoint k, but a timepoint k_t for every trace t, i.e., time evolves asynchronously between different traces.

$T \models^{a} \mathsf{F}\varphi$ if $\exists k_t \geq 0$, for each $t \in T$: $\{t[k_t, \infty) \mid t \in T\} \models^{a} \varphi$
$T \models^{a} \mathsf{G}\varphi$ if $\forall k_t \geq 0$, for each $t \in T$: $\{t[k_t, \infty) \mid t \in T\} \models^{a} \varphi$,
$T \models^{a} \psi \mathsf{U} \varphi$ if $\exists k_t \geq 0$, for each $t \in T$: $\{t[k_t, \infty) \mid t \in T\} \models^{a} \varphi$, and
$\forall k'_t < k_t$, for each $t \in T$: $\{t[k'_t, \infty) \mid t \in T\} \models^{a} \psi$, and
$T \models^{a} \psi \mathsf{R} \varphi$ if $\forall k_t \geq 0$, for each $t \in T$: $\{t[k_t, \infty) \mid t \in T\} \models^{a} \varphi$ or
$\exists k'_t < k_t$, for each $t \in T$: $\{t[k'_t, \infty) \mid t \in T\} \models^{a} \psi$.

We call expressions of the form $\psi \vee \varphi$ *splitjunctions* to emphasise on the team semantics where we split a team into two parts. Similarly, the $\vee$-operator is referred to as a *splitjunction*.

Let us illustrate the difference between synchronous and asynchronous semantics with an example involving the F operator. Similar examples can be constructed for the other temporal operators (but for X) as well.

▶ **Example 2.1.** Let $T = \{\{p\}\emptyset^{\omega}, \emptyset\{p\}\emptyset^{\omega}\}$. We have that $T \models^{a} \mathsf{F}p$, as we can pick $k_t = 0$ if $t = \{p\}\emptyset^{\omega}$, and $k_t = 1$ if $t = \emptyset\{p\}\emptyset^{\omega}$. On the other hand, there is no single k such that $T[k, \infty) \models^{s} p$, as the occurrences of p are at different positions. Consequently $T \not\models^{s} \mathsf{F}p$.

Moreover, consider the formula $\mathsf{F}p \vee \mathsf{F}p$ which is satisfied by T on both semantics. However, $\mathsf{F}p$ is not satisfied by T under synchronous semantics. Accordingly, we need to distinguish the two disjuncts $\mathsf{F}p$ and $\mathsf{F}p$ of $\mathsf{F}p \vee \mathsf{F}p$ to assign them to different teams.

In contrast, synchronous satisfaction implies asynchronous satisfaction, i.e., $T \models^{s} \varphi$ implies $T \models^{a} \varphi$. The simplest way to prove this is by applying downward closure, singleton equivalence, and flatness (see Fig. 1). Example 2.1 shows that the converse does not hold.

Next, we define the most important verification problems for LTL in team semantics setting, namely satisfiability and two variants of the model checking problem: For classical

LTL, one studies the path checking problem and the model checking problem. The difference between these two problems lies in the type of structures one considers. Recall that a model of an LTL formula is a single trace. In the path checking problem, a trace t and a formula φ are given, and one has to decide whether $t \models^{c} \varphi$. This problem has applications to runtime verification and monitoring of reactive systems [23, 26]. In the model checking problem, a Kripke structure $\mathcal{K}$ and a formula φ are given, and one has to decide whether every execution trace t of $\mathcal{K}$ satisfies φ.

The satisfiability problem of LTL under team semantics is defined as follows.

Problem: LTL satisfiability w.r.t. teams (TSAT*) for $\star \in \{\mathrm{a}, \mathrm{s}\}$.
Input: LTL formula φ.
Question: Is there a non-empty team T such that $T \models^{\star} \varphi$?

The non-emptiness condition is necessary, as otherwise every formula is satisfiable due to the empty team property (see Fig. 1).

We consider the generalisation of the path checking problem for LTL (denoted by LTL-PC), which asks for a given ultimately periodic trace t and a given formula φ, whether $t \models^{c} \varphi$ holds. In the team semantics setting, the corresponding question is whether a given finite team comprised of ultimately periodic traces satisfies a given formula. Such a team is given by a team encoding $\mathcal{T}$. To simplify our notation, we will write $\mathcal{T} \models^{\star} \varphi$ instead of $[\![\mathcal{T}]\!] \models^{\star} \varphi$.

Problem: TeamPathChecking (TPC*) for $\star \in \{\mathrm{a}, \mathrm{s}\}$.
Input: LTL formula φ and a finite team encoding $\mathcal{T}$.
Question: $\mathcal{T} \models^{\star} \varphi$?

Consider the generalised model checking problem where one checks whether the team of traces of a Kripke structure satisfies a given formula. This is the natural generalisation of the model checking problem for classical semantics, denoted by LTL-MC, which asks, for a given Kripke structure $\mathcal{K}$ and a given LTL formula φ, whether $t \models^{c} \varphi$ for every trace t of $\mathcal{K}$.

A Kripke structure $\mathcal{K} = (W, R, \eta, w_I)$ consists of a finite set W of worlds, a left-total transition relation $R \subseteq W \times W$, a labeling function $\eta \colon W \to 2^{\mathrm{AP}}$, and an initial world $w_I \in W$. A path π through $\mathcal{K}$ is an infinite sequence $\pi = \pi(0)\pi(1)\pi(2)\cdots \in W^\omega$ such that $\pi(0) = w_I$ and $(\pi(i), \pi(i+1)) \in R$ for every $i \geq 0$. The trace of π is defined as $t(\pi) = \eta(\pi(0))\eta(\pi(1))\eta(\pi(2))\cdots \in (2^{\mathrm{AP}})^\omega$. The Kripke structure $\mathcal{K}$ induces the team $T(\mathcal{K}) = \{t(\pi) \mid \pi \text{ is a path through } \mathcal{K}\}$.

Problem: TeamModelChecking (TMC*) for $\star \in \{\mathrm{a}, \mathrm{s}\}$.
Input: LTL formula φ and a Kripke structure $\mathcal{K}$.
Question: $T(\mathcal{K}) \models^{\star} \varphi$?

3 Basic Properties

We consider several standard properties of team semantics (cf., e.g. [9]) and verify which of these hold for our two semantics for LTL. These properties are later used to analyse the complexity of the satisfiability and model checking problems. To simplify our notation, $\models^{\star}$ denotes $\models^{\mathrm{a}}$ or $\models^{\mathrm{s}}$. See Figure 1 for the definitions of the properties and a summary for which semantics the properties hold. The positive results follow via simple inductive arguments. For the fact that synchronous semantics is not union closed, consider teams $T = \{\{p\}\emptyset^\omega\}$ and $T' = \{\emptyset\{p\}\emptyset^\omega\}$. Then, we have $T \models^{\mathrm{s}} \mathsf{F}p$ and $T' \models^{\mathrm{s}} \mathsf{F}p$ but $T \cup T' \not\models^{\mathrm{s}} \mathsf{F}p$. Note also that flatness is equivalent of being both downward and union closed.

It turns out that, by Figure 1, LTL under asynchronous team semantics is essentially classical LTL with a bit of universal quantification: for a team T and an LTL-formula φ,

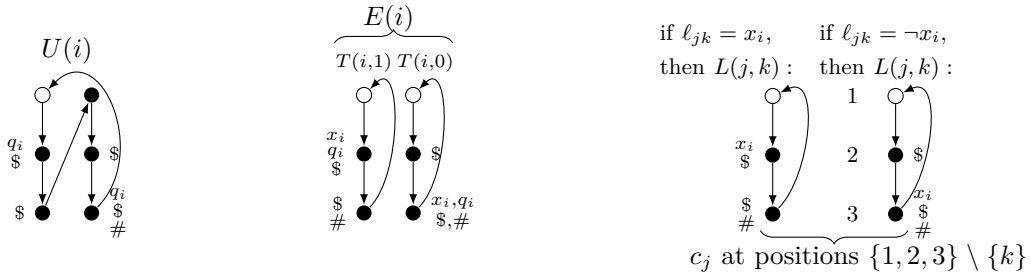


Figure 2 Traces for the reduction defined in the proof of Lemma 4.1.

we have $T \models^{\mathrm{a}} \varphi$ if and only if $\forall t \in T : t \models^{\mathrm{c}} \varphi$. This however does not mean that LTL under asynchronous team semantics is not worth of a study; it only means that asynchronous LTL is essentially classical LTL if we do not introduce additional atomic formulas that describe properties of teams directly. This is a common phenomenon in the team semantics setting. For instance, team semantics of first-order logic has the flatness property, but its extension by so-called *dependence atoms*, is equi-expressive with existential second-order logic [32]. Extensions of LTL under team semantics are discussed in Section 5.

At this point, it should not come as a surprise that, due to the flatness property and singleton equivalence, the complexity of satisfiability, path checking, and model checking for LTL under asynchronous team semantics coincides with those of classical LTL semantics. Firstly, note that an LTL-formula φ is satisfiable under asynchronous or synchronous team semantics if and only if there is a singleton team that satisfies the formula. Secondly, note that to check whether a given team satisfies φ under asynchronous semantics, it is enough to check whether each trace in the team satisfies φ under classical LTL; this can be computed by an AC^0-circuit using oracle gates for LTL-PC. Putting these observations together, we obtain the following results from the identical results for LTL under classical semantics [22, 23, 26, 31].

The circuit complexity class AC^i encompass of polynomial sized circuits of depth $O(\log^i(n))$ and unbounded fan-in; NC^i is similarly defined but with bounded fan-in. A language A is *constant-depth reducible* to a language B, in symbols $A \leq_{\mathrm{cd}} B$, if there exists a logtime-uniform AC^0-circuit family with oracle gates for B that decides membership in A. In this context, *logtime-uniform* means that there exists a deterministic Turing machine that can check the structure of the circuit family $\mathcal{C}$ in time $O(\log |\mathcal{C}|)$. For further information on circuit complexity, we refer the reader to the textbook of Vollmer [35]. Furthermore, logDCFL is the set of languages which are logspace reducible to a deterministic context-free language.

▶ **Proposition 3.1.**
1. $\mathrm{TMC}^{\mathrm{a}}$, $\mathrm{TSAT}^{\mathrm{a}}$, *and* $\mathrm{TSAT}^{\mathrm{s}}$ *are* PSPACE*-complete w.r.t.* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-reductions.*
2. $\mathrm{TPC}^{\mathrm{a}}$ *is in* $\mathrm{AC}^1(\mathrm{logDCFL})$ *and* NC^1*-hard w.r.t.* $\leq_{\mathrm{cd}}$*-reductions.*

4 Classification of Decision Problems Under Synchronous Semantics

In this section, we examine the computational complexity of path and model checking with respect to the synchronous semantics. Our main result settles the complexity of $\mathrm{TPC}^{\mathrm{s}}$. It turns out that this problem is harder than the asynchronous version.

▶ **Lemma 4.1.** $\mathrm{TPC}^{\mathrm{s}}$ *is* PSPACE*-hard w.r.t.* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-reductions.*

Proof. Determining whether a given quantified Boolean formula (qBf) is valid (QBF-VAL) is a well-known PSPACE-complete problem [25]. The problem stays PSPACE-complete if the matrix (i.e., the propositional part) of the given qBf is in 3CNF. To prove the claim of the lemma, we will show that QBF-VAL $\leq_{\mathrm{m}}^{\mathrm{p}}$ TPC$^{\mathrm{s}}$. Given a quantified Boolean formula φ, we stipulate, w.l.o.g., that φ is of the form $\exists x_1 \forall x_2 \cdots Q x_n \chi$, where $\chi = \bigwedge_{j=1}^{m} \bigvee_{k=1}^{3} \ell_{jk}$, $Q \in \{\exists, \forall\}$, and $x_1, \ldots, x_n$ are exactly the free variables of χ and pairwise distinct.

In the following we define a reduction which is composed of two functions f and g. Given a qBf φ, the function f will define an LTL-formula and g will define a team such that φ is valid if and only if $g(\varphi) \models^{\mathrm{s}} f(\varphi)$. Essentially, the team $g(\varphi)$ will contain three kinds of traces, see Figure 2: (i) traces which are used to mimic universal quantification ($U(i)$ and $E(i)$), (ii) traces that are used to simulate existential quantification ($E(i)$), and (iii) traces used to encode the matrix of φ ($L(j,k)$). Moreover the trace $T(i,1)$ ($T(i,0)$, resp.) is used inside the proof to encode an assignment that maps the variable x_i true (false, resp.). Note that, $U(i), T(i,1), T(i,0), L(j,k)$ are technically singleton sets of traces. For convenience, we identify them with the traces they contain.

Next we inductively define the reduction function f that maps qBf-formulas to LTL-formulas:

$$f(\chi) := \bigvee_{i=1}^{n} \mathsf{F} x_i \vee \bigvee_{i=1}^{m} \mathsf{F} c_i,$$

where χ is the 3CNF-formula $\bigwedge_{j=1}^{m} \bigvee_{k=1}^{3} \ell_{jk}$ with free variables $x_1, \ldots, x_n$,

$$f(\exists x_i \psi) := (\mathsf{F} q_i) \vee f(\psi),$$
$$f(\forall x_i \psi) := \big(\$ \vee (\neg q_i \mathsf{U} q_i) \vee \mathsf{F}[\# \wedge \mathsf{X} f(\psi)]\big) \mathsf{U} \#.$$

The reduction function g that maps qBf-formulas to teams is defined as follows with respect to the traces in Figure 2.

$$g(\chi) := \bigcup_{j=1}^{m} L(j,1) \cup L(j,2) \cup L(j,3),$$

where χ is the 3CNF-formula $\bigwedge_{j=1}^{m} \bigvee_{k=1}^{3} \ell_{jk}$ with free variables $x_1, \ldots, x_n$ and

$$g(\exists x_i \psi) := E(i) \cup g(\psi),$$
$$g(\forall x_i \psi) := U(i) \cup E(i) \cup g(\psi).$$

In Fig. 2, the first position of each trace is marked with a white circle. For instance, the trace of $U(i)$ is then encoded via

$$(\varepsilon, \emptyset\{q_i, \$\}\{\$\}\emptyset\{\$\}\{q_i, \$, \#\}).$$

The reduction function showing QBF-VAL $\leq_{\mathrm{m}}^{\mathrm{p}}$ TPC$^{\mathrm{s}}$ is then $\varphi \mapsto \langle g(\varphi), f(\varphi) \rangle$. Clearly $f(\varphi)$ and $g(\varphi)$ can be computed in linear time with respect to $|\varphi|$.

Intuitively, for the existential quantifier case, the formula $(\mathsf{F} q_i) \vee f(\psi)$ allows to continue in $f(\psi)$ with exactly one of $T(i,1)$ or $T(i,0)$. If $b \in \{0,1\}$ is a truth value then selecting $T(i,b)$ in the team is the same as setting x_i to b. For the case of $f(\forall x_i \psi)$, the formula $(\neg q_i \mathsf{U} q_i) \vee \mathsf{F}[\# \wedge \mathsf{X} f(\psi)]$ with respect to the team $(U(i) \cup E(i))[0, \infty)$ is similar to the existential case choosing x_i to be 1 whereas for $(U(i) \cup E(i))[3, \infty)$ one selects x_i to be 0. The use of the until operator in combination with \$ and # then forces both cases to happen.

Let $\varphi' = Q'x_{n'+1} \cdots Qx_n\chi$, where $Q', Q \in \{\exists, \forall\}$ and let I be an assignment of the variables in $\{x_1, \ldots, x_{n'}\}$ for $n' \leq n$. Then, let

$$g(I, \varphi') := g(\varphi') \cup \bigcup_{x_i \in \mathrm{Dom}(I)} T(i, I(x_i)).$$

We claim $I \models \varphi'$ if and only if $g(I, \varphi') \models^{\mathrm{s}} f(\varphi')$.

Note that when $\varphi' = \varphi$ it follows that $I = \emptyset$ and that $g(I, \varphi') = g(\varphi)$. Accordingly, the lemma follows from the claim of correctness. The claim is proven by induction on the number of quantifier alternations in φ'. The details can be found in the full version [21]. ◀

The matching upper bound follows via a PSPACE algorithm implementing the semantics in straightforward way. The details can be found in the full version [21].

▶ **Theorem 4.2.** $\mathrm{TPC}^{\mathrm{s}}$ *is* PSPACE*-complete w.r.t.* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-reductions.*

The next theorem deals with model checking of the splitjunction-free fragment of LTL under synchronous team semantics.

▶ **Theorem 4.3.** $\mathrm{TMC}^{\mathrm{s}}$ *restricted to splitjunction-free formulas is in* PSPACE.

Proof. Fix $\mathcal{K} = (W, R, \eta, w_I)$ and a splitjunction-free formula φ. We define $S_0 = \{w_I\}$ and $S_{i+1} = \{w' \in W \mid (w, w') \in R \text{ for some } w \in S_i\}$ for all $i \geq 0$. By the pigeonhole principle, this sequence is ultimately periodic with a characteristic (s, p) with $s + p \leq 2^{|W|}$.[3] Next, we define a trace t over $\mathrm{AP} \cup \{\overline{p} \mid p \in \mathrm{AP}\}$ via

$$t(i) = \{p \in \mathrm{AP} \mid p \in \eta(w) \text{ for all } w \in S_i\} \cup \{\overline{p} \mid p \notin \eta(w) \text{ for all } w \in S_i\}$$

that reflects the team semantics of (negated) atomic formulas, which have to hold in every element of the team.

An induction over the construction of φ shows that $T(\mathcal{K}) \models^{\mathrm{s}} \varphi$ if and only if $t \models^{\mathrm{c}} \overline{\varphi}$, where $\overline{\varphi}$ is obtained from φ by replacing each negated atomic proposition $\neg p$ by $\overline{p}$. To conclude the proof, we show that $t \models^{\mathrm{c}} \overline{\varphi}$ can be checked in non-deterministic polynomial space, exploiting the fact that t is ultimately periodic and of the same characteristic as $S_0S_1S_2\cdots$. However, as $s + p$ might be exponential, we cannot just construct a finite representation of t of characteristic (s, p) and then check satisfaction in polynomial space.

Instead, we present an on-the-fly approach which is inspired by similar algorithms in the literature. It is based on two properties:

1. Every S_i can be represented in polynomial space, and from S_i one can compute S_{i+1} in polynomial time.
2. For every LTL formula $\overline{\varphi}$, there is an equivalent non-deterministic Büchi automaton $\mathcal{A}_{\overline{\varphi}}$ of exponential size (see, e.g., [1] for a formal definition of Büchi automata and for the construction of $\mathcal{A}_{\overline{\varphi}}$). States of $\mathcal{A}_{\overline{\varphi}}$ can be represented in polynomial space and given two states, one can check in polynomial time, whether one is a successor of the other.

These properties allow us to construct both t and a run of $\mathcal{A}_{\overline{\varphi}}$ on t on the fly. The details can be found in the full version [21]. ◀

[3] The characteristic of an encoding (t_0, t_1) of an ultimately periodic trace $t_0t_1t_1t_1\cdots$ is the pair $(|t_0|, |t_1|)$. Slightly abusively, we say that $(|t_0|, |t_1|)$ is the characteristic of $t_0t_1t_1t_1\cdots$, although this is not unique.

The complexity of general model checking problem is left open. It is trivially PSPACE-hard, due to Theorem 4.2 and the fact that finite teams of ultimately periodic traces can be represented by Kripke structures. However, the problem is potentially much harder, as one has to deal with infinitely many splits of possibly uncountable teams with non-periodic traces, if a split occurs under the scope of a G-operator. Currently, we are working on interesting language-theoretic problems one encounters when trying to generalise our algorithms for the general path checking problem and for the splitjunction-free model checking problem, e.g., how complex can an LTL-definable split be, if the team to be split is one induced by a Kripke structure.

5 Extensions

In this section we take a brief look into extensions of our logics by dependence atoms and contradictory negation. Contradictory negation combined with team semantics allows for powerful constructions. For instance, the complexity of model checking for propositional logic jumps from NC^1 to PSPACE [27], whereas the complexity of validity and satisfiability jumps all the way to alternating exponential time with polynomially many alternations (ATIME-ALT(exp, pol)) [18].

Formally, we define that $T \models^{\star} \sim\varphi$ if $T \not\models \varphi$. Note that the negation $\sim$ is not equivalent to the negation $\neg$ of atomic propositions defined earlier, i.e., $\sim p$ and $\neg p$ are not equivalent. In the following, problems of the form $\mathrm{TPC}^{\mathrm{a}}(\sim)$, etc., refer to LTL-formulas with negation $\sim$.

Also, we are interested in atoms expressible in first-order (FO) logic over the atomic propositions; the most widely studied ones are dependence, independence, and inclusion atoms [9]. The notion of generalised atoms in the setting of first-order team semantics was introduced by Kuusisto [24]. It turns out that the algorithm for $\mathrm{TPC}^{\mathrm{s}}$ is very robust to such strengthenings of the logic under consideration.

We consider FO-formulas over the signature $(A_p)_{p\in\mathrm{AP}}$, where each A_p is a unary predicate. Furthermore, we interpret a team T as a relational structure $\mathfrak{A}(T)$ over the same signature with universe T such that $t \in T$ is in $A_p^{\mathfrak{A}}$ if and only if $p \in t(0)$. The formulas then express properties of the atomic propositions holding in the initial positions of traces in T. An FO-formula φ *FO-defines* the atomic formula D with $T \models^{\star} D \iff \mathfrak{A}(T) \models \varphi$. In this case, D is also called an *FO-definable generalised atom*.

For instance, the dependence atom $\mathrm{dep}(x; y)$ is FO-definable by $\forall t\forall t'((A_x(t) \leftrightarrow A_x(t')) \to \big(A_y(t) \leftrightarrow A_y(t')\big))$, for $x, y \in \mathrm{AP}$. We call an LTL-formula extended by a generalised atom D an $\mathrm{LTL}(D)$-formula. Similarly, we lift this notion to *sets of generalised atoms* as well as to the corresponding decision problems, i.e., $\mathrm{TPC}^{\mathrm{s}}(D)$ is the path checking problem over synchronous semantics with LTL formulas which may use the generalised atom D.

The result of Theorem 4.2 can be extended to facilitate also the contradictory negation and first-order definable generalised atoms.

▶ **Theorem 5.1.** *Let $\mathcal{D}$ be a finite set of first-order definable generalised atoms. Then $\mathrm{TPC}^{\mathrm{s}}(\mathcal{D})$ and $\mathrm{TPC}^{\mathrm{s}}(\sim)$ are* PSPACE*-complete w.r.t. $\leq^{\mathrm{p}}_{\mathrm{m}}$-reductions.*

The next proposition translates a result from Hannula et al. [18] to our setting. They show completeness for ATIME-ALT(exp, pol) for the satisfiability problem of propositional team logic with negation. This logic coincides with LTL-formulas without temporal operators under team semantics.

▶ **Proposition 5.2** ([18])**.** $\mathrm{TSAT}^{\mathrm{a}}(\sim)$ *and* $\mathrm{TSAT}^{\mathrm{s}}(\sim)$ *for formulas without temporal operators are complete for* ATIME-ALT(exp, pol) *w.r.t. $\leq^{\mathrm{p}}_{\mathrm{m}}$-reductions.*

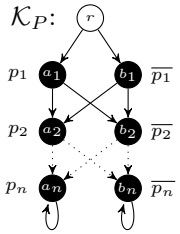


Figure 3 Kripke structure for the proof of Theorem 5.3.

▶ **Theorem 5.3.** $\mathrm{TMC}^{\mathrm{a}}(\sim)$ *and* $\mathrm{TMC}^{\mathrm{s}}(\sim)$ *are hard for* $\mathrm{ATIME\text{-}ALT}(\exp, \mathrm{pol})$ *w.r.t.* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-reductions.*

Proof. We will state a reduction from the satisfiability problem of propositional team logic with negation $\sim$ (short $\mathrm{PL}(\sim)$). The stated hardness then follows from Proposition 5.2.

For $P = \{p_1, \ldots, p_n\}$, consider the traces starting from the root r of the Kripke structure $\mathcal{K}_P$ depicted in Figure 3 using proposition symbols $p_1, \ldots, p_n, \overline{p_1}, \ldots, \overline{p_n}$. Each trace in the model corresponds to a propositional assignment on P. For $\varphi \in \mathrm{PL}(\sim)$, let φ^* denote the $\mathrm{LTL}(\sim)$-formula obtained by simultaneously replacing each (non-negated) variable p_i by $\mathsf{F}p_i$ and each negated variable $\neg p_i$ by $\mathsf{F}\overline{p_i}$. Let P denote the set of variables that occur in φ. Define $\top := (p \vee \neg p)$ and $\bot := p \wedge \neg p$, then $T(\mathcal{K}_P) \models^\star \big(\top \vee ((\sim\bot) \wedge \varphi^*)\big)$ *if and only if* $T' \models^\star \varphi^*$ for some non-empty $T' \subseteq T(\mathcal{K}_P)$. It is easy to check that $T' \models^\star \varphi^*$ *if and only if* the propositional team corresponding to T' satisfies φ and thus the above holds if and only if φ is satisfiable. ◀

In the following, we define the semantics for dependence atoms. For Teams $T \subseteq (2^{\mathrm{AP}})^\omega$ we define $T \models^\star \mathrm{dep}(p_1, \ldots, p_n; q_1, \ldots, q_m)$ if

$$\forall t, t' \in T: \ (t(0) \overset{p_1}{\Leftrightarrow} t'(0), \ldots, t(0) \overset{p_n}{\Leftrightarrow} t'(0)) \text{ implies } (t(0) \overset{q_1}{\Leftrightarrow} t'(0), \ldots, t(0) \overset{q_m}{\Leftrightarrow} t'(0)),$$

where $t(i) \overset{p}{\Leftrightarrow} t(j)$ means the sets $t(i)$ and $t(j)$ agree on proposition p, i.e., both contain p or not. Observe that the formula $\mathrm{dep}(;p)$ merely means that p has to be constant on the team. Often, due to convenience we will write $\mathrm{dep}(p)$ instead of $\mathrm{dep}(;p)$. Note that the hyperproperties 'input determinism' now can be very easily expressed via the formula $\mathrm{dep}(i_1, \ldots, i_n; o_1, \ldots, o_m)$, where i_j are the (public) input variables and o_j are the (public) output variables.

Problems of the form $\mathrm{TSAT}^{\mathrm{a}}(\mathrm{dep})$, etc., refer to LTL-formulas with dependence operator dep. The following proposition follows from the corresponding result for classical LTL using downwards closure and the fact that on singleton teams dependence atoms are trivially fulfilled.

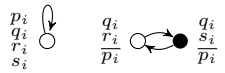
▶ **Proposition 5.4.** $\mathrm{TSAT}^{\mathrm{a}}(\mathrm{dep})$ *and* $\mathrm{TSAT}^{\mathrm{s}}(\mathrm{dep})$ *are* PSPACE*-complete.*

In the following, we will show a lower bound while the matching upper bound still is open.

▶ **Theorem 5.5.** $\mathrm{TPC}^{\mathrm{a}}(\mathrm{dep})$ *is* PSPACE*-hard w.r.t.* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-reductions.*

Proof. As in the proof of Lemma 4.1, we reduce from QBF-VAL.

Consider a given quantified Boolean formula $\exists x_1 \forall x_2 \cdots Q x_n \chi$, where $\chi = \bigwedge_{j=1}^{m} \bigvee_{k=1}^{3} \ell_{jk}$, $Q \in \{\exists, \forall\}$, and $x_1, \ldots, x_n$ are exactly the free variables of χ and pairwise distinct. We will use two traces for each variable x_i (gadget for x_i) as shown in Figure 4.



everywhere $p_j, \overline{p_j}$ for $i \neq j$

Figure 4 Traces in the proof of Theorem 5.5.

Intuitively, the proposition p_i marks that the variable x_i is set true while the proposition $\overline{p}_i$ marks that x_i is set false, q_i encodes that the gadget is used to quantify x_i, and s_i, r_i are auxiliary propositions. Picking the left trace corresponds to setting x_i to true and picking the right trace corresponds to setting x_i to false. In the following, we omit the p_j and $\overline{p_j}$, when $j \neq i$, for readability. Then, the team T is defined as

$$T := \{(\varepsilon, \{p_i, q_i, r_i, s_i\}), (\varepsilon, \{q_i, r_i, \overline{p}_i\}\{q_i, s_i, \overline{p}_i\}) \mid 1 \leq i \leq n\}.$$

Next, we recursively define the LTL(dep)-formula used in the reduction: $f(\chi)$ is obtained from χ by substituting every positive literal x_i by p_i and negated literal $\neg x_i$ by $\overline{p_i}$, $f(\exists x_i \psi) := \big(q_i \wedge \mathrm{dep}(p_i)\big) \vee f(\psi)\big)$, and

$$f(\forall x_i \psi) := \mathsf{G}\Big(\big(\mathrm{dep}(p_i) \wedge q_i \wedge r_i\big) \vee \big(s_i \wedge f(\psi)\big)\Big).$$

In the existential quantification of x_i, the splitjunction requires for the x_i-trace-pair to put $(\varepsilon, \{p_i, q_i, r_i, s_i\})$ into the left or right subteam (of the split). The trace $(\varepsilon, \{q_i, r_i, \overline{p}_i\}\{q_i, s_i, \overline{p}_i\})$ has to go to the opposite subteam as $\mathrm{dep}(p_i)$ requires p_i to be of constant value. (Technically both of the traces could be put to the right subteam, but this logic is downwards closed and, accordingly, this allows to omit this case.) As explained before, we existentially quantify x_i by this split. For universal quantification, the idea is a bit more involved. To verify $T \models^{\star} \mathsf{G}\theta$, where $\mathsf{G}\theta = f(\forall x_i \psi)$ essentially two different teams T' for which $T' \models \theta$ need to be verified.
(1.) $(\varepsilon, \{p_i, q_i, r_i, s_i\}), (\varepsilon, \{q_i, r_i, \overline{p}_i\}\{q_i, s_i, \overline{p}_i\}) \in T'$. In this case, $(\varepsilon, \{p_i, q_i, r_i, s_i\})$ must be put to the right subteam of the split and $(\varepsilon, \{q_i, r_i, \overline{p}_i\}\{q_i, s_i, \overline{p}_i\})$ to the left subteam, setting x_i true.
(2.) $(\varepsilon, \{p_i, q_i, r_i, s_i\}), (\varepsilon, \{q_i, s_i, \overline{p}_i\}\{q_i, r_i, \overline{p}_i\}) \in T'$. In this case, $(\varepsilon, \{p_i, q_i, r_i, s_i\})$ must be put to the left and $(\varepsilon, \{q_i, s_i, \overline{p}_i\}\{q_i, r_i, \overline{p}_i\})$ to the right subteam, implicitly forcing x_i to be false. These observations are utilised to prove that $\langle \exists x_1 \forall x_2 \cdots Q x_n \chi \rangle \in$ QBF-VAL if and only if $\langle f(\exists x_1 \forall x_2 \cdots Q x_n \chi), T \rangle \in \mathrm{TPC}^{\mathrm{a}}(\mathrm{dep})$. The reduction is polynomial time computable in the input size. ◀

The following result from Virtema talks about the validity problem of propositional team logic.

▶ **Proposition 5.6** ([34]). *Validity of propositional logic with dependence atoms is* NEXPTIME*-complete w.r.t.* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-reductions.*

▶ **Theorem 5.7.** $\mathrm{TMC}^{\mathrm{a}}(\mathrm{dep})$ *and* $\mathrm{TMC}^{\mathrm{s}}(\mathrm{dep})$ *are* NEXPTIME*-hard w.r.t.* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-reductions.*

Proof. The proof of this result uses the same construction idea as in the proof of Theorem 5.3, but this time from a different problem, namely, validity of propositional logic with dependence atoms which settles the lower bound by Proposition 5.6. Due to downwards closure the validity of propositional formulas with dependence atoms boils down to model checking the maximal team in the propositional (and not in the trace) setting, which essentially is achieved by $T(\mathcal{K})$, where $\mathcal{K}$ is the Kripke structure from the proof of Theorem 5.3. ◀

6 LTL under Team Semantics vs. HyperLTL

LTL under team semantics expresses hyperproperties [6], that is, sets of teams, or equivalently, sets of sets of traces. Recently, HyperLTL [5] was proposed to express information flow properties, which are naturally hyperproperties. For example, input determinism can be expressed as follows: every pair of traces that coincides on their input variables, also coincides on their output variables (this can be expressed in LTL with team semantics by a dependence atom dep as sketched is Section 5). To formalise such properties, HyperLTL allows to quantify over traces. This results in a powerful formalism with vastly different properties than LTL [14]. After introducing syntax and semantics of HyperLTL, we compare the expressive power of LTL under team semantics and HyperLTL.

The formulas of HyperLTL are given by the grammar

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

where p ranges over atomic propositions in AP and where π ranges over a given countable set $\mathcal{V}$ of *trace variables*. The other Boolean connectives and the temporal operators release R, eventually F, and always G are derived as usual, due to closure under negation. A sentence is a closed formula, i.e., one without free trace variables.

The semantics of HyperLTL is defined with respect to trace assignments that are a partial mappings $\Pi\colon \mathcal{V} \to (2^{\mathrm{AP}})^\omega$. The assignment with empty domain is denoted by $\Pi_\emptyset$. Given a trace assignment Π, a trace variable π, and a trace t, denote by $\Pi[\pi \to t]$ the assignment that coincides with Π everywhere but at π, which is mapped to t. Further, $\Pi[i, \infty)$ denotes the assignment mapping every π in Π's domain to $\Pi(\pi)[i, \infty)$. For teams T and trace-assignments Π we define

$$\begin{array}{ll}
(T,\Pi) \models^{\mathrm{h}} p_\pi & \text{if } p \in \Pi(\pi)(0), \\
(T,\Pi) \models^{\mathrm{h}} \neg\psi & \text{if } (T,\Pi) \not\models^{\mathrm{h}} \psi, \\
(T,\Pi) \models^{\mathrm{h}} \psi_1 \vee \psi_2 & \text{if } (T,\Pi) \models^{\mathrm{h}} \psi_1 \text{ or } (T,\Pi) \models^{\mathrm{h}} \psi_2, \\
(T,\Pi) \models^{\mathrm{h}} \mathsf{X}\psi & \text{if } (T,\Pi[1,\infty)) \models^{\mathrm{h}} \psi, \\
(T,\Pi) \models^{\mathrm{h}} \psi_1 \mathsf{U} \psi_2 & \text{if } \exists k \geq 0 : (T,\Pi[k,\infty)) \models^{\mathrm{h}} \psi_2 \text{ and } \forall 0 \leq k' < k : (T,\Pi[k',\infty)) \models^{\mathrm{h}} \psi_1, \\
(T,\Pi) \models^{\mathrm{h}} \exists \pi.\psi & \text{if } \exists t \in T : (T,\Pi[\pi \to t]) \models^{\mathrm{h}} \psi, \text{ and} \\
(T,\Pi) \models^{\mathrm{h}} \forall \pi.\psi & \text{if } \forall t \in T : (T,\Pi[\pi \to t]) \models^{\mathrm{h}} \psi.
\end{array}$$

We say that T satisfies a sentence φ, if $(T, \Pi_\emptyset) \models^{\mathrm{h}} \varphi$, and write $T \models^{\mathrm{h}} \varphi$. The semantics of HyperLTL are synchronous, i.e., the semantics of the until refers to a single k. Accordingly, one could expect that HyperLTL is closer related to LTL under synchronous team semantics than to LTL under asynchronous team semantics. In the following, we refute this intuition.

Formally, a HyperLTL sentence φ and an LTL formula φ' under synchronous (asynchronous) team semantics are equivalent, if for all teams T: $T \models^{\mathrm{h}} \varphi$ if and only if $T \models^{\mathrm{s}} \varphi'$ ($T \models^{\mathrm{a}} \varphi'$). In the following, let $\forall$-HyperLTL denote that set of HyperLTL sentences of the form $\forall \pi.\, \psi$ with quantifier-free ψ, i.e., sentences with a single universal quantifier.

▶ **Theorem 6.1.** 1. *No* LTL*-formula under synchronous or asynchronous team semantics is equivalent to* $\exists \pi.p_\pi$.
2. *No* HyperLTL *sentence is equivalent to* $\mathsf{F}p$ *under synchronous team semantics.*
3. LTL *under asynchronous team semantics is as expressive as* $\forall$-HyperLTL.

Proof.

1. Consider $T = \{\emptyset^\omega, \{p\}\emptyset^\omega\}$. We have $T \models^{\mathrm{h}} \exists \pi.p_\pi$. Assume there is an equivalent LTL formula under team semantics, call it φ. Then, $T \models^{\star} \varphi$ and thus $\{\emptyset^\omega\} \models^{\star} \varphi$ by downwards closure. Hence, by equivalence, $\{\emptyset^\omega\} \models^{\mathrm{h}} \exists \pi.p_\pi$, yielding a contradiction.

2. Bozzelli et al. proved that the property encoded by $\mathsf{F}p$ under synchronous team semantics cannot be expressed in HyperLTL [3].
3. Let φ be an LTL-formula and define $\varphi_h := \forall\pi.\varphi'$, where φ' is obtained from φ by replacing each atomic proposition p by p_π. Then, due to singleton equivalence, $T \models^{\mathrm{a}} \varphi$ if and only if $T \models^{\mathrm{h}} \varphi_h$. For the other implication, let $\varphi = \forall\pi.\psi$ be a HyperLTL sentence with quantifier-free ψ and let ψ' be obtained from ψ by replacing each atomic proposition p_π by p. Then, again due to the singleton equivalence, we have $T \models^{\mathrm{h}} \varphi$ if and only if $T \models^{\mathrm{a}} \psi'$.

◀

Note that these separations are obtained by very simple formulas, and are valid for LTL(dep) formulas, too. In particular, the HyperLTL formulas are all negation-free.

▶ **Corollary 6.2.** HyperLTL *and* LTL *under synchronous team semantics are of incomparable expressiveness and* HyperLTL *is strictly more expressive than* LTL *under asynchronous team semantics.*

7 Conclusion

We introduced synchronous and asynchronous team semantics for linear temporal logic LTL, studied complexity and expressive power of related logics, and compared them to HyperLTL. We concluded that LTL under team semantics is a valuable logic which allows to express relevant hyperproperties and complements the expressiveness of HyperLTL while allowing for computationally simpler decision problems. We conclude with some directions of future work and open problems.

1. We showed that some important properties that cannot be expressed in HyperLTL (such as uniform termination) can be expressed by LTL-formulas in synchronous team semantics. Moreover input determinism can be expressed in LTL(dep). What other important and practical hyperproperties can be expressed in LTL under team semantics? What about in its extensions with dependence, inclusion, and independence atoms, or the contradictory negation.
2. We showed that with respect to expressive power HyperLTL and LTL under synchronous team semantics are incomparable. What about the extensions of LTL under team semantics? For example the HyperLTL formula $\exists\pi.p_\pi$ is expressible in LTL($\sim$). Can we characterise the expressive power of relevant extensions of team LTL as has been done in first-order and modal contexts?
3. We studied the complexity of path-checking, model checking, and satisfiability problems of team LTL and its extensions with dependence atoms and the contradictory negation. Many problems are still open: Can we show matching upper bounds for the hardness results of Section 5? What is the complexity of $\mathrm{TMC}^{\mathrm{s}}$ when splitjunctions are allowed? What happens when LTL is extended with inclusion or independence atoms?
4. Can we give a natural team semantics to CTL^* and compare it to $\mathrm{HyperCTL}^*$ [5]?

References

1 Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking.* The MIT Press, 2008.

2 Borzoo Bonakdarpour and Bernd Finkbeiner. Runtime verification for HyperLTL. In Yliès Falcone and César Sánchez, editors, *RV 2016*, volume 10012 of *LNCS*, pages 41–45. Springer, 2016.

3 Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Unifying hyper and epistemic temporal logics. In Andrew M. Pitts, editor, *FoSSaCS 2015*, volume 9034 of *LNCS*, pages 167–182. Springer, 2015.

4 Julian Bradfield. On the structure of events in boolean games. In *Logics for Dependence and Independence*. Dagstuhl Reports, 2015.

5 Michael R. Clarkson, Bernd Finkbeiner, Masoud Koleini, Kristopher K. Micinski, Markus N. Rabe, and César Sánchez. Temporal logics for hyperproperties. In Martín Abadi and Steve Kremer, editors, *POST 2014*, volume 8414 of *LNCS*, pages 265–284. Springer, 2014.

6 Michael R. Clarkson and Fred B. Schneider. Hyperproperties. *Journal of Computer Security*, 18(6):1157–1210, 2010.

7 Pedro R. D'Argenio, Gilles Barthe, Sebastian Biewer, Bernd Finkbeiner, and Holger Hermanns. Is your software on dope? - formal analysis of surreptitiously "enhanced" programs. In Hongseok Yang, editor, *ESOP 2017*, volume 10201 of *LNCS*, pages 83–110. Springer, 2017.

8 Arnaud Durand, Miika Hannula, Juha Kontinen, Arne Meier, and Jonni Virtema. Approximation and dependence via multiteam semantics. In *FoIKS 2016*, pages 271–291, 2016.

9 Arnaud Durand, Juha Kontinen, and Heribert Vollmer. Expressivity and complexity of dependence logic. In *Dependence Logic: Theory and Applications*, pages 5–32. Birkhäuser, 2016.

10 Bernd Finkbeiner and Christopher Hahn. Deciding Hyperproperties. In Josée Desharnais and Radha Jagadeesan, editors, *CONCUR 2016*, volume 59 of *LIPIcs*, pages 13:1–13:14, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

11 Bernd Finkbeiner, Christopher Hahn, Marvin Stenger, and Leander Tentrup. Monitoring hyperproperties. In Shuvendu K. Lahiri and Giles Reger, editors, *RV 2017*, volume 10548 of *LNCS*, pages 190–207. Springer, 2017.

12 Bernd Finkbeiner, Christian Müller, Helmut Seidl, and Eugen Zalinescu. Verifying security policies in multi-agent workflows with loops. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *CCS 2017*, pages 633–645. ACM, 2017. doi:10.1145/3133956.

13 Bernd Finkbeiner, Markus N. Rabe, and César Sánchez. Algorithms for model checking HyperLTL and HyperCTL*. In Daniel Kroening and Corina S. Pasareanu, editors, *CAV 2015 (Part I)*, volume 9206 of *LNCS*, pages 30–48. Springer, 2015.

14 Bernd Finkbeiner and Martin Zimmermann. The first-order logic of hyperproperties. In Heribert Vollmer and Brigitte Vallée, editors, *STACS 2017*, volume 66 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

15 Pietro Galliani. Inclusion and exclusion dependencies in team semantics - on some logics of imperfect information. *Ann. Pure Appl. Logic*, 163(1):68–84, 2012.

16 Erich Grädel, Juha Kontinen, Jouko Väänänen, and Heribert Vollmer. Logics for dependence and independence (Dagstuhl seminar 15261). *Dagstuhl Reports*, 5(6):70–85, 2015.

17 Erich Grädel and Jouko Väänänen. Dependence and independence. *Studia Logica*, 101(2):399–410, 2013.

18 Miika Hannula, Juha Kontinen, Jonni Virtema, and Heribert Vollmer. Complexity of propositional logics in team semantic. *ACM Trans. Comput. Logic*, 19(1):2:1–2:14, 2018.

19 Wilfrid Hodges. Compositional semantics for a language of imperfect information. *Logic Journal of the IGPL*, 5(4):539–563, 1997.

20 Andreas Krebs, Arne Meier, and Jonni Virtema. A team based variant of CTL. In Fabio Grandi, Martin Lange, and Alessio Lomuscio, editors, *TIME 2015*, pages 140–149. IEEE Computer Society, 2015.

21 Andreas Krebs, Arne Meier, Jonni Virtema, and Martin Zimmermann. Team semantics for the specification and verification of hyperproperties. *CoRR*, abs/1709.08510, 2017. arXiv:1709.08510.

22 Lars Kuhtz. *Model checking finite paths and trees*. PhD thesis, Saarland University, 2010.

23 Lars Kuhtz and Bernd Finkbeiner. LTL path checking is efficiently parallelizable. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris E. Nikoletseas, and Wolfgang Thomas, editors, *ICALP 2009 (Part II)*, volume 5556 of *LNCS*, pages 235–246. Springer, 2009.

24 Antti Kuusisto. A Double Team Semantics for Generalized Quantifiers. *Journal of Logic, Language and Information*, 24(2):149–191, 2015.

25 Richard Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM J. Comput.*, 6(3):467–480, 1977.

26 Nicolas Markey and Philippe Schnoebelen. Model checking a path. In Roberto M. Amadio and Denis Lugiez, editors, *CONCUR 2003*, volume 2761 of *LNCS*, pages 248–262. Springer, 2003.

27 Julian-Steffen Müller. *Satisfiability and Model Checking in Team Based Logics*. PhD thesis, Leibniz University of Hannover, 2014.

28 Amir Pnueli. The temporal logic of programs. In *FOCS 1977*, pages 46–57. IEEE Computer Society, 1977.

29 Markus N. Rabe. *A Temporal Logic Approach to Information-flow Control*. PhD thesis, Saarland University, 2016.

30 Ilya Shpitser. Causal inference and logics of dependence and independence. In *Logics for Dependence and Independence*. Dagstuhl Reports, 2015.

31 A. Prasad Sistla and Edmund M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, 1985.

32 Jouko Väänänen. *Dependence Logic*. Cambridge University Press, 2007.

33 Jouko Väänänen. Modal Dependence Logic. In *New Perspectives on Games and Interaction*. Amsterdam University Press, Amsterdam, 2008.

34 Jonni Virtema. Complexity of validity for propositional dependence logics. *Inf. Comput.*, 253:224–236, 2017. doi:10.1016/j.ic.2016.07.008.

35 Heribert Vollmer. *Introduction to Circuit Complexity – A Uniform Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 1999.