Linear-time Temporal Logic with Team Semantics: Expressivity and Complexity

Jonni Virtema
Institute for Theoretical Computer Science, Leibniz Universität Hannover, Germany
Department of Computer Science, University of Sheffield, United Kingdom

Jana Hofmann
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Bernd Finkbeiner
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Juha Kontinen
Department of Mathematics and Statistics, University of Helsinki, Finland

Fan Yang
Department of Mathematics and Statistics, University of Helsinki, Finland

Abstract

We study the expressivity and complexity of model checking linear temporal logic with team semantics (TeamLTL). TeamLTL, despite being a purely modal logic, is capable of defining hyperproperties, i.e., properties which relate multiple execution traces. TeamLTL has been introduced quite recently and only few results are known regarding its expressivity and its model checking problem. We relate the expressivity of TeamLTL to logics for hyperproperties obtained by extending LTL with trace and propositional quantifiers (HyperLTL and HyperQPTL). By doing so, we obtain a number of model checking results for TeamLTL and identify its undecidability frontier. In particular, we show decidability of model checking of the so-called *left-flat* fragment of any downward closed TeamLTL-extension. Moreover, we establish that the model checking problem of TeamLTL with Boolean disjunction and inclusion atoms is undecidable.

2012 ACM Subject Classification Theory of computation → Logic and verification; Theory of computation → Modal and temporal logics

Keywords and phrases Linear temporal logic, Hyperproperties, Model Checking, Expressivity

Funding *Jana Hofmann and Bernd Finkbeiner:* Partially supported by the German Research Foundation (DFG) as part of the Collaborative Research Center "Foundations of Perspicuous Software Systems" (TRR 248, 389792660) and by the European Research Council (ERC) Grant OSARES (No. 683300).
Jonni Virtema: Supported by the DFG grant VI 1045/1-1.
Juha Kontinen: Supported by grant 308712 of the Academy of Finland.
Fan Yang: Supported by grants 330525 and 308712 of Academy of Finland, and Research Funds of University of Helsinki.

1 Introduction

Linear-time temporal logic (LTL) is one of the most prominent logics for the specification and verification of reactive and concurrent systems. Practical model checking tools like SPIN, NuSMV, and many others ([29, 6, 11]) automatically verify whether a given computer system, such as a hardware circuit or a communication protocol, is correct with respect to its LTL specification. The basic principle, as introduced in 1977 by Amir Pnueli [40], is to specify the correctness of a program as a set of infinite sequences, called *traces*, which define the acceptable executions of the system.

Hyperproperties, i.e., properties which relate multiple execution traces, cannot be specified in LTL. Such properties are of prime interest in information flow security, where dependencies between the secret inputs and the publicly observable outputs of a system are considered potential security violations. Commonly known properties of that type are noninterference [42, 38] or observational determinism [50]. In other settings, relations between traces are explicitly desirable: robustness properties, for example, state that similar inputs lead to similar outputs. Hyperproperties are not limited to the area of information flow control. E.g., distributivity and other system properties like fault tolerance can be expressed as hyperproperties [17].

The main approach to specify hyperproperties has been to extend temporal logics like LTL, CTL, and QPTL with explicit trace and path quantification, resulting in logics like HyperLTL [7], HyperCTL* [7], and HyperQPTL [41, 9]. Most frequently used is HyperLTL, which can express noninterference as follows: $\forall\pi.\,\forall\pi'.\,\Box(\bigwedge_{i\in I} i_\pi \leftrightarrow i_{\pi'}) \rightarrow \Box(\bigwedge_{o\in O} o_\pi \leftrightarrow o_{\pi'})$. The formula states that any two traces which globally agree on the value of the public inputs I also globally agree on the public outputs O. Consequently, the value of secret inputs cannot affect the value of the publicly observable outputs.

It is not clear, however, whether quantification over traces is the best way to express hyperproperties. The success of LTL over first-order logics for the specification of linear-time properties stems from the fact that its modal operators replace explicit quantification of points in time. This allows for a much more concise and readable formulation of the same property. The natural question to ask is whether a purely modal logic for hyperproperties would have similar advantages. A candidate for such a logic is LTL with *team semantics* [34]. Under team semantics, LTL expresses hyperproperties without explicit references to traces. Instead, each subformula is evaluated with respect to a set of traces, called a *team*. Temporal operators advance time on all traces of the current team. Using the split operator $\vee$, teams can be split during the evaluation of a formula, which enables us to express properties of subsets of traces.

As an example, consider the property that there is a point in time, common for all traces, after which a certain event a does not occur any more. We need a propositional and a trace quantifier to express such a property in HyperQPTL (it is not expressible in HyperLTL). The formula $\exists p.\,\forall\pi.\,\Diamond p \wedge \Box(p \rightarrow \Box\neg a_\pi)$ states that there is a p-sequence $s \in (2^{\{p\}})^\omega$ such that p is set at least once, and if $p \in s[i]$, then a is not set on all traces π on all points in time starting from i. The same property can be expressed in TeamLTL without any quantification simply as $\Diamond\Box\neg a$. The formula exploits the synchronous semantics of TeamLTL by stating that there is a point such that for all future points all traces have a not set. As a second example, consider the case that an unknown input determines the behaviour of the system. Depending on the input, its execution traces either agree on a or on b. We can express the property in HyperLTL with three trace quantifiers: $\exists\pi_1.\,\exists\pi_2.\,\forall\pi.\,\Box(a_{\pi_1} \leftrightarrow a_\pi) \vee \Box(b_{\pi_2} \leftrightarrow b_\pi)$. In TeamLTL, the same property can be simply expressed as $\Box(a \ovee \neg a) \vee \Box(b \ovee \neg b)$. The Boolean or operator $\ovee$ expresses that in the current team, either the left side holds on all traces or the right side does.

The use of the $\ovee$ operator reveals another strength of TeamLTL: its modularity. The research on team semantics (see related work section) has a rich tradition of studying extensions of team logics with new atomic statements and operators. They constitute a well-defined way to increase a logic's expressiveness in a step-by-step manner. Besides $\ovee$, examples are Boolean negation $\sim$, the inclusion atom $\subseteq$, and universal subteam quantifiers A and $\overset{1}{\mathsf{A}}$. Inclusion atoms have been found to be fascinating for their ability to express recursion in the first-order setting; the expressivity of FO($\subseteq$) coincides with greatest fixed point logic and hence PTIME [20]. In turn, all LTL-definable properties can be expressed by

TeamLTL-formulae of the form $\overset{1}{\mathsf{A}}\varphi$. With the introduction of generalised atoms, TeamLTL even permits custom extensions. Possibly most interesting in the context of hyperproperties are dependence atoms. A dependence atom $\mathrm{dep}(x_1, \ldots, x_n)$ is satisfied by a team X if any two assignments assigning the same values to the variables $x_1, \ldots, x_{n-1}$ also assign the same value to x_n. For example, the TeamLTL formula $(\Box\mathrm{dep}(i_1, i_2, o)) \vee (\Box\mathrm{dep}(i_2, i_3, o))$ states that the executions of the system can be decomposed into two parts; in the first part, the output o is determined by the inputs i_1 and i_2, and in the second part, o is determined by the inputs i_2 and i_3.

Temporal team logics constitute a new, fundamentally different approach to specify hyperproperties. While HyperLTL and other quantification-based hyperlogics have been studied extensively (see section on related work), only few results are known about the expressive power and complexity of TeamLTL and its variants. In particular, we know very little about how the expressivity of the two approaches compares. What is known is that HyperLTL and TeamLTL are incomparable in expressivity [34] and that the model checking problem of TeamLTL without splitjunctions $\vee$ (what makes the logic significantly weaker) is in PSPACE [34]. On the other hand, it was recently shown that the complexity of satisfiability and model checking of TeamLTL with Boolean negation $\sim$ is equivalent to the decision problem of third-order arithmetic [36] and hence highly undecidable.

Our contribution. We advance the understanding of team-based logics for hyperproperties by exploring the relative expressivity of TeamLTL and temporal hyperlogics like HyperLTL, as well as the decidability frontier of the model checking problem of TeamLTL. Our expressivity and model checking results are summarized in Table 1 and Table 2. We identify expressively complete extensions of TeamLTL (displayed on the left of Table 1) that can express all (all downward closed, resp.) Boolean relations on LTL-properties of teams, and present several translations from team logics to hyperlogics. We begin by approaching the decidability frontier of TeamLTL from above, and tackle a question posed in [36]: *Does some sensible restriction to the use of Boolean negation in* TeamLTL($\sim$) *yield a decidable logic?* We show that already a very restricted access to $\sim$ leads to high undecidability, whereas already the use of inclusion atoms $\subseteq$ together with Boolean disjunctions $\varovee$ suffices for undecidable model checking. Furthermore, we establish that these complexity results transfer to the satisfiability problem of the related logics. Next, regarding the expressivity of TeamLTL, we show that its extensions with all (all downward closed, resp.) atomic LTL-properties of teams translate to simple fragments of HyperQPTL$^+$. Consequently, known decidability results for quantification-based hyperlogics enable us to approach the decidability frontier of TeamLTL extensions from below. We establish an efficient translation from the so-called *k-coherent fragment* of TeamLTL($\sim$) to the universal fragment of HyperLTL (for which model checking is PSPACE-complete [19]) and thereby obtain EXPSPACE model checking for the fragment. Finally, we show that the so-called *left-flat* fragment of TeamLTL($\varovee, \overset{1}{\mathsf{A}}$) enjoys decidable model checking via a translation to $\overset{u}{\exists}{}^*_p\forall^*_\pi$HyperQPTL.

Related work. The development of team semantics began with the introduction of Dependence Logic [46], which adds the concept of functional dependence to first-order logic by means of new atomic dependence formulae. During the past decade, team semantics has been generalised to propositional [49], modal [47], temporal [33], and probabilistic [13] frameworks, and fascinating connections to fields such as database theory [23], statistics [12], real valued computation [24], and quantum information theory [30] has been identified. In the modal team semantics setting, model checking and satisfiability problems have been shown to be decidable, see [26, page 627] for an overview of the complexity landscape. Expressivity and definability of related logics is also well understood, see, e.g. [27, 32, 43]. The study

$$
\begin{array}{ccc}
 & & \text{(assuming left-flatness)} \\
\mathrm{TeamLTL}(\ovee, \overset{1}{\mathsf{A}}) & \overset{\text{Thm.14}}{\leq} & \overset{u}{\exists}{}^{*}_{q}\forall_{\pi}\mathrm{HyperQPTL} \\
 & \overset{\text{Thm.6}}{\leq} & \exists_{p}\overset{u}{Q}{}^{*}_{p}\forall_{\pi}\mathrm{HyperQPTL}^{+} \\
\wedge^{\dagger} & & \\
\mathrm{TeamLTL}(\ovee, \sim\!\bot, \overset{1}{\mathsf{A}}) & \overset{\text{Thm.6}}{\leq} & \exists_{p}\overset{u}{Q}{}^{*}_{p}\exists^{*}_{\pi}\forall_{\pi}\mathrm{HyperQPTL}^{+} \\
 & & \\
\mid\!\wedge\ [36] & & \text{(assuming } k\text{-coherence)} \\
\mathrm{TeamLTL}(\sim) & \overset{\text{Thm.9}}{\leq} & \forall^{k}\mathrm{HyperLTL}
\end{array}
$$

Table 1 Expressivity results. The logics $\mathrm{TeamLTL}(\ovee, \sim\!\bot, \overset{1}{\mathsf{A}})$ and $\mathrm{TeamLTL}(\overset{1}{\mathsf{A}}, \ovee)$ can express *all/all downward closed* atomic LTL-properties of teams (see the discussion at the end of Section 2). † holds since $\mathrm{TeamLTL}(\overset{1}{\mathsf{A}}, \ovee)$ is downward closed.

Logic	Model Checking Result
TeamLTL without $\vee$	in PSPACE [34]
k-coherent TeamLTL($\sim$)	in EXPSPACE [Thm. 10]
left-flat $\mathrm{TeamLTL}(\ovee, \overset{1}{\mathsf{A}})$	in EXPSPACE [Thm. 15]
$\mathrm{TeamLTL}(\subseteq, \ovee)$	Σ^0_1-hard [Thm. 2]
$\mathrm{TeamLTL}(\subseteq, \ovee, \mathsf{A})$	Σ^1_1-hard [Thm. 3]
TeamLTL($\sim$)	complete for third-order arithmetic [36]

Table 2 Complexity results.

of temporal logics with team semantics, was initiated in [33], where team semantics for computational tree logic CTL was given. The idea to develop team-based logics for hyperproperties was coined in [34], where TeamLTL was first introduced and shown incomparable to HyperLTL. The interest on logics for hyperproperties, so-called hyperlogics, was sparked by the introduction of HyperLTL and HyperCTL* [7]. Many temporal logics have since been extended with trace and path quantification to obtain various hyperlogics, e.g., to express asynchronous hyperproperties [22, 4], hyperproperties on finite traces [21], probabilistic hyperproperties [1], or timed hyperproperties [28]. Model checking HyperLTL and the strictly more expressive HyperQPTL is decidable, though k-EXPSPACE-complete, where k is the number of quantifier alternations in the formula [19, 41]. Model checking HyperQPTL$^+$, on the other hand, is undecidable [16]. The expressivity of HyperLTL, HyperCTL*, and HyperQPTL has been compared to first-order and second-order hyperlogics resulting in a hierarchy of hyperlogics [9]. Beyond model checking and expressivity questions, especially HyperLTL has been studied extensively. This includes its satisfiability [15, 37], runtime monitoring [18, 2] and enforcement problems [10], as well as synthesis [17].

2 Basics of TeamLTL

Let us start by recalling the syntax of LTL from the literature. Fix a set AP of *atomic propositions*. The set of formulae of LTL (over AP) is generated by the following grammar:

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \bigcirc \varphi \mid \varphi\,\mathcal{U}\,\varphi \mid \varphi\,\mathcal{W}\,\varphi, \qquad \text{where } p \in \mathrm{AP}.$$

We adopt, as is common in studies on team logics, the convention that formulae are given in negation normal form. The logical constants $\top, \bot$ and connectives $\rightarrow, \leftrightarrow$ are defined as usual (e.g., $\bot := p \land \neg p$ and $\top := p \lor \neg p$), and $\Diamond \varphi := \top \,\mathcal{U}\, \varphi$ and $\Box \varphi := \varphi \,\mathcal{W}\, \bot$.

A *trace* t over AP is an infinite sequence from $(2^{\mathrm{AP}})^\omega$. For a natural number $i \in \mathbb{N}$, we denote by $t[i]$ the ith element of t and by $t[i,\infty]$ the postfix $(t[j])_{j \geq i}$ of t. The satisfaction relation $(t,i) \models \varphi$, for LTL formulae φ, is defined as usual, see e.g., [39]. We use $[\![\varphi]\!]_{(t,i)} \in \{0,1\}$ to denote the truth value of φ on (t,i). A *(temporal) team* is a pair (T,i) consisting a set of traces $T \subseteq (2^{\mathrm{AP}})^\omega$ and a natural number $i \in \mathbb{N}$ representing the time step. We write $T[i]$ and $T[i,\infty]$ to denote the sets $\{t[i] \mid t \in T\}$ and $\{t[i,\infty] \mid t \in T\}$, respectively.

Let us next introduce the logic LTL interpreted with team semantics (denoted TeamLTL). TeamLTL was first studied in [34], where it was called LTL with *synchronous* team semantics. The satisfaction relation $(T,i) \models \varphi$ for TeamLTL is defined as follows:

$$
\begin{array}{lll}
(T,i) \models p & \text{iff} & \forall t \in T : p \in t[i] \\
(T,i) \models \neg p & \text{iff} & \forall t \in T : p \notin t[i] \\
(T,i) \models \varphi \land \psi & \text{iff} & (T,i) \models \varphi \text{ and } (T,i) \models \psi \\
(T,i) \models \bigcirc \varphi & \text{iff} & (T,i+1) \models \varphi \\
(T,i) \models \varphi \land \psi & \text{iff} & (T,i) \models \varphi \text{ and } (T,i) \models \psi \\
(T,i) \models \bigcirc \varphi & \text{iff} & (T,i+1) \models \varphi \\
(T,i) \models \varphi \lor \psi & \text{iff} & (T_1,i) \models \varphi \text{ and } (T_2,i) \models \psi, \text{ for some } T_1, T_2 \text{ s.t. } T_1 \cup T_2 = T \\
(T,i) \models \varphi \,\mathcal{U}\, \psi & \text{iff} & \exists k \geq i \text{ such that } (T,k) \models \psi \text{ and } \forall m : i \leq m < k \Rightarrow (T,m) \models \varphi \\
(T,i) \models \varphi \,\mathcal{W}\, \psi & \text{iff} & \forall k \geq i : (T,k) \models \varphi \text{ or } \exists m \text{ such that } i \leq m \leq k \text{ and } (T,m) \models \psi
\end{array}
$$

Note that $(T,i) \models \bot$ iff $T = \emptyset$. Two formulae φ and ψ are *equivalent* (written $\varphi \equiv \psi$), if the equivalence $(T,i) \models \varphi$ iff $(T,i) \models \psi$ holds for every (T,i). We say that a logic $\mathcal{L}_2$ is *at least as expressive* as a logic $\mathcal{L}_1$ (written $\mathcal{L}_1 \leq \mathcal{L}_2$) if for every $\mathcal{L}_1$-formula φ, there exists an $\mathcal{L}_2$-formula ψ such that $\varphi \equiv \psi$. We write $\mathcal{L}_1 \equiv \mathcal{L}_2$ if both $\mathcal{L}_1 \leq \mathcal{L}_2$ and $\mathcal{L}_2 \leq \mathcal{L}_1$ hold. The following are important semantic properties of formulae from the team semantics literature:

(Downward closure) If $(T,i) \models \varphi$ and $S \subseteq T$, then $(S,i) \models \varphi$.
(Empty team property) $(\emptyset, i) \models \varphi$.
(Flatness) $(T,i) \models \varphi$ iff $(\{t\},i) \models \varphi$ for all $t \in T$.
(Singleton equivalence) $(\{t\},i) \models \varphi$ iff $(t,i) \models \varphi$.

A logic has one of the above properties if every formula of the logic has the property. TeamLTL satisfies downward closure, singleton equivalence, and the empty team property [34]. However, it does not satisfy flatness; for instance, the formula $\Diamond p$ is not flat.

The power of team semantics comes with the ability to enrich logics with novel atomic statements describing properties of teams. We thereby easily get a hierarchy of team logics of different expressiveness. The most prominent examples of such atoms are *dependence atoms* $\mathrm{dep}(\varphi_1,\ldots,\varphi_n,\psi)$ and *inclusion atoms* $\varphi_1,\ldots,\varphi_n \subseteq \psi_1,\ldots,\psi_n$, with $\varphi_1,\ldots,\varphi_n,\psi,\psi_1,\ldots,\psi_n$ being LTL-formulae. The dependence atom states that the truth value of ψ is functionally determined by that of $\varphi_1,\ldots,\varphi_n$. The inclusion atom states that each value combination of $\varphi_1,\ldots,\varphi_n$ must also occur as a value combination for $\psi_1,\ldots,\psi_n$. Their formal semantics is defined as:

$$(T,i) \models \mathrm{dep}(\varphi_1,\ldots,\varphi_n,\psi) \text{ iff } \forall t,t' \in T : \Big(\bigwedge_{1 \leq j \leq n} [\![\varphi_j]\!]_{(t,i)} = [\![\varphi_j]\!]_{(t',i)} \Big) \Rightarrow [\![\psi]\!]_{(t,i)} = [\![\psi]\!]_{(t',i)}$$

$$(T,i) \models \varphi_1,\ldots,\varphi_n \subseteq \psi_1,\ldots,\psi_n \text{ iff } \forall t \in T\, \exists t' \in T : \bigwedge_{1 \leq j \leq n} [\![\varphi_j]\!]_{(t,i)} = [\![\psi_j]\!]_{(t',i)}$$

As an example, let $o_1, \ldots, o_n$ be some observable outputs and s be a secret. The atom $(o_1, \ldots, o_n, s) \subseteq (o_1, \ldots, o_n, \neg s)$ expresses a form of *non-inference* by stating that an observer cannot infer the current value of the secret from the outputs. We also consider other connectives known in the team semantics literature: *Boolean disjunction* $\otimes$, *Boolean negation* $\sim$, and *universal subteam quantifiers* A and $\overset{1}{\mathsf{A}}$, with their semantics defined as:

$$\begin{array}{lll}
(T,i) \models \varphi \otimes \psi & \text{iff} & (T,i) \models \varphi \text{ or } (T,i) \models \psi \\
(T,i) \models \sim \varphi & \text{iff} & (T,i) \not\models \varphi \\
(T,i) \models \mathsf{A}\varphi & \text{iff} & \forall S \subseteq T : (S,i) \models \varphi \\
(T,i) \models \overset{1}{\mathsf{A}}\varphi & \text{iff} & \forall t \in T : (\{t\},i) \models \varphi
\end{array}$$

If $\mathcal{A}$ is a collection of atoms and connectives, we let TeamLTL($\mathcal{A}$) denote the extension of TeamLTL with the atoms and connectives in $\mathcal{A}$. For any atom or connective $\circ$, we write simply TeamLTL($\mathcal{A}, \circ$) instead of TeamLTL($\mathcal{A} \cup \{\circ\}$).

TeamLTL($\sim$) is a very expressive logic; all of the above connectives and atoms, as well as many others, have been shown to be definable in TeamLTL($\sim$) [25, 36]. To systematically explore less expressive variants of TeamLTL, we introduce two representative logics of different expressiveness, namely TeamLTL($\otimes, \overset{1}{\mathsf{A}}$) and TeamLTL($\otimes, \sim\bot, \overset{1}{\mathsf{A}}$). The expression $\sim\bot$ can be used to enforce non-emptiness of a team. What makes these logics good representatives is their semantic property that they can express a general class of Boolean relations. Let B be a set of n-ary Boolean relations. We define the property $[\varphi_1, \ldots, \varphi_n]_B$ for an n-tuple $(\varphi_1, \ldots, \varphi_n)$ of LTL-formulae:

$$(T,i) \models [\varphi_1, \ldots, \varphi_n]_B \quad \text{iff} \quad \{(\llbracket \varphi_1 \rrbracket_{(t,i)}, \ldots, \llbracket \varphi_n \rrbracket_{(t,i)}) \mid t \in T\} \in B.$$

The logic TeamLTL($\otimes, \sim\bot, \overset{1}{\mathsf{A}}$) is expressively complete with respect to all $[\varphi_1, \ldots, \varphi_n]_B$. That is, for every set of Boolean relations B and LTL-formulae $\varphi_1, \ldots, \varphi_n$, the property $[\varphi_1, \ldots, \varphi_n]_B$ is expressible in TeamLTL($\otimes, \sim\bot, \overset{1}{\mathsf{A}}$). Furthermore, TeamLTL($\otimes, \overset{1}{\mathsf{A}}$) can express all downward closed ($S_1 \in B$ & $S_2 \subseteq S_1$ imply $S_2 \in B$) B. These results are reformulated and proved using so-called *generalised atoms* in Appendix A. Note that, e.g., k-ary inclusion and dependence atoms can be defined using suitable Boolean relations B. Indeed it follows that, from the expressivity point-of-view, TeamLTL($\otimes, \overset{1}{\mathsf{A}}$) and TeamLTL($\otimes, \sim\bot, \overset{1}{\mathsf{A}}$) subsume all extensions of TeamLTL with downward closed (resp. all) *atomic notions of dependence*, i.e., atoms which state some sort of functional (in)dependence, like the dependence atom (which is downward closed) or the inclusion atom (which is not).

3 Undecidable Extensions of TeamLTL

In [36], Lück established that the model checking problem for TeamLTL($\sim$) is highly undecidable. The proof heavily utilises the interplay between Boolean negation $\sim$ and disjunction $\vee$; it was left as an open problem whether some sensible restrictions on the use of the Boolean negation would lead toward discovering decidable logics. We show that, on the contrary, the decidability bounds are much tighter. Already TeamLTL($\subseteq, \otimes$) (which is subsumed by TeamLTL($\otimes, \sim\bot, \overset{1}{\mathsf{A}}$)) is undecidable, and already very restricted access to $\sim$ (namely, a single use of the A quantifier) leads to high undecidability.

We define the model checking problem based on Kripke structures $\mathrm{K} = (W, R, \eta, w_0)$, where W is a finite set of states, $R \subseteq W^2$ the transition relation, $\eta \colon W \to 2^{\mathrm{AP}}$ a labelling function, and $w_0 \in W$ an initial state of W. A path σ through K is an infinite sequence $\sigma \in W^\omega$ such that $\sigma[0] = w_0$ and $(\sigma[i], \sigma[i+1]) \in R$ for every $i \geq 0$. The trace of σ is defined as $t(\sigma) := \eta(\sigma[0])\eta(\sigma[1])\cdots \in (2^{\mathrm{AP}})^\omega$. A Kripke structure K induces a set of traces $\mathrm{Traces}(\mathrm{K}) = \{t(\sigma) \mid \sigma \text{ is a path through K}\}$.

▶ **Definition 1.** *The* model checking problem of a logic $\mathcal{L}$ *is the following decision problem: Given a formula* $\varphi \in \mathcal{L}$ *and a Kripke structure* K *over* AP*, determine whether* $(\mathit{Traces}(K), 0) \models \varphi$.

Our undecidability results are obtained by reductions from *non-deterministic 3-counter machines*. A non-deterministic 3-counter machine M consists of a list I of n instructions that manipulate three counters C_l, C_m, and C_r. All instructions are of the following forms:

- C_a^+ goto $\{j_1, j_2\}$, $\quad$ C_a^- goto $\{j_1, j_2\}$, $\quad$ if $C_a = 0$ goto $j_1$else goto j_2,

where $a \in \{l, m, r\}$, $0 \leq j_1, j_2 < n$. A *configuration* is a tuple (i, j, k, t), where $0 \leq i < n$ is the next instruction to be executed, and $j, k, t \in \mathbb{N}$ are the current values of the counters C_l, C_m, and C_r. The execution of the instruction $i\colon C_a^+$ goto $\{j_1, j_2\}$ ($i\colon C_a^-$ goto $\{j_1, j_2\}$, resp.) increments (decrements, resp.) the value of the counter C_a by 1. The next instruction is selected nondeterministically from the set $\{j_1, j_2\}$. The instruction $i\colon$ if $C_a = 0$ goto j_1, else goto j_2 checks whether the value of the counter C_a is currently 0 and proceeds to the next instruction accordingly. The *consecution relation* $\rightsquigarrow_{\mathrm{c}}$ of configurations is defined as usual. The *lossy consecution relation* $(i_1, i_2, i_3, i_4) \rightsquigarrow_{\mathrm{lc}} (j_1, j_2, j_3, j_4)$ of configurations holds if $(i_1, i_2', i_3', i_4') \rightsquigarrow_{\mathrm{c}} (j_1, j_2', j_3', j_4')$ holds for some $i_2', i_3', i_4', j_2', j_3', j_4'$ with $i_2 \geq i_2'$, $i_3 \geq i_3'$, $i_4 \geq i_4'$, $j_2' \geq j_2$, $j_3' \geq j_3$, and $j_4' \geq j_4$. A *(lossy) computation* is an infinite sequence of (lossy) consecutive configurations starting from the initial configuration $(0, 0, 0, 0)$. A (lossy) computation is *b-recurring* if the instruction labelled b occurs infinitely often in it. Deciding whether a given non-deterministic 3-counter machine has a b-recurring (b-recurring lossy) computation for a given b is Σ_1^1-complete (Σ_1^0-complete, resp.) [3, 44].

We reduce the existence of a b-recurring lossy computation of a given 3-counter machine M and an instruction label b to the model checking problem of TeamLTL$(\subseteq, \circledvee)$. We also illustrate that with a single instance of A we can enforce non-lossy computation instead.

▶ **Theorem 2.** *Model checking for* TeamLTL$(\subseteq, \circledvee)$ *is* Σ_1^0*-hard.*

Proof. Given a set I of instructions of a 3-counter machine M, and an instruction label b, we construct a TeamLTL$(\subseteq, \circledvee)$-formula $\varphi_{I,b}$ and a Kripke structure K_I such that

$$\big(\mathrm{Traces}(\mathrm{K}_I), 0\big) \models \varphi_{I,b} \quad \text{iff} \quad M \text{ has a } b\text{-recurring lossy computation.} \tag{1}$$

The Σ_1^0-hardness then follows since our construction is clearly computable. The idea is the following: Put $n := |I|$. A set T of traces using propositions $\{c_l, c_m, c_r, d, 0, \ldots, n-1\}$ encodes the sequence $(\vec{c}_j)_{j\in\mathbb{N}}$ of configurations, if for each $j \in \mathbb{N}$ and $\vec{c}_j = (i, v_l, v_m, v_r)$

- $t[j] \cap \{0, \ldots, n-1\} = \{i\}$, for all $t \in T$,
- $|\{t[j, \infty] \mid c_s \in t[j], t \in T\}| = v_s$, for each $s \in \{l, m, r\}$.

Hence, we use $T[j, \infty]$ to encode the configuration $\vec{c}_j$; the propositions $0, \ldots, n-1$ are used to encode the next instruction, and c_l, c_m, c_r, d are used to encode the values of the counters. The proposition d is a dummy proposition used to separate traces with identical postfixes with respect to c_l, c_m, and c_r. The Kripke structure $\mathrm{K}_I = (W, R, \eta, w_0)$ over the set of propositions $\{c_l, c_m, c_r, d, 0, \ldots, n-1\}$ is defined such that every possible sequence of configurations of M starting from $(0, 0, 0, 0)$ can be encoded by some team $(T, 0)$, where $T \subseteq \mathrm{Traces}(\mathrm{K}_I)$. A detailed construction of the formula $\varphi_{I,b}$ and the Kripke structure K_I together with a detailed proof for the fact that (1) indeed holds can be found in Appendix B. ◀

The underlying reason for utilising lossy computations in the above proof is the following: In our encoding, we use the cardinality of the set $|\{t[j,\infty] \mid c_l \in t[j], t \in T\}|$ to encode the value of the counter C_l in the jth configuration. It might, however, happen that two distinct traces $t, t' \in T$ have the same postfix, that is, $t[j,\infty] = t'[j,\infty]$, for some $j \in \mathbb{N}$. The collapse of two traces encoding distinct increments of the counter C_l then corresponds to the uncontrollable decrement of the counter values in lossy computations. Using the universal team quantifier A we can forbid this effect, and encode non-lossy computations. The proof of the following theorem can be found in Appendix B.

▶ **Theorem 3.** *Model checking for* $\mathrm{TeamLTL}(\subseteq, \otimes, \mathsf{A})$ *is* Σ^1_1*-hard. This holds already for the fragment with a single occurrence of* A.

As is common in the LTL-setting, the model checking problem of $\mathrm{TeamLTL}(\subseteq, \otimes)$ can be embedded in its satisfiability problem using auxiliary propositions and $\mathrm{TeamLTL}(\subseteq, \otimes)$-formulae. A formula φ is satisfiable, if there exists a non-empty T such that $(T, 0) \models \varphi$. We thus obtain the following corollary, which is also proven in Appendix B.

▶ **Corollary 4.** *The satisfiability problems for* $\mathrm{TeamLTL}(\subseteq, \otimes)$ *and* $\mathrm{TeamLTL}(\subseteq, \otimes, \mathsf{A})$ *are* Σ^0_1*-hard and* Σ^1_1*-hard, resp.*

4 Quantification-based Hyperlogics and Team Semantics

In this section, we define those quantification-based hyperlogics against which we compare TeamLTL in the rest of the paper. TeamLTL and HyperLTL are known to have orthogonal expressivity [34] but apart from that, nothing is known about the relationship between the different variants of TeamLTL and other temporal hyperlogics such as HyperQPTL [41, 9]. We aim to identify fragments of the logics with similar expressivity to better understand the relative expressivity of TeamLTL for the specification of hyperproperties.

HyperQPTL$^+$ [16] is a temporal logic for hyperproperties. It subsumes HyperLTL and HyperQPTL, so we proceed to give a definition of HyperQPTL$^+$ and define the latter logics as fragments. HyperQPTL$^+$ extends LTL with explicit trace quantification and quantification of atomic propositions. As such, it also subsumes QPTL, which can express all ω-regular properties. Fix an infinite set $\mathcal{V}$ of trace variables. HyperQPTL$^+$ has three types of quantifiers, one for traces and two for propositional quantification.

$$\varphi ::= \forall \pi.\, \varphi \mid \exists \pi.\, \varphi \mid \overset{u}{\forall} p.\, \varphi \mid \overset{u}{\exists} p.\, \varphi \mid \forall p.\, \varphi \mid \exists p.\, \varphi \mid \psi$$
$$\psi ::= p_\pi \mid \neg p_\pi \mid \psi \vee \psi \mid \psi \wedge \psi \mid \bigcirc \psi \mid \psi \,\mathcal{U}\, \psi \mid \psi \,\mathcal{W}\, \psi$$

Here, $p \in \mathrm{AP}$, $\pi \in \mathcal{V}$, and $\forall \pi$ and $\exists \pi$ stand for universal and existential trace quantifiers, $\forall p$ and $\exists p$ stand for (non-uniform) propositional quantifiers, and $\overset{u}{\forall} p$ and $\overset{u}{\exists} p$ stand for uniform propositional quantifiers. We also study two syntactic fragments of HyperQPTL$^+$. HyperQPTL is HyperQPTL$^+$ without non-uniform propositional quantifiers, and HyperLTL is HyperQPTL$^+$ without any propositional quantifiers. In the context of HyperQPTL, we also write $\forall p$ and $\exists p$ instead of $\overset{u}{\forall} p$ and $\overset{u}{\exists} p$. For an LTL-formula φ and trace variable π, we let φ_π denote the HyperLTL-formula obtained from φ by replacing all proposition symbols p by their indexed versions p_π. We extend this convention to tuples of formulae as well.

The semantics of HyperQPTL$^+$ is defined over a set T of traces. Intuitively, the atomic formula p_π asserts that p holds on trace π. Uniform propositional quantifications $\overset{u}{\forall} p$ and $\overset{u}{\exists} p$ add an atomic proposition p such that all traces agree on the valuation of p on any given time step i, whereas non-uniform propositional quantifications $\forall p$ and $\exists p$ colour the traces in T in an arbitrary manner. Non-uniform propositional quantification thus implements true second-order quantification, whereas uniform propositional quantification can be interpreted as a quantification of a set of points in time.

A *trace assignment* is a function $\Pi : \mathcal{V} \to T$ that maps each trace variable in $\mathcal{V}$ to some trace in T. A *modified trace assignment* $\Pi[\pi \mapsto t]$ is equal to Π except that $\Pi[\pi \mapsto t](\pi) = t$. For any subset $A \subseteq \mathrm{AP}$, we write $t \upharpoonright A$ for the projection of t on A (i.e., $(t \upharpoonright A)[i] := t[i] \cap A$ for all $i \in \mathbb{N}$). For any two trace assignments Π and Π', we write $\Pi =_A \Pi'$, if $\big(\Pi(\pi) \upharpoonright A\big) = \big(\Pi'(\pi) \upharpoonright A\big)$ for all $\pi \in \mathcal{V}$. Similarly, $T =_A T'$ whenever $\{t \upharpoonright A \mid t \in T\} = \{t \upharpoonright A \mid t \in T'\}$. For a sequence $s \in (2^{\{p\}})^\omega$ over a single propositional variable p, we write $T[p \mapsto s]$ for the set of traces obtained from T by reinterpreting p on all traces as in s while ensuring that $T[p \mapsto s] =_{\mathrm{AP}\setminus\{p\}} T$. We use $\Pi[p \mapsto s]$ accordingly. The satisfaction relation $\Pi, i \models_T \varphi$ for HyperQPTL$^+$-formulae φ is defined as follows:

$$
\begin{array}{lll}
\Pi, i \models_T p_\pi & \text{iff} & p \in \Pi(\pi)[i] \\
\Pi, i \models_T \varphi_1 \vee \varphi_2 & \text{iff} & \Pi, i \models_T \varphi_1 \text{ or } \Pi, i \models_T \varphi_2 \\
\Pi, i \models_T \neg p_\pi & \text{iff} & p \notin \Pi(\pi)[i] \\
\Pi, i \models_T \varphi_1 \wedge \varphi_2 & \text{iff} & \Pi, i \models_T \varphi_1 \text{ and } \Pi, i \models_T \varphi_2 \\
\Pi, i \models_T p_\pi & \text{iff} & p \in \Pi(\pi)[i] \\
\Pi, i \models_T \neg p_\pi & \text{iff} & p \notin \Pi(\pi)[i] \\
\Pi, i \models_T \varphi_1 \vee \varphi_2 & \text{iff} & \Pi, i \models_T \varphi_1 \text{ or } \Pi, i \models_T \varphi_2 \\
\Pi, i \models_T \varphi_1 \wedge \varphi_2 & \text{iff} & \Pi, i \models_T \varphi_1 \text{ and } \Pi, i \models_T \varphi_2 \\
\Pi, i \models_T \bigcirc \varphi & \text{iff} & \Pi, i+1 \models_T \varphi \\
\Pi, i \models_T \varphi_1 \,\mathcal{U}\, \varphi_2 & \text{iff} & \exists k \geq i \text{ s.t. } \Pi, k \models_T \varphi_2 \text{ and } \forall m : i \leq m < k \Rightarrow \Pi, m \models_T \varphi_1 \\
\Pi, i \models_T \varphi_1 \,\mathcal{W}\, \varphi_2 & \text{iff} & \forall k \geq i : \Pi, k \models_T \varphi_1 \text{ or } \exists m : i \leq m \leq k : \Pi, m \models_T \varphi_2 \\
\Pi, i \models_T \exists \pi.\, \varphi & \text{iff} & \Pi[\pi \mapsto t], i \models_T \varphi \text{ for some } t \in T \\
\Pi, i \models_T \forall \pi.\, \varphi & \text{iff} & \Pi[\pi \mapsto t], i \models_T \varphi \text{ for all } t \in T \\
\Pi, i \models_T \overset{u}{\exists} p.\, \varphi & \text{iff} & \Pi[p \mapsto s], i \models_{T[p \mapsto s]} \varphi \text{ for some } s \in (2^{\{p\}})^\omega \\
\Pi, i \models_T \overset{u}{\forall} p.\, \varphi & \text{iff} & \Pi[p \mapsto s], i \models_{T[p \mapsto s]} \varphi \text{ for all } s \in (2^{\{p\}})^\omega \\
\Pi, i \models_T \exists p.\, \varphi & \text{iff} & \Pi', i \models_{T'} \varphi \text{ for some } T' \subseteq (2^{\mathrm{AP}})^\omega \text{and } \Pi' \colon \mathcal{V} \to T' \text{ such that} \\
 & & T =_{\mathrm{AP}\setminus\{p\}} T' \text{ and } \Pi =_{\mathrm{AP}\setminus\{p\}} \Pi' \\
\Pi, i \models_T \forall p.\, \varphi & \text{iff} & \Pi', i \models_{T'} \varphi \text{ for all } T' \subseteq (2^{\mathrm{AP}})^\omega \text{and } \Pi' \colon \mathcal{V} \to T' \text{ such that} \\
 & & T =_{\mathrm{AP}\setminus\{p\}} T' \text{ and } \Pi =_{\mathrm{AP}\setminus\{p\}} \Pi'
\end{array}
$$

In the sequel, we describe fragments of HyperQPTL$^+$ by restricting the quantifier prefixes of formulae. We use $\exists_\pi \,/\, \forall_\pi$ to denote trace quantification, $\overset{u}{\exists}_p \,/\, \overset{u}{\forall}_p$ for uniform propositional quantification, and $\exists_p \,/\, \forall_p$ for non-uniform propositional quantification. We use $\exists$ ($\forall$, resp.) if we do not need to distinguish between the different types of existential (universal, resp.) quantifiers. We write Q to refer to both $\exists$ and $\forall$. For a logic L and a regular expression e, we write eL to denote the set of L-formulae whose quantifier prefixes are generated by e. E.g., $\forall^*\exists^*$HyperQPTL refers to HyperQPTL-formulae with quantifier prefix $\{\overset{u}{\forall}_p, \forall_\pi\}^*\{\overset{u}{\exists}_p, \exists_\pi\}^*$.

Next we relate the expressivity of extensions of TeamLTL to fragments of HyperQPTL$^+$. We show that TeamLTL($\mathbin{\mathpalette\make@circled\vee}$, $\overset{1}{\mathsf{A}}$) and TeamLTL($\mathbin{\mathpalette\make@circled\vee}$, $\sim\!\bot$, $\overset{1}{\mathsf{A}}$) can be translated to the prefix fragments $\exists_p \overset{u}{Q}{}^*_p \exists^*_\pi \forall_\pi$ and $\exists_p \overset{u}{Q}{}^*_p \forall_\pi$ of HyperQPTL$^+$. The translations provide insight into the limits of the expressivity of different extensions of TeamLTL. In particular, they show that in order to simulate the generation of subteams with the $\vee$-operator in TeamLTL, one existential second-order quantifier $\exists_p$ is sufficient. Meanwhile, the difference between downward closed team properties and general team properties manifests itself by a different need for trace quantifiers: for downward closed properties, a single $\forall_\pi$ quantifier is enough, whereas in the general case, a $\exists^*_\pi\forall_\pi$ quantifier alternation is needed.

As a prerequisite for the translation, we establish that evaluating TeamLTL($\varovee, \sim\bot, \overset{1}{\mathsf{A}}$)-formulae can only create countably many different teams. For a given team (T, i) and TeamLTL($\varovee, \sim\bot, \overset{1}{\mathsf{A}}$)-formula φ, the verification of $(T, i) \models \varphi$ boils down to checking statements of the form $(S, j) \models \psi$, where $S \in \mathcal{S}_T \subseteq 2^T$ for some set $\mathcal{S}_T$, $j \in \mathbb{N}$, and ψ is an atomic formula, together with expressions of the form $S_1 = S_2 \cup S_3$, where $S_1, S_2, S_3 \in \mathcal{S}_T$. The following lemma, proven in Appendix C, implies that the set $\mathcal{S}_T$ can be fixed as a countable set that depends only on T.

▶ **Lemma 5.** *For every set T of traces over a countable* AP*, there exists a countable $\mathcal{S}_T \subseteq 2^T$ such that, for every* TeamLTL($\varovee, \sim\bot, \overset{1}{\mathsf{A}}$)*-formula φ and $i \in \mathbb{N}$, $(T, i) \models \varphi$ iff $(T, i) \models^* \varphi$, where the satisfaction relation $\models^*$ is defined the same way as $\models$ except that in the semantic clause for $\vee$ we require additionally that the two subteams $T_1, T_2 \in \mathcal{S}_T$.*

Using of the above lemma, we obtain translations from the most interesting extensions of TeamLTL to weak prefix fragments of HyperQPTL$^+$; for details and proofs see Appendix C.

▶ **Theorem 6.** *For every $\varphi \in$* TeamLTL($\varovee, \sim\bot, \overset{1}{\mathsf{A}}$) *there exists an equivalent* HyperQPTL$^+$*-formula φ^* in the $\exists_p \overset{u}{Q}{}^*_p \exists^*_\pi \forall_\pi$ fragment. If $\varphi \in$* TeamLTL($\varovee, \overset{1}{\mathsf{A}}$)*, φ^* can be defined in the $\exists_p \overset{u}{Q}{}^*_p \forall_\pi$ fragment. The size of φ^* is linear w.r.t. the size of φ.*

5 Decidable fragments of TeamLTL

In this section, we further study the expressivity landscape between the frameworks of TeamLTL and HyperLTL. We utilise these connections to prove decidability of the model checking problem of certain variants of TeamLTL. We compare the expressivity of extensions of TeamLTL that satisfy certain semantic invariances to that of $\forall^*$HyperLTL and $\overset{u}{\exists}{}^*_p\forall_\pi$HyperQPTL. Thereby, we provide a partial answer to an open problem posed in [34] concerning the complexity of the model checking problem of TeamLTL and its extensions. The problem is known to be in PSPACE for the fragment of TeamLTL without $\vee$ [34]. However, for TeamLTL with $\vee$, no meaningful upper bounds for the problem was known before. The best previous upper bound could be obtained from TeamLTL($\sim$), for which the problem is highly undecidable [36]. The reason for this lack of results is that developing algorithms for team logics with $\vee$ turned out to be comparatively hard. The main source of difficulty is that the semantic definition of $\vee$ does not yield any reasonable compositional brute force algorithm: the verification of $(T, i) \models \varphi \vee \psi$ with T generated by a finite Kripke structure proceeds by checking that $(T_1, i) \models \varphi$ and $(T_2, i) \models \psi$ for some $T_1 \cup T_2 = T$, but it can well be that T_1 and T_2 cannot be generated from any finite Kripke structure whatsoever. The main results of this section are the decidability of the model checking problems of the *k-coherent fragment* of TeamLTL($\sim$) and the *left-flat fragment* of TeamLTL($\varovee, \overset{1}{\mathsf{A}}$). We obtain inclusions to EXPSPACE by translations to $\forall^*$HyperLTL and $\overset{u}{\exists}{}^*_p\forall_\pi$HyperQPTL.

5.1 The k-coherent fragment and $\forall^*$HyperLTL

The universal fragment of HyperLTL is one of the most studied fragments as it contains the set of *safety* hyperproperties expressible in HyperLTL [8]. In particular, formulae of the form

$\forall\pi_1\ldots\forall\pi_k.\,\psi$ state k-safety properties (if ψ is a safety LTL formula) [14], where non-satisfying trace sets contain bad prefixes of at most k traces. In general, $\forall^k$HyperLTL formulae satisfy the following inherent invariance: $\emptyset, i \models_T \varphi$ iff $\emptyset, i \models_{T'} \varphi$, for all $T' \subseteq T$ s.t. $|T'| \leq k$. That is, a $\forall^k$HyperLTL-formula φ is satisfied by a trace set T, iff it is satisfied by all subsets of T of size at most k. This property is called *k-coherence* in the team semantics literature [31]. The main result of this section is that all k-coherent properties expressible in TeamLTL($\sim$) are expressible in $\forall^k$HyperLTL. This implies that, with respect to trace properties, all logics between TeamLTL($\overset{1}{\mathsf{A}}$) and TeamLTL($\sim$) are equi-expressive to $\forall$HyperLTL, i.e., LTL.

▶ **Definition 7.** *Let $\mathcal{A}$ be any collection of atoms and connectives introduced so far. A formula φ in* TeamLTL($\mathcal{A}$) *is said to be k-*coherent *($k \in \mathbb{N}$) if for every team (T, i),*

$$(T, i) \models \varphi \quad \textit{iff} \quad (S, i) \models \varphi \textit{ for every } S \subseteq T \textit{ with } |S| \leq k.$$

We will next show that, with respect to k-coherent properties, TeamLTL($\sim$) is at most as expressive as $\forall^k$HyperLTL. We define a translation from TeamLTL($\sim$) to $\forall^*$HyperLTL that preserves the satisfaction relation with respect to teams of bounded size. Given a finite set Φ of trace variables, the translation is defined as follows:

$$p^\Phi := \bigwedge_{\pi\in\Phi} p_\pi \qquad (\neg p)^\Phi := \bigwedge_{\pi\in\Phi} \neg p_\pi \qquad (\sim\varphi)^\Phi := \neg\varphi^\Phi$$

$$(\bigcirc\varphi)^\Phi := \bigcirc\varphi^\Phi \qquad (\varphi\wedge\psi)^\Phi := \varphi^\Phi\wedge\psi^\Phi \qquad (\varphi\vee\psi)^\Phi := \bigvee_{\Phi_0\cup\Phi_1=\Phi} \varphi^{\Phi_0}\wedge\psi^{\Phi_1}$$

$$(\varphi\,\mathcal{U}\,\psi)^\Phi := \varphi^\Phi\,\mathcal{U}\,\psi^\Phi \qquad (\varphi\,\mathcal{W}\,\psi)^\Phi := \varphi^\Phi\,\mathcal{W}\,\psi^\Phi$$

where $\neg\varphi^\Phi$ stands for the negation of φ^Φ in negation normal form. The following lemma, from which the subsequent theorem follows, is proved by induction. See Appendix D for detailed proofs.

▶ **Lemma 8.** *Let φ be a formula of* TeamLTL($\sim$) *and $\Phi = \{\pi_1, \ldots, \pi_k\}$ a finite set of trace variables. For any team (T, i) with $|T| \leq k$, any set $S \supseteq T$ of traces, and any assignment Π with $\Pi[\Phi] = T$, we have that $(T, i) \models \varphi$ iff $\Pi, i \models_S \varphi^\Phi$. Furthermore, if φ is downward closed and $T \neq \emptyset$, then $(T, i) \models \varphi$ iff $\emptyset, i \models_T \forall\pi_1\ldots\forall\pi_k.\,\varphi^\Phi$.*

▶ **Theorem 9.** *Every k-coherent property that is definable in* TeamLTL($\sim$) *is also definable in* $\forall^k$HyperLTL*.*

Since model checking for $\forall^*$HyperLTL is PSPACE-complete and its data complexity (model checking with a fixed formula) is NL-complete [19], and since the above translation from TeamLTL($\sim$) to $\forall^*$HyperLTL is exponential for any k, we get the following corollary:

▶ **Corollary 10.** *For any fixed $k \in \mathbb{N}$, the model checking problem for* TeamLTL($\sim$)*, restricted to k-coherent properties, is in* EXPSPACE*, and in* NL *for data complexity.*

Clearly $(T, i) \models \overset{1}{\mathsf{A}}\varphi$ iff $\emptyset, i \models_T \forall\pi.\,\varphi_\pi$ for any $\varphi \in$ LTL, and hence we obtain the following:

▶ **Corollary 11.** *The restriction of* TeamLTL($\overset{1}{\mathsf{A}}$) *to formulae of the form $\overset{1}{\mathsf{A}}\varphi$ is expressively equivalent to* $\forall$HyperLTL*.*

While model checking for k-coherent properties is decidable, checking whether a given formula defines a k-coherent property is not, in general, decidable.

▶ **Theorem 12.** *Checking whether a* TeamLTL($\subseteq, \circledvee$)*-formula is* 1*-coherent is undecidable.*

Proof. The idea of the undecidability proof is as follows: Given any TeamLTL($\subseteq, \otimes$)-formula φ, we can use a simple rewriting rule to obtain an LTL-formula φ^* such that φ is not satisfiable (in the sense of TeamLTL) if and only if φ is 1-coherent and φ^* is not satisfiable (in the LTL-sense). Now, since checking LTL-satisfiability can be done in PSPACE [45] and non-satisfiability for TeamLTL($\subseteq, \otimes$) is Π^0_1-hard by Corollary 4, it follows that checking 1-coherence is Π^0_1-hard as well. For a detailed proof, see Appendix D. ◀

The same holds for any extension of TeamLTL with an undecidable satisfiability or validity problem and whose formulae can be computably translated to TeamLTL while preserving satisfaction over singleton teams.

5.2 The left-flat fragment and HyperQPTL$^+$

In this subsection, we show that formulae φ from the left-flat fragment of TeamLTL($\otimes, \overset{1}{\mathsf{A}}$) (defined below) can be translated to HyperQPTL formulae that are linear in the size of φ. The known model checking algorithm of HyperQPTL [41] then immediately yields a model checking algorithm for the left-flat fragment of TeamLTL($\otimes, \overset{1}{\mathsf{A}}$).

▶ **Definition 13** (The left-flat fragment)**.** *Let $\mathcal{A}$ be a collection of atoms and connectives. A* TeamLTL($\mathcal{A}$)*-formula belongs to the left-flat fragment if in each of its subformulae of the form $\psi \mathcal{U} \varphi$ or $\psi \mathcal{W} \varphi$, ψ is a flat formula (as defined in Section 2).*

Such defined fragment allows for arbitrary use of the $\Diamond$ operator, and therefore remains incomparable to HyperLTL [34]. For instance, $\Diamond \mathrm{dep}(a, b) \vee \Diamond \mathrm{dep}(c, d)$ is a nontrivial formula in this fragment. It states that the set of traces can be partitioned into two parts, one where eventually a determines the value of b, and another where eventually c determines the value of d. The property is not expressible in HyperLTL, because HyperLTL cannot state the property "there is a point in time at which p holds on all (or infinitely many) traces" [5].

It follows from Theorem 12 that checking whether a TeamLTL($\subseteq, \otimes$)-formula belongs to the left-flat fragment is undecidable (as flatness equals 1-coherency). Nevertheless, a decidable syntax for left-flat formulae can be obtained by using the operator $\overset{1}{\mathsf{A}}$. Formulae $\overset{1}{\mathsf{A}}\psi$ are always flat and equivalent to ψ if ψ is flat. Therefore, in Definition 13, instead of imposing the semantic condition of ψ being flat in subformulae $\psi \mathcal{U} \varphi$ and $\psi \mathcal{W} \varphi$, we could require that the subformulae must be of the form $(\overset{1}{\mathsf{A}}\psi) \mathcal{U} \varphi$ or $(\overset{1}{\mathsf{A}}\psi) \mathcal{W} \varphi$.

We now describe a translation from the left-flat fragment of TeamLTL($\otimes, \overset{1}{\mathsf{A}}$) to the $\overset{u}{\exists}{}^*_p \forall_\pi$ fragment of HyperQPTL. In this translation, we make use of the fact that satisfaction of flat formulae φ can be determined with the usual (single-traced) LTL semantics. In the evaluation of φ, it is thus sufficient to consider only finitely many subteams, whose temporal behaviour can be reflected by existentially quantified q-sequences. The quantified sequences refer to points in time, at which subformulae have to hold for a trace to belong to a team.

A left-flat TeamLTL($\otimes, \overset{1}{\mathsf{A}}$)-formula φ will be translated into a formula with existential propositional quantifiers followed by a single trace quantifier. The existential propositional quantifiers either indicate a point in time at which a subformula of φ_i is evaluated or resolve the decision for $\otimes$-choices. For subformulae, we use propositions r^{φ_i}, and if the same subformula occurs multiple times, it is associated with different r^{φ_i}. For the resolution of $\otimes$-choices we use propositions $d^{\varphi \otimes \psi}$. Additionally, r is a free proposition for the point in time at which φ is to be evaluated. The universal quantifier $\forall \pi$ sorts each trace into one of the finitely many teams.

Let $\forall\pi.\,\hat{\psi}$ be the HyperLTL formula given by Theorem 9 for any flat formula ψ (since a flat formula is 1-coherent). We translate φ inductively with respect to r:

$$
\begin{aligned}
[p, r] &\coloneqq \Box(r_\pi \to p_\pi)\\
[\neg p, r] &\coloneqq \Box(r_\pi \to \neg p_\pi)\\
[\bigcirc \varphi, r] &\coloneqq \Box(r_\pi \leftrightarrow \bigcirc r_\pi^\varphi) \wedge [\varphi, r^\varphi]\\
[\overset{1}{\mathsf{A}}\varphi, r] &\coloneqq \Box(r_\pi \to \hat{\varphi})\\
[\varphi \wedge \psi, r] &\coloneqq [\varphi, r] \wedge [\psi, r]\\
[\varphi \vee \psi, r] &\coloneqq [\varphi, r] \vee [\psi, r]\\
[\varphi \varovee \psi, r] &\coloneqq (d_\pi^{\varphi \varovee \psi} \to [\varphi, r]) \wedge (\neg d_\pi^{\varphi \varovee \psi} \to [\psi, r])\\
[\varphi\, \mathcal{W}\, \psi, r] &\coloneqq \Box(r_\pi \to r_\pi^\varphi\, \mathcal{W} (r_\pi^\psi \wedge \bigcirc\Box\neg r_\pi^\psi))
\end{aligned}
$$

Now, let $r^1, \dots r^n$ be the free propositions occurring in $[\varphi, r]$ and π the free trace variable. Define the following $\overset{u}{\exists}{}_p^*\forall_\pi$ HyperQPTL formula: $\overset{u}{\exists} r \overset{u}{\exists} r^1 \dots \overset{u}{\exists} r^n.\, \forall\pi.\, r_\pi \wedge \bigcirc\Box\neg r_\pi \wedge [\varphi, r]$. Correctness of the translation can be argued intuitively as follows. The left-flat formula φ can be evaluated independently from the other traces in a team. Therefore, the operators $\mathcal{U}$ and $\mathcal{W}$, whose right-hand sides argue only about a single point in time, can only generate finitely many teams. Thus, there are only finitely many points of synchronization, all of which are quantified existentially. Every trace fits into one of the teams described by the quantified propositional variables. We verify that the translation is indeed correct in Appendix D. As the construction in Theorem 9 yields a formula $\hat{\varphi}$ whose size is linear in the original formula φ, the translation is obviously linear. We therefore state the following theorem.

▶ **Theorem 14.** *For every formula φ from the left-flat fragment of* TeamLTL($\varovee, \overset{1}{\mathsf{A}}$)*, we can compute an equivalent $\overset{u}{\exists}{}_p^*\forall_\pi$* HyperQPTL *formula of size linear in the size of φ.*

Recall that the model checking problem of HyperLTL formulae with one quantifier alternation is EXPSPACE-complete [19] in the size of the formula, and PSPACE-complete in the size of the Kripke structure [19]. These results directly transfer to HyperQPTL [41] (in which HyperQPTL was called HyperLTL *with extended quantification* instead): for model checking a HyperQPTL formula, the Kripke structure can be extended by two states generating all possible q-sequences. Since the translation from TeamLTL($\varovee, \overset{1}{\mathsf{A}}$) to HyperQPTL yields a formula in the $\overset{u}{\exists}{}_p^*\forall_\pi$ fragment with a single quantifier alternation and preserves the size of the formula, we obtain the following theorem.

▶ **Theorem 15.** *The model checking problem for left-flat* TeamLTL($\varovee, \overset{1}{\mathsf{A}}$)*-formulae is in* EXPSPACE*, and in* PSPACE *for data complexity.*

6 Conclusion

We studied TeamLTL under the synchronous semantics. TeamLTL is a powerful but not yet well-studied logic that can express hyperproperties without explicit quantification over traces or propositions. As such, properties which need various different quantifiers in traditional (quantification-based) hyperlogics become expressible in a concise fashion. One of the main advantages of TeamLTL is the ability to equip it with a range of atomic statements and connectives to obtain logics of varying expressivity and complexity.

We systematically studied TeamLTL with respect to two of the main questions related to logics: the decision boundary of its model checking problem and its expressivity compared to other logics for hyperproperties. We related the expressivity of TeamLTL to the hyperlogics HyperLTL, HyperQPTL, and HyperQPTL$^+$, which are obtained by extending the traditional temporal logics LTL and QPTL with trace quantifiers. We discovered that the logics TeamLTL($\varovee, \overset{1}{\mathsf{A}}$) and TeamLTL($\varovee, \sim\bot, \overset{1}{\mathsf{A}}$) are expressively complete with respect to all downward closed, and all atomic notions of dependence, respectively. We were able to show that TeamLTL($\varovee, \sim\bot, \overset{1}{\mathsf{A}}$) can be expressed in a fragment of HyperQPTL$^+$. Furthermore, for k-coherent properties, TeamLTL($\sim$) is subsumed by $\forall^*$HyperLTL. Finally, the left-flat fragment of TeamLTL($\varovee, \overset{1}{\mathsf{A}}$) can be translated to HyperQPTL. The last two results induce efficient model checking algorithms for the respective logics. In addition, we showed that model checking of TeamLTL($\subseteq, \varovee$) is already undecidable, and that the additional use of the A quantifier makes the problem highly undecidable.

We conclude with some open problems and directions for future work: What is the complexity of model checking for TeamLTL (with the disjunction $\vee$ but without additional atoms and connectives)? Is it decidable, and is there a translation to HyperQPTL? An interesting avenue for future work is also to explore team semantics of more expressive logics than LTL such as linear time μ-calculus, or branching time logics such as CTL* and the full modal μ-calculus.

References

1 Erika Ábrahám and Borzoo Bonakdarpour. Hyperpctl: A temporal logic for probabilistic hyperproperties. In Annabelle McIver and András Horváth, editors, *Quantitative Evaluation of Systems - 15th International Conference, QEST 2018, Beijing, China, September 4-7, 2018, Proceedings*, volume 11024 of *Lecture Notes in Computer Science*, pages 20–35. Springer, 2018. URL: `https://doi.org/10.1007/978-3-319-99154-2_2`, `doi:10.1007/978-3-319-99154-2\_2`.

2 Shreya Agrawal and Borzoo Bonakdarpour. Runtime verification of k-safety hyperproperties in hyperltl. In *IEEE 29th Computer Security Foundations Symposium, CSF 2016, Lisbon, Portugal, June 27 - July 1, 2016*, pages 239–252. IEEE Computer Society, 2016. URL: `https://doi.org/10.1109/CSF.2016.24`, `doi:10.1109/CSF.2016.24`.

3 Rajeev Alur and Thomas A. Henzinger. A really temporal logic. *J. ACM*, 41(1):181–204, 1994. URL: `https://doi.org/10.1145/174644.174651`, `doi:10.1145/174644.174651`.

4 Jan Baumeister, Norine Coenen, Borzoo Bonakdarpour, Bernd Finkbeiner, and César Sánchez. A temporal logic for asynchronous hyperproperties. In Alexandra Silva and K. Rustan M. Leino, editors, *Computer Aided Verification - 33rd International Conference, CAV 2021, Virtual Event, July 20-23, 2021, Proceedings, Part I*, volume 12759 of *Lecture Notes in Computer Science*, pages 694–717. Springer, 2021. URL: `https://doi.org/10.1007/978-3-030-81685-8_33`, `doi:10.1007/978-3-030-81685-8\_33`.

5 Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Unifying hyper and epistemic temporal logics. In *Proceedings of FoSSaCS*, volume 9034 of *LNCS*, pages 167–182. Springer, 2015. `doi:10.1007/978-3-662-46678-0\_11`.

6 Alessandro Cimatti, Edmund M. Clarke, Fausto Giunchiglia, and Marco Roveri. NuSMV: A new symbolic model verifier. In *Proc. International Conference on Computer Aided Verification*, pages 495–499, July 1999.

7 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 *Lecture Notes in Computer Science*, pages 265–284. Springer, 2014. `doi:10.1007/978-3-642-54792-8_15`.

8 Michael R. Clarkson and Fred B. Schneider. Hyperproperties. *Journal of Computer Security*, 18(6):1157–1210, 2010. `doi:10.3233/JCS-2009-0393`.

9 Norine Coenen, Bernd Finkbeiner, Christopher Hahn, and Jana Hofmann. The hierarchy of hyperlogics. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science,*

LICS 2019, Vancouver, BC, Canada, June 24-27, 2019, pages 1–13. IEEE, 2019. URL: `https://doi.org/10.1109/LICS.2019.8785713`, `doi:10.1109/LICS.2019.8785713`.

10 Norine Coenen, Bernd Finkbeiner, Christopher Hahn, Jana Hofmann, and Yannick Schillo. Runtime enforcement of hyperproperties. *To appear at the 19th International Symposium on Automated Technology for Verification and Analysis (ATVA 2021)*, 2021.

11 Byron Cook, Eric Koskinen, and Moshe Vardi. Temporal property verification as a program analysis task. In *Proc. Computer Aided Verification*, pages 333–348, July 2011.

12 Jukka Corander, Antti Hyttinen, Juha Kontinen, Johan Pensar, and Jouko Väänänen. A logical approach to context-specific independence. *Ann. Pure Appl. Logic*, 170(9):975–992, 2019. URL: `https://doi.org/10.1016/j.apal.2019.04.004`, `doi:10.1016/j.apal.2019.04.004`.

13 Arnaud Durand, Miika Hannula, Juha Kontinen, Arne Meier, and Jonni Virtema. Probabilistic team semantics. In *FoIKS*, volume 10833 of *Lecture Notes in Computer Science*, pages 186–206. Springer, 2018. `doi:10.1007/978-3-319-90050-6\_11`.

14 Bernd Finkbeiner, Lennart Haas, and Hazem Torfah. Canonical representations of k-safety hyperproperties. In *32nd IEEE Computer Security Foundations Symposium, CSF 2019, Hoboken, NJ, USA, June 25-28, 2019*, pages 17–31. IEEE, 2019. URL: `https://doi.org/10.1109/CSF.2019.00009`, `doi:10.1109/CSF.2019.00009`.

15 Bernd Finkbeiner and Christopher Hahn. Deciding hyperproperties. In Josée Desharnais and Radha Jagadeesan, editors, *27th International Conference on Concurrency Theory, CONCUR 2016, August 23-26, 2016, Québec City, Canada*, volume 59 of *LIPIcs*, pages 13:1–13:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. URL: `https://doi.org/10.4230/LIPIcs.CONCUR.2016.13`, `doi:10.4230/LIPIcs.CONCUR.2016.13`.

16 Bernd Finkbeiner, Christopher Hahn, Jana Hofmann, and Leander Tentrup. Realizing ømega-regular hyperproperties. In Shuvendu K. Lahiri and Chao Wang, editors, *Computer Aided Verification - 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21-24, 2020, Proceedings, Part II*, volume 12225 of *Lecture Notes in Computer Science*, pages 40–63. Springer, 2020. URL: `https://doi.org/10.1007/978-3-030-53291-8_4`, `doi:10.1007/978-3-030-53291-8\_4`.

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

18 Bernd Finkbeiner, Christopher Hahn, Marvin Stenger, and Leander Tentrup. Monitoring hyperproperties. *Formal Methods Syst. Des.*, 54(3):336–363, 2019. URL: `https://doi.org/10.1007/s10703-019-00334-z`, `doi:10.1007/s10703-019-00334-z`.

19 Bernd Finkbeiner, Markus N. Rabe, and César Sánchez. Algorithms for model checking HyperLTL and HyperCTL*. In *Proceedings of CAV*, volume 9206 of *LNCS*, pages 30–48. Springer, 2015. `doi:10.1007/978-3-319-21690-4\_3`.

20 Pietro Galliani and Lauri Hella. Inclusion Logic and Fixed Point Logic. In *CSL 2013*, pages 281–295, 2013.

21 Giuseppe De Giacomo, Paolo Felli, Marco Montali, and Giuseppe Perelli. Hyperldlf: a logic for checking properties of finite traces process logs. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 1859–1865. ijcai.org, 2021. URL: `https://doi.org/10.24963/ijcai.2021/256`, `doi:10.24963/ijcai.2021/256`.

22 Jens Oliver Gutsfeld, Markus Müller-Olm, and Christoph Ohrem. Automata and fixpoints for asynchronous hyperproperties. *Proc. ACM Program. Lang.*, 5(POPL):1–29, 2021. URL: `https://doi.org/10.1145/3434319`, `doi:10.1145/3434319`.

23 Miika Hannula and Juha Kontinen. A finite axiomatization of conditional independence and inclusion dependencies. *Inf. Comput.*, 249:121–137, 2016. `doi:10.1016/j.ic.2016.04.001`.

24 Miika Hannula, Juha Kontinen, Jan Van den Bussche, and Jonni Virtema. Descriptive complexity of real computation and probabilistic independence logic. In Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller, editors, *LICS '20: 35th Annual ACM/IEEE*

Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020, pages 550–563. ACM, 2020. URL: https://doi.org/10.1145/3373718.3394773, doi:10.1145/3373718.3394773.

25 Miika Hannula, Juha Kontinen, Jonni Virtema, and Heribert Vollmer. Complexity of propositional logics in team semantic. *ACM Trans. Comput. Log.*, 19(1):2:1–2:14, 2018. doi:10.1145/3157054.

26 Lauri Hella, Antti Kuusisto, Arne Meier, and Jonni Virtema. Model checking and validity in propositional and modal inclusion logics. *J. Log. Comput.*, 29(5):605–630, 2019. URL: https://doi.org/10.1093/logcom/exz008, doi:10.1093/logcom/exz008.

27 Lauri Hella, Kerkko Luosto, Katsuhiko Sano, and Jonni Virtema. The expressive power of modal dependence logic. In Rajeev Goré, Barteld P. Kooi, and Agi Kurucz, editors, *Advances in Modal Logic 10, invited and contributed papers from the tenth conference on "Advances in Modal Logic," held in Groningen, The Netherlands, August 5-8, 2014*, pages 294–312. College Publications, 2014. URL: http://www.aiml.net/volumes/volume10/Hella-Luosto-Sano-Virtema.pdf.

28 Hsi-Ming Ho, Ruoyu Zhou, and Timothy M. Jones. On verifying timed hyperproperties. In Johann Gamper, Sophie Pinchinat, and Guido Sciavicco, editors, *26th International Symposium on Temporal Representation and Reasoning, TIME 2019, October 16-19, 2019, Málaga, Spain*, volume 147 of *LIPIcs*, pages 20:1–20:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. URL: https://doi.org/10.4230/LIPIcs.TIME.2019.20, doi:10.4230/LIPIcs.TIME.2019.20.

29 Gerard J. Holzmann. The model checker SPIN. *IEEE Transactions on Software Engineering*, 23:279–295, 1997.

30 Tapani Hyttinen, Gianluca Paolini, and Jouko Väänänen. A Logic for Arguing About Probabilities in Measure Teams. *Arch. Math. Logic*, 56(5-6):475–489, 2017. doi:10.1007/s00153-017-0535-x.

31 Jarmo Kontinen. Coherence and computational complexity of quantifier-free dependence logic formulas. *Studia Logica*, 101(2):267–291, 2013. URL: http://dx.doi.org/10.1007/s11225-013-9481-8, doi:10.1007/s11225-013-9481-8.

32 Juha Kontinen, Julian-Steffen Müller, Henning Schnoor, and Heribert Vollmer. A van benthem theorem for modal team semantics. In Stephan Kreutzer, editor, *24th EACSL Annual Conference on Computer Science Logic, CSL 2015, September 7-10, 2015, Berlin, Germany*, volume 41 of *LIPIcs*, pages 277–291. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015. URL: https://doi.org/10.4230/LIPIcs.CSL.2015.277, doi:10.4230/LIPIcs.CSL.2015.277.

33 Andreas Krebs, Arne Meier, and Jonni Virtema. A team based variant of CTL. In Fabio Grandi, Martin Lange, and Alessio Lomuscio, editors, *22nd International Symposium on Temporal Representation and Reasoning, TIME 2015, Kassel, Germany, September 23-25, 2015*, pages 140–149. IEEE Computer Society, 2015. URL: https://doi.org/10.1109/TIME.2015.11, doi:10.1109/TIME.2015.11.

34 Andreas Krebs, Arne Meier, Jonni Virtema, and Martin Zimmermann. Team Semantics for the Specification and Verification of Hyperproperties. In Igor Potapov, Paul Spirakis, and James Worrell, editors, *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018)*, volume 117 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:16, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.MFCS.2018.10.

35 Antti Kuusisto. A double team semantics for generalized quantifiers. *Journal of Logic, Language and Information*, 24(2):149–191, 2015. doi:10.1007/s10849-015-9217-4.

36 Martin Lück. On the complexity of linear temporal logic with team semantics. *Theoretical Computer Science*, 2020. doi:https://doi.org/10.1016/j.tcs.2020.04.019.

37 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, January 13-16, 2020, Barcelona, Spain*, volume 152 of *LIPIcs*, pages 29:1–29:16. Schloss Dagstuhl - Leibniz-Zentrum

für Informatik, 2020. URL: `https://doi.org/10.4230/LIPIcs.CSL.2020.29`, `doi:10.4230/LIPIcs.CSL.2020.29`.

38 John McLean. Proving noninterference and functional correctness using traces. *Journal of Computer Security*, 1(1):37–58, 1992. URL: `https://doi.org/10.3233/JCS-1992-1103`, `doi:10.3233/JCS-1992-1103`.

39 Nir Piterman and Amir Pnueli. Temporal logic and fair discrete systems. In Edmund M. Clarke, Thomas A. Henzinger, Helmut Veith, and Roderick Bloem, editors, *Handbook of Model Checking*, pages 27–73. Springer, 2018. URL: `https://doi.org/10.1007/978-3-319-10575-8_2`, `doi:10.1007/978-3-319-10575-8\_2`.

40 Amir Pnueli. The Temporal Logic of Programs. In *FOCS 1977*, pages 46–57, 1977.

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

42 A. W. Roscoe. CSP and determinism in security modelling. In *Proceedings of the 1995 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 8-10, 1995*, pages 114–127. IEEE Computer Society, 1995. URL: `https://doi.org/10.1109/SECPRI.1995.398927`, `doi:10.1109/SECPRI.1995.398927`.

43 Katsuhiko Sano and Jonni Virtema. Characterising modal definability of team-based logics via the universal modality. *Ann. Pure Appl. Log.*, 170(9):1100–1127, 2019. URL: `https://doi.org/10.1016/j.apal.2019.04.009`, `doi:10.1016/j.apal.2019.04.009`.

44 Philippe Schnoebelen. Lossy counter machines decidability cheat sheet. In Antonín Kucera and Igor Potapov, editors, *Reachability Problems, 4th International Workshop, RP 2010, Brno, Czech Republic, August 28-29, 2010. Proceedings*, volume 6227 of *Lecture Notes in Computer Science*, pages 51–75. Springer, 2010. URL: `https://doi.org/10.1007/978-3-642-15349-5_4`, `doi:10.1007/978-3-642-15349-5\_4`.

45 A. Prasad Sistla and Edmund M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, 1985. URL: `https://doi.org/10.1145/3828.3837`, `doi:10.1145/3828.3837`.

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

47 Jouko Väänänen. Modal dependence logic. In *New Perspectives on Games and Interaction*, 2008.

48 Fan Yang and Jouko Väänänen. Propositional logics of dependence. *Annals of Pure and Applied Logic*, 167(7):557 – 589, 2016. URL: `http://www.sciencedirect.com/science/article/pii/S0168007216300203`, `doi:https://doi.org/10.1016/j.apal.2016.03.003`.

49 Fan Yang and Jouko Väänänen. Propositional team logics. *Annals of Pure and Applied Logic*, 168(7):1406 – 1441, 2017. URL: `http://www.sciencedirect.com/science/article/pii/S0168007217300088`, `doi:https://doi.org/10.1016/j.apal.2017.01.007`.

50 Steve Zdancewic and Andrew C. Myers. Observational determinism for concurrent program security. In *16th IEEE Computer Security Foundations Workshop (CSFW-16 2003), 30 June - 2 July 2003, Pacific Grove, CA, USA*, page 29. IEEE Computer Society, 2003. URL: `https://doi.org/10.1109/CSFW.2003.1212703`, `doi:10.1109/CSFW.2003.1212703`.

A Generalised Atoms in TeamLTL

Section 3 established that already very weak fragments of TeamLTL($\sim$) have highly undecidable model checking. One way to obtain expressive, but computationally well behaved, team-logics for hyperproperties is to carefully introduce novel atoms stating those atomic properties that are of the most interest, and then seeing whether decidable logics, or fragments, with those atoms can be uncovered. In Section 2 we already saw a few examples of important atomic statements; namely dependence and inclusion atoms. More generally, arbitrary properties of teams induce *generalised atoms*. These atoms were first introduced, in the first-order team semantics setting, by Kuusisto [35] using generalised quantifiers. Propositional generalised atoms essentially encode second-order truth tables.

▶ **Definition 16** (Generalised atoms for LTL). *An n-ary* generalised atom *is an n-ary operator $\#_G(\varphi_1, \dots, \varphi_n)$, with an associated nonempty set G of n-ary relations over the Boolean domain $\{0,1\}$, that applies only to* LTL*-formulae $\varphi_1, \dots, \varphi_n$. Its team semantics is defined as:*

$$(T, i) \models \#_G(\varphi_1, \dots, \varphi_n) \quad \text{iff} \quad \{(\llbracket\varphi_1\rrbracket_{(t,i)}, \dots, \llbracket\varphi_n\rrbracket_{(t,i)}) \mid t \in T\} \in G.$$

E.g., dependence atoms of type $\mathrm{dep}(\varphi, \psi)$ can be expressed as generalised atoms via the second-order truth table $G := \{A \subseteq \{0,1\}^2 \mid \text{ if } (a, b_1), (a, b_2) \in A \text{ then } b_1 = b_2\}$. Thus, when considered as a generalised atom, the dependence atom is, in fact, a collection of generalised atoms; one for each arity. Note that $\overset{1}{\mathsf{A}}$ can be interpret as a downward closed generalised atom where $G := \{\emptyset, \{1\}\}$, when applied to TeamLTL-formulae. From now on, we treat $\overset{1}{\mathsf{A}}$ as a generalised atom. Often this distinction has no effect, for example, $\mathrm{TeamLTL}(\ovee, \overset{1}{\mathsf{A}})$-formula $\overset{1}{\mathsf{A}}\varphi$ is equivalent with the $\mathrm{TeamLTL}(\overset{1}{\mathsf{A}})$-formula $\overset{1}{\mathsf{A}}\varphi'$, where φ' is obtained from φ by replacing all $\ovee$ by $\vee$, and simply eliminating the symbols $\overset{1}{\mathsf{A}}$ from the formula. We denote by $\mathcal{A}_{\mathrm{all}}$ ($\mathcal{A}_{\mathrm{dc}}$, resp.) the collection of all (all downward closed, resp.) generalised atoms.

It is straightforward to verify (by induction) that for any collection $\mathcal{A}$ of downward closed atoms and connectives (a connective is downward closed, if it preserves downward closure), the logic $\mathrm{TeamLTL}(\mathcal{A})$ is downward closed as well. For instance, $\mathrm{TeamLTL}(\mathrm{dep}, \ovee, \overset{1}{\mathsf{A}})$ is downward closed.

Propositional and modal logic with $\ovee$ are known to be expressively complete with respect to nonempty downward closed properties (that are closed under so-called *team bisimulations*)[27, 48]. The next proposition establishes that, in the LTL setting, $\mathrm{TeamLTL}(\ovee, \overset{1}{\mathsf{A}})$ is very expressive among the downward closed logics; all downward closed atoms can be expressed in the logic. The translation given in the proposition is inspired by an analogous one given in [48] for propositional team logics.

▶ **Proposition 17.** *For any n-ary generalised atom $\#_G$ and* LTL*-formulae $\varphi_1, \dots, \varphi_n$, we have that*

$$\#_G(\varphi_1, \dots, \varphi_n) \equiv \bigovee_{R \in G} \bigvee_{(b_1, \dots, b_n) \in R} \big(\overset{1}{\mathsf{A}}(\varphi_1^{b_1} \wedge \cdots \wedge \varphi_n^{b_n}) \wedge {\sim}\bot\big)$$

where $\varphi_i^1 := \varphi_i$ and $\varphi_i^0 := \neg\varphi_i$ in negation normal form. If $\#_G$ is downward closed, the above translation can be simplified to

$$\#_G(\varphi_1, \dots, \varphi_n) \equiv \bigovee_{R \in G} \bigvee_{(b_1, \dots, b_n) \in R} \overset{1}{\mathsf{A}}(\varphi_1^{b_1} \wedge \cdots \wedge \varphi_n^{b_n}).$$

Consequently, $\mathrm{TeamLTL}(\mathcal{A}_{\mathrm{dc}}, \ovee) \equiv \mathrm{TeamLTL}(\ovee, \overset{1}{\mathsf{A}})$ *and* $\mathrm{TeamLTL}(\mathcal{A}_{\mathrm{all}}, \ovee) \equiv \mathrm{TeamLTL}(\ovee, \overset{1}{\mathsf{A}}, {\sim}\bot) \leq \mathrm{TeamLTL}({\sim})$*, where ${\sim}\bot$ is read as a generalised atom stating that the team is non-empty. The elimination of each atom yields a doubly exponential disjunction over linear sized formulae.*

Proof. For any $\vec{b} = (b_1, \dots, b_n) \in R \in G$, put

$$\vec{\varphi}^{\vec{b}} = \varphi_1^{b_1} \wedge \cdots \wedge \varphi_n^{b_n}.$$

Put also

$$[\![\vec{\varphi}]\!]_{(t,i)} = ([\![\varphi_1]\!]_{(t,i)}, \ldots, [\![\varphi_n]\!]_{(t,i)}).$$

for any trace t. Now, by the empty team property, we have

$$\begin{aligned}(S,i) \models \overset{1}{\mathsf{A}}\vec{\varphi}^{\vec{b}} &\Leftrightarrow S = \emptyset \text{ or } \forall t \in S : [\![\vec{\varphi}]\!]_{(t,i)} = \vec{b} \\ &\Leftrightarrow \{[\![\vec{\varphi}]\!]_{(t,i)} \mid t \in S\} \subseteq \{\vec{b}\}.\end{aligned}$$

Thus,

$$\begin{aligned}&(T,i) \models \bigvee_{\vec{b}\in R} \overset{1}{\mathsf{A}}\vec{\varphi}^{\vec{b}} \\ \Leftrightarrow\ & \forall \vec{b} \in R, \exists T_{\vec{b}} \text{ s.t. } T = \bigcup_{\vec{b}\in R} T_{\vec{b}} \text{ and } \{[\![\vec{\varphi}]\!]_{(t,i)} \mid t \in T_{\vec{b}}\} \subseteq \{\vec{b}\} \\ \Leftrightarrow\ & \{[\![\vec{\varphi}]\!]_{(t,i)} \mid t \in T\} \subseteq R.\end{aligned}$$

Since $(S,i) \models \sim\bot \Leftrightarrow S \neq \emptyset$, we have similarly that

$$(S,i) \models \overset{1}{\mathsf{A}}\vec{\varphi}^{\vec{b}} \wedge \sim\bot \ \Leftrightarrow\ \{[\![\vec{\varphi}]\!]_{(t,i)} \mid t \in S\} = \{\vec{b}\},$$

and thus

$$(T,i) \models \bigvee_{\vec{b}\in R} (\overset{1}{\mathsf{A}}\vec{\varphi}^{\vec{b}} \wedge \sim\bot) \Leftrightarrow \{[\![\vec{\varphi}]\!]_{(t,i)} \mid t \in T\} = R.$$

Finally, if $\#_G$ is downward closed, then

$$\begin{aligned}&(T,i) \models \bigovee_{R\in G} \bigvee_{\vec{b}\in R} \overset{1}{\mathsf{A}}\vec{\varphi}^{\vec{b}} \\ \Leftrightarrow\ & (T,i) \models \bigvee_{\vec{b}\in R} \overset{1}{\mathsf{A}}\vec{\varphi}^{\vec{b}} \text{ for some } R \in G \\ \Leftrightarrow\ & \{[\![\vec{\varphi}]\!]_{(t,i)} \mid t \in T\} \subseteq R \text{ for some } R \in G \\ \Leftrightarrow\ & \{[\![\vec{\varphi}]\!]_{(t,i)} \mid t \in T\} \in G && (\because \#_G \text{ is downward closed}) \\ \Leftrightarrow\ & (T,i) \models \#_G(\varphi_1, \ldots, \varphi_n).\end{aligned}$$

Similarly, if no additional condition is assumed for $\#_G$, then $(T,i) \models \bigovee_{R\in G} \bigvee_{\vec{b}\in R} (\overset{1}{\mathsf{A}}\vec{\varphi}^{\vec{b}} \wedge \sim\bot) \Leftrightarrow$ $(T,i) \models \#_G(\varphi_1, \ldots, \varphi_n)$. ◀

The operator $\overset{1}{\mathsf{A}}$ used in the above translation is essential for the result. We now illustrate with an example that the logic TeamLTL($\varovee$) without the operator $\overset{1}{\mathsf{A}}$ is strictly less expressive than TeamLTL($\varovee, \overset{1}{\mathsf{A}}$) or TeamLTL($\mathcal{A}_{\mathrm{dc}}, \varovee$).

▶ **Proposition 18.** *The formula* $\overset{1}{\mathsf{A}} \Diamond p$ *is not expressible in* TeamLTL($\varovee$). *Thus* TeamLTL($\varovee$) $<$ TeamLTL($\varovee, \overset{1}{\mathsf{A}}$).

Proof. Define trace sets

$$T_{\mathbb{N}^+} := \{\, \{\}^n \{p\} \{\}^\omega \mid n \in \mathbb{N} \,\}, \quad T_0 := \{\, \{\}^\omega \,\},$$

and $T_{\mathbb{N}} := T_{\mathbb{N}^+} \cup T_0$. Clearly $(T_{\mathbb{N}^+}, 0) \models \overset{1}{\mathsf{A}} \Diamond p$ and $(T_{\mathbb{N}}, 0) \not\models \overset{1}{\mathsf{A}} \Diamond p$. We now show, by induction, that for every TeamLTL($\varovee$)-formula φ, $i \in \mathbb{N}$, and infinite subset $T \subseteq T_{\mathbb{N}^+}$,

$$(T, i) \models \varphi \Leftrightarrow (T \cup T_0, i) \models \varphi.$$

Since TeamLTL($\varovee$) is downward closed, it suffices to show the " $\Rightarrow$ " direction. The base case that $\varphi = p$ or $\neg p$ follows from the observation that $(S, i) \not\models p$ and $(S, i) \not\models \neg p$, for any infinite $S \subseteq T_{\mathbb{N}}$. We now only give details for the most interesting inductive cases.

If $(T, i) \models \varphi \vee \psi$, then there exist $T_1, T_2 \subseteq T$ s.t. $T_1 \cup T_2 = T$, $(T_1, i) \models \varphi$ and $(T_2, i) \models \psi$, where either T_1 or T_2 is infinite, as T is infinite. W.l.o.g. we way assume that T_1 is infinite. Now by induction hypothesis, we have that

$$(T_1 \cup T_0) \cup T_2 = T \cup T_0, \ (T_1 \cup T_0, i) \models \varphi, \text{ and } (T_2, i) \models \psi.$$

Therefore, $(T \cup T_0, i) \models \varphi \vee \psi$.

If $(T, i) \models \varphi \mathcal{U} \psi$, then there exists $k \geq i$ s.t. $(T, k) \models \psi$ and $(T, j) \models \varphi$ whenever $i \leq j < k$. By induction hypothesis, we have that $(T \cup T_0, k) \models \psi$ and $(T \cup T_0, j) \models \varphi$ whenever $i \leq j < k$. Hence, we conclude that $(T \cup T_0, i) \models \varphi \mathcal{U} \psi$.

◀

It is a facinating open problem whether variants of the infamous Kamp's theorem can be developed for TeamLTL($\varovee, \overset{1}{\mathsf{A}}$) and TeamLTL($\sim$). In the modal team semantics setting variants of the famous van Benthem-Rosen characterisation theorem have been shown for the modal logic analogues ML($\varovee$) [27] and ML($\sim$) [32] of TeamLTL($\varovee, \overset{1}{\mathsf{A}}$) and TeamLTL($\sim$), respectively.

B Missing Proofs of Section 3

▷ Claim 19. The claim (1) on page 7 holds.

Proof. We start by defining the formula $\varphi_{I,b}$ that enforces that the configurations encoded by $T[i, \infty]$, $i \in \mathbb{N}$, encode an accepting computation of the counter machine.

Define $\varphi_{I,b} := (\theta_{\text{comp}} \wedge \theta_{b-\text{rec}}) \vee_{\text{L}} \top$, where $\vee_{\text{L}}$ is a shorthand for the following condition:

$$(T, i) \models \varphi \vee_{\text{L}} \psi \text{ iff } \exists T_1, T_2 \text{ s.t. } T_1 \neq \emptyset, \ T_1 \cup T_2 = T, (T_1, i) \models \varphi \text{ and } (T_2, i) \models \psi.$$

The disjunction $\vee_{\text{L}}$ can be defined using $\subseteq$, $\vee$, and a built-in trace $\{p\}^\omega$, where p is a fresh proposition (see e.g., [26, Lemma 3.4]). The formula $\theta_{b-\text{rec}} := \Box \Diamond b$ describes the b-recurrence condition of the computation. The other formula θ_{comp}, which we define below in steps, states that the encoded computation is a legal one.

First, define

$$\text{singleton} := \Box \bigwedge_{a \in \text{AP}} (a \varovee \neg a), \qquad \text{c}_s\text{-decrease} := c_s \vee (\neg c_s \wedge \bigcirc \neg c_s), \text{ for } s \in \{l, m, r\}.$$

The intuitive idea behind the above formulae are as follows: A team satisfying the formula singleton contains at most a single trace with respect to the propositions in AP. If a team (T, i) satisfies c_s-decrease, then the number of traces in $T[i+1, \infty]$ satisfying c_s is less or equal to the number of traces in $T[i, \infty]$ satisfying c_s. In our encoding of counters this would mean that the value of the counter c in the configuration $\vec{c}_{i+1}$ is less or equal to its value in the configuration $\vec{c}_i$. Thus c_s-decrease will be handy below for encoding lossy computation.

Next, for each instruction label i, we define a formula θ_i describing the result of the execution of the instruction:

- For the instruction $i\colon C_l^+$ goto $\{j,j'\}$, define
$$\theta_i := \bigcirc(j \otimes j') \wedge \big((\text{singleton} \wedge \neg c_l \wedge \bigcirc c_l) \vee \mathrm{c_l}\text{-decrease}\big) \wedge \mathrm{c_r}\text{-decrease} \wedge \mathrm{c_m}\text{-decrease}\,.$$
- For the instruction $i\colon C_l^-$ goto $\{j,j'\}$, define
$$\theta_i := \bigcirc(j \otimes j') \wedge \big((c_l \wedge \bigcirc \neg c_l) \vee_{\mathrm{L}} \mathrm{c_l}\text{-decrease}\big) \wedge \mathrm{c_r}\text{-decrease} \wedge \mathrm{c_m}\text{-decrease}\,.$$
- For the instrunctions $i\colon C_s^+$ goto $\{j,j'\}$ and $i\colon C_s^-$ goto $\{j,j'\}$ with $s \in \{m,r\}$, the formulae θ_i are defined analogously with the indices l, m, and r permuted.
- For the instruction i: if $C_s = 0$ goto j, else goto j', define
$$\theta_i := \big(\bigcirc(\neg c_s \wedge j) \otimes (\top \subseteq c_s \wedge \bigcirc j')\big) \wedge \mathrm{c_l}\text{-decrease} \wedge \mathrm{c_m}\text{-decrease} \wedge \mathrm{c_r}\text{-decrease}\,.$$

Finally, define $\theta_{\text{comp}} := \Box \bigotimes_{i<n}(i \wedge \theta_i)$. We next describe the intuition of the above formulae. The left-most conjunct of θ_i for $i\colon C_l^+$ goto $\{j,j'\}$ expresses that after executing the instruction i, the label of the next instruction is either j or j'. The third and the fourth conjunct express that the values of counters C_r and C_m will not increase, but might decrease. The second conjunct expresses that the value of the counter C_l might increase by one, stay the same, or decrease. The meaning of θ_i for $i\colon C_l^-$ goto $\{j,j'\}$ is similar. Finally, the formula θ_i for i: if $C_s = 0$ goto j, else goto j' expresses that, in the lossy execution of i, a) the values of the counters C_l, C_m, and C_r might decrease, but cannot increase, b) the next instruction is either j or j', c) if the next instruction is j then the value of the counter C_s, after the lossy execution, is 0, and d) if the next instruction is j' then the value of the counter C_s, before the lossy execution, was not 0.

Next, we define the Kripke structure $\mathrm{K}_I = (W, R, \eta, w_0)$ over the set of propositions $\{c_l, c_m, c_r, d, 0, \dots, n-1\}$. The structure is defined such that every possible sequence of configurations of M starting from $(0,0,0,0)$ can be encoded by some team $(T,0)$, where $T \subseteq \mathrm{Traces}(\mathrm{K}_I)$. Define $W := \{(i,j,k,t,l) \mid 0 \le i < n \text{ and } j,k,t,l \in \{0,1\}\}$, $w_0 := (0,0,0,0,0)$, $R := W \times W$, and η as the valuation such that $\eta\big((i,j,k,t,l)\big) \cap \{0,\dots,n-1\} = i$,

- $c_l \in \eta\big((i,j,k,t,l)\big)$ if $j = 1$,
- $c_m \in \eta\big((i,j,k,t,l)\big)$ if $k = 1$,
- $c_r \in \eta\big((i,j,k,t,l)\big)$ if $t = 1$,
- and $d \in \eta\big((i,j,k,t,l)\big)$ if $l = 1$.

Assume first that M has a b-recurring lossy computation and let $(\vec{c}_j)_{j\in\mathbb{N}}$ be the related sequence of configurations of M. Let $T \subseteq \mathrm{Traces}(\mathrm{K}_I)$ be a set of traces that encodes $(\vec{c}_j)_{j\in\mathbb{N}}$ in the way described above, and such that, for every $j \in \mathbb{N}$,

- if $\vec{c}_j = (i, v_l, v_m, v_r)$ and the instruction labelled i is of the form $i\colon C_s^+$ goto $\{j,j'\}$, then there is at most one $t \in T[j,\infty]$ such that $c_s \notin t[0]$ but $c_s \in t[1]$,
- if i is not of the form $i\colon C_s^+$ goto $\{j,j'\}$, then there is no $t \in T[j,\infty]$ such that $c_s \notin t[0]$ but $c_s \in t[1]$.

The aforementioned condition makes sure that the traces in T that encode the incrementation of counter values do not change erratically. Clearly, such a T always exists, given that $(\vec{c}_j)_{j\in\mathbb{N}}$ encodes a lossy computation. Furthermore, since T encodes $(\vec{c}_j)_{j\in\mathbb{N}}$ and the related b-recurrent lossy computation follows the instructions in I, we have that $(T,0) \models \theta_{\text{comp}} \wedge \theta_{b-\text{rec}}$. Finally, as $\emptyset \neq T \subseteq \mathrm{Traces}(\mathrm{K}_I)$, $\big(\mathrm{Traces}(\mathrm{K}_I), 0\big) \models (\theta_{\text{comp}} \wedge \theta_{b-\text{rec}}) \vee_{\mathrm{L}} \top$ follows.

Assume then that $\big(\mathrm{Traces}(\mathrm{K}_I), 0\big) \models \varphi_{I,b}$. Hence there exists some nonempty subset T of $\mathrm{Traces}(\mathrm{K}_I)$ such that $(T,0) \models \theta_{\text{comp}} \wedge \theta_{b-\text{rec}}$. It is now easy to construct a sequence $(\vec{c}_j)_{j\in\mathbb{N}}$ of configurations that encode a b-recurrent lossy computation for M; for each $j \in \mathbb{N}$, define $\vec{c}_j = (i, v_l, v_m, v_r)$ such that $\bigcup T[j] \cap \{0,\dots,n-1\} = \{i\}$, and $|\{t[j,\infty] \mid c_s \in t[j], t \in T\}| = v_s$, for each $s \in \{l,m,r\}$. ◀

▶ **Theorem 3.** *Model checking for* $\mathrm{TeamLTL}(\subseteq, \otimes, \mathsf{A})$ *is* Σ_1^1*-hard. This holds already for the fragment with a single occurrence of* A.

Proof. The proof is analogous to that of Theorem 2. The following modifications are required to shift from lossy computation to non-lossy one. Firstly, we define a formula θ_{diff} that expresses that if two traces differ (i.e., $t \neq t'$) then all of their postfixes differ as well (i.e., $t[j,\infty] \neq t'[j,\infty]$, for each $j \in \mathbb{N}$):

$$\begin{aligned}\theta_{\text{diff}} := \mathsf{A}\Big(\Box\big((c_l \varovee \neg c_l) \wedge (c_m \varovee \neg c_m) \wedge (c_r \varovee \neg c_r) \wedge (d \varovee \neg d)\big)\\ \varovee \Box\Diamond\big((\top \subseteq c_l \wedge \bot \subseteq c_l) \varovee (\top \subseteq c_m \wedge \bot \subseteq c_m)\\ \varovee (\top \subseteq c_r \wedge \bot \subseteq c_r) \varovee (\top \subseteq d \wedge \bot \subseteq d)\big)\Big).\end{aligned}$$

For $s \in \{l, m, r\}$, instead of using the formula $\mathrm{c_s}$-decrease in Theorem 2, we make use of $\mathrm{c_s}$-preserve defined as:

$$\mathrm{c_s}\text{-preserve} := \big(c_s \wedge \bigcirc c_s\big) \vee \big(\neg c_s \wedge \bigcirc \neg c_s\big).$$

Finally, we define

$$\varphi_{I,b} := (\theta_{\text{diff}} \wedge \theta'_{\text{comp}} \wedge \theta_{b-\text{rec}}) \vee_{\mathrm{L}} \top,$$

where $\theta'_{\text{comp}} := \Box \bigovoid_{i<n} (i \wedge \theta'_i)$ and θ'_i for each instruction i is defined as follows:

- For the instruction $i\colon C_l^+$ goto $\{j, j'\}$, define
  $\theta'_i := \bigcirc(j \varovee j') \wedge \big((\text{singleton} \wedge \neg c_l \wedge \bigcirc c_l) \vee_{\mathrm{L}} \mathrm{c_l}\text{-preserve}\big) \wedge \mathrm{c_r}\text{-preserve} \wedge \mathrm{c_m}\text{-preserve}\,.$
- For the instruction $i\colon C_l^-$ goto $\{j, j'\}$, define
  $\theta'_i := \bigcirc(j \varovee j') \wedge \big((\text{singleton} \wedge c_l \wedge \bigcirc \neg c_l) \vee_{\mathrm{L}} \mathrm{c_l}\text{-preserve}\big) \wedge \mathrm{c_r}\text{-preserve} \wedge \mathrm{c_m}\text{-preserve}\,.$
- For $i\colon C_s^+$ goto $\{j, j'\}$ and $i\colon C_s^-$ goto $\{j, j'\}$ with $s \in \{m, r\}$, the formulae θ'_i are defined analogously with the indices l, m, and r permuted.
- For the instruction i: if $C_s = 0$ goto j, else goto j', define
  $\theta'_i := \big((\neg c_s \wedge \bigcirc j) \varovee (\top \subseteq c_s \wedge \bigcirc j')\big) \wedge \mathrm{c_l}\text{-preserve} \wedge \mathrm{c_m}\text{-preserve} \wedge \mathrm{c_r}\text{-preserve}\,.$

◀

▶ **Corollary 4.** *The satisfiability problems for* TeamLTL$(\subseteq, \varovee)$ *and* TeamLTL$(\subseteq, \varovee, \mathsf{A})$ *are* Σ^0_1*-hard and* Σ^1_1*-hard, resp.*

Proof. Given a finite Kripke structure $\mathrm{K} = (W, R, \eta, w_0)$ over a finite set of propositions AP, we introduce a fresh propositional variable p_w, for each $w \in W$, and let K' be the unique extension of K in which p_w holds only in w. We then construct a formula $\theta_{\mathrm{K}'}$ whose intention is to characterise the team $\big(\text{Traces}(\mathrm{K}'), 0\big)$. Define

$$\begin{aligned}\theta_{\mathrm{K}'} := p_{w_0} \wedge \Box \bigvee_{w \in W} \Bigg(p_w \wedge \Big(\bigwedge_{w \neq v \in W} \neg p_v\Big) \wedge \Big(\bigwedge_{(w,v) \in R} \top \subseteq \bigcirc p_v\Big) \wedge \Big(\bigcirc \bigvee_{(w,v) \in R} p_v\Big)\\ \wedge \bigwedge_{p \in \eta(w)} p \wedge \bigwedge_{p \in \mathrm{AP} \setminus \eta(w)} \neg p\Bigg).\end{aligned}$$

It is not hard to verify that the only sets of traces over $\mathrm{AP} \cup \{p_w \mid w \in W\}$ for which $(T, 0) \models \theta_{\mathrm{K}'}$ holds are the empty set and $\text{Traces}(\mathrm{K}')$. Now, given a finite Kripke structure K and a formula φ of TeamLTL$(\subseteq, \varovee)$ (TeamLTL$(\subseteq, \varovee, \mathsf{A})$, resp.) it holds that $(\text{Traces}(\mathrm{K}), 0) \models \varphi$ iff $\theta_{\mathrm{K}'} \wedge \varphi$ is satisfiable. ◀

C Missing Proofs of Section 4

▶ **Lemma 5.** *For every set T of traces over a countable* AP*, there exists a countable set $\mathcal{S}_T \subseteq 2^T$ such that, for every* $\mathrm{TeamLTL}(\ovee, \sim\!\bot, \overset{1}{\mathsf{A}})$*-formula φ and $i \in \mathbb{N}$,*

$$(T, i) \models \varphi \quad \textit{iff} \quad (T, i) \models^* \varphi, \tag{2}$$

where the satisfaction relation $\models^*$ is defined the same way as $\models$ except that in the semantic clause for $\vee$ we require additionally that the two subteams $T_1, T_2 \in \mathcal{S}_T$.

Proof. We first define inductively and nondeterministically a function $\mathrm{Sub}\colon (2^T \times \mathbb{N}) \times \mathrm{TeamLTL}(\ovee, \mathcal{A}) \to 2^T$ as follows:

- If φ is an atomic formula or a generalised atom, define $\mathrm{Sub}\big((S, j), \varphi\big) := \{S\}$
- $\mathrm{Sub}\big((S, j), \bigcirc\varphi\big) := \mathrm{Sub}\big((S, i+1), \varphi\big)$.
- $\mathrm{Sub}\big((S, j), \varphi \wedge \psi\big) := \mathrm{Sub}\big((S, j), \varphi\big) \cup \mathrm{Sub}\big((S, j), \psi\big)$.
- $\mathrm{Sub}\big((S, j), \varphi \ovee \psi\big) := \mathrm{Sub}\big((S, j), \varphi\big) \cup \mathrm{Sub}\big((S, j), \psi\big)$.
- $\mathrm{Sub}\big((S, j), \varphi \vee \psi\big) := \mathrm{Sub}\big((S_1, j), \varphi\big) \cup \mathrm{Sub}\big((S_2, j), \psi\big) \cup \{S\}$, where subsets S_1 and S_2 of S are guessed nondeterministically such that $S_1 \cup S_2 = S$.
- $\mathrm{Sub}\big((S, j), \psi\,\mathcal{U}\,\varphi\big) := \mathrm{Sub}\big((S, j), \psi\,\mathcal{W}\,\varphi\big) := \bigcup_{i \geq j} \big(\mathrm{Sub}\big((S, i), \psi\big) \cup \mathrm{Sub}\big((S, i), \varphi\big)\big)$.

Clearly $\mathrm{Sub}\big((S, j), \psi\big)$ is a countable set. Now, define

$$\mathcal{S}_T := \bigcup_{\substack{j \in \mathbb{N} \\ \psi \in \mathrm{TeamLTL}(\ovee, \mathcal{A})}} \mathrm{Sub}\big((T, j), \psi\big).$$

The set $\mathcal{S}_T$ is a countable union of countable sets, and thus itself countable. For each team (S, j) and formula ψ, assuming $\mathcal{S}_T \supseteq \mathrm{Sub}\big((S, j), \psi\big)$, it is not hard to show by induction that (2) holds. ◀

▶ **Theorem 6.** *For every $\varphi \in \mathrm{TeamLTL}(\ovee, \sim\!\bot, \overset{1}{\mathsf{A}})$ there exists an equivalent* $\mathrm{HyperQPTL}^+$*-formula φ^* in the $\exists_p \overset{u}{Q}{}^*_p \exists^*_\pi \forall_\pi$ fragment. If $\varphi \in \mathrm{TeamLTL}(\ovee, \overset{1}{\mathsf{A}})$, φ^* can be defined in the $\exists_p \overset{u}{Q}{}^*_p \forall_\pi$ fragment. The size of φ^* is linear w.r.t. the size of φ.*

Proof. Let $q^{\mathcal{S}_T}$, q, and r be distinct propositional variables. We define a compositional translation $\mathrm{TR}_{(q,r)}$ such that for every team (T, i) and $\mathrm{TeamLTL}(\ovee, \sim\!\bot, \overset{1}{\mathsf{A}})$-formula φ,

$$(T, i) \models \varphi \text{ iff } \Pi, i \models_T \exists q^{\mathcal{S}_T} \overset{u}{\exists} q \overset{u}{\exists} r \big(\mathrm{TR}_{(q,r)}(\varphi) \wedge \varphi_{\mathrm{aux}}\big), \text{ where } \varphi_{\mathrm{aux}} := \forall \pi.\, q^{\mathcal{S}_T}_\pi \wedge q_\pi \wedge r_\pi. \tag{3}$$

We first fix some conventions. All quantified variables in the translation below are assumed to be fresh and distinct. We also assume that the uniformly quantified propositional variables $q, r, \ldots$ are true in exactly one level, that is, they satisfy the formula $\forall \pi. \Diamond q_\pi \wedge \Box(q_\pi \to \bigcirc\Box\neg q_\pi)$, which we will omit in the presentation of the translation for simplicity.

The idea behind the translation is the following. Let (T', i) denote the team obtained from (T, i) by evaluating the quantifier $\exists q^{\mathcal{S}_T}$.

- The variable $q^{\mathcal{S}_T}$ is used to encode the countable set $\mathcal{S}_T$ of sets of traces given by Lemma 5. To be precise, for each $i \in \mathbb{N}$, $q^{\mathcal{S}_T}$ encodes the set $(\{t \in T' \mid q^{\mathcal{S}_T} \in t[i]\} \upharpoonright \mathrm{AP}) \in \mathcal{S}_T$.
- The uniformly quantified variable q in $\mathrm{TR}_{(q,r)}$ is used to encode an element of $\mathcal{S}_T$ using $q^{\mathcal{S}_T}$: If $q \in t[i]$, then q encodes the set $\{t \in T' \mid q^{\mathcal{S}_T} \in t[i]\} \upharpoonright \mathrm{AP}$.
- The uniformly quantified variable r in $\mathrm{TR}_{(q,r)}$ is used to encode the time step i of a team (T, i): If $r \in t[i]$, then r encodes the time step i.

After fixing a suitable interpretation for $q^{\mathcal{S}_T}$, teams (S,i) can be encoded with pairs of uniformly quantified variables (q,r), whenever $S \in \mathcal{S}_T$. The formula φ_{aux} expresses that the pair (q,r) encodes the team (T,i) in question.

The translation $\mathrm{TR}_{(q,r)}$ is defined inductively as follows:

$$
\begin{aligned}
\mathrm{TR}_{(q,r)}(\ell) &:= \forall\pi.\, \big(\Diamond(q_\pi \wedge q_\pi^{\mathcal{S}_T}) \to \Diamond(r_\pi \wedge \ell_\pi)\big), \quad \text{where } \ell \in \{p, \neg p\}\\
\mathrm{TR}_{(q,r)}(\varphi \otimes \psi) &:= \mathrm{TR}_{(q,r)}(\varphi) \vee \mathrm{TR}_{(q,r)}(\psi), \quad \mathrm{TR}_{(q,r)}(\varphi \wedge \psi) := \mathrm{TR}_{(q,r)}(\varphi) \wedge \mathrm{TR}_{(q,r)}(\psi)\\
\mathrm{TR}_{(q,r)}(\bigcirc\varphi) &:= \overset{u}{\exists} r'.\, \big(\Box(r_\pi \leftrightarrow \bigcirc r'_\pi) \wedge \mathrm{TR}_{(q,r')}(\varphi)\big), \quad \mathrm{TR}_{(q,r)}(\sim\!\bot) := \exists\pi.\, \Diamond(q_\pi \wedge q_\pi^{\mathcal{S}_T})\\
\mathrm{TR}_{(q,r)}(\overset{1}{\mathsf{A}}\varphi) &:= \forall\pi. \big(\Diamond(q_\pi \wedge q_\pi^{\mathcal{S}_T}) \to \Diamond(r_\pi \wedge \varphi^*_\pi)\big), \quad \text{(where the } \varphi^* \text{ is an LTL-formula}\\
&\qquad \text{obtained from } \varphi \text{ by removing all } \overset{1}{\mathsf{A}} \text{ and using a careful bottom-up linear}\\
&\qquad \text{procedure to eliminate all } \otimes \text{ and } \sim\!\bot\text{. Detailed exposition below the proof.)}\\
\mathrm{TR}_{(q,r)}(\varphi \vee \psi) &:= \overset{u}{\exists} q_1 q_2.\, \big(\varphi_\cup(q, q_1, q_2) \wedge \mathrm{TR}_{(q_1,r)}(\varphi) \wedge \mathrm{TR}_{(q_2,r)}(\psi)\big)\\
\mathrm{TR}_{(q,r)}(\varphi\,\mathcal{U}\,\psi) &:= \overset{u}{\exists} r'.\, \Big(r \preceq r' \wedge \mathrm{TR}_{(q,r')}(\psi) \wedge \overset{u}{\forall} r''.\, \big((r \preceq r'' \wedge r'' \prec r') \to \mathrm{TR}_{(q,r'')}(\varphi)\big)\Big)\\
\mathrm{TR}_{(q,r)}(\varphi\,\mathcal{W}\,\psi) &:= \forall r'.\, \Big(r \preceq r' \to \Big(\mathrm{TR}_{(q,r')}(\varphi) \vee \exists r''.\, \big(r'' \preceq r' \wedge \mathrm{TR}_{q,r''}(\psi)\big)\Big)\Big)
\end{aligned}
$$

where $\varphi_\cup(q,q',q'') := \Diamond(q_\pi \wedge q_\pi^{\mathcal{S}_T}) \leftrightarrow \Diamond\big((q'_\pi \vee q''_\pi) \wedge q_\pi^{\mathcal{S}_T}\big)$, $r \prec r' := \Box(r_\pi \leftrightarrow \bigcirc\Diamond r'_\pi)$, and $r \preceq r' := \Box(r_\pi \to \Diamond r'_\pi)$.

The formula $\mathrm{TR}_{(q,r)}(\varphi) \wedge \varphi_{\mathrm{aux}}$ can be transformed to an equivalent prenex formula in the fragment $\overset{u}{Q}{}^*_p \exists^*_\pi \forall_\pi$, and if $\varphi \in \mathrm{TeamLTL}(\otimes, \overset{1}{\mathsf{A}})$, the translation is in the $\overset{u}{Q}{}^*_p \forall_\pi$ fragment. The translations are clearly linear. The equivalence (3) is proved by a routine inductive argument; we omit the detailed proof. ◀

Next we describe the procedure needed for the case $\overset{1}{\mathsf{A}}$ of the translation of Theorem 6. It suffices to establish a linear translation $^*\colon \mathrm{TeamLTL}(\otimes, \sim\!\bot) \to \mathrm{TeamLTL}$ such that $\overset{1}{\mathsf{A}}\varphi \equiv \overset{1}{\mathsf{A}}\varphi^*$, for every $\varphi \in \mathrm{TeamLTL}(\otimes, \sim\!\bot)$. The translation is written bottom-up, and the symbols $\otimes$ and $\sim\!\bot$ do not occur in the formulae ψ and θ below. (We wish to remind the reader that all $\mathrm{TeamLTL}(\otimes)$ formulae are satisfied by the empty team.) For formulae φ in which $\sim\!\bot$ does not occur, the translation only replaces the symbols $\otimes$ by $\vee$. In other cases the translation depends on the position of $\sim\!\bot$ in the subformulae. Below ψ and θ are allowed to be empty (i.e, $\sim\!\bot \wedge \psi$ covers the case $\sim\!\bot$).

$$
\begin{aligned}
(\sim\!\bot \wedge \psi) \vee (\sim\!\bot \wedge \theta) &\mapsto \sim\!\bot \wedge (\psi \wedge \theta)\\
(\sim\!\bot \wedge \psi) \vee \theta &\mapsto \sim\!\bot \wedge \psi\\
(\sim\!\bot \wedge \psi) \otimes (\sim\!\bot \wedge \theta) &\mapsto \sim\!\bot \wedge (\psi \vee \theta)\\
(\sim\!\bot \wedge \psi) \otimes \theta &\mapsto \psi \vee \theta\\
(\sim\!\bot \wedge \psi) \wedge (\sim\!\bot \wedge \theta) &\mapsto \sim\!\bot \wedge (\psi \wedge \theta)\\
(\sim\!\bot \wedge \psi) \wedge \theta &\mapsto \sim\!\bot \wedge (\psi \wedge \theta)\\
\bigcirc(\sim\!\bot \wedge \psi) &\mapsto \sim\!\bot \wedge \bigcirc\psi\\
(\sim\!\bot \wedge \psi)\,\mathcal{U}(\sim\!\bot \wedge \theta) &\mapsto \sim\!\bot \wedge (\psi\,\mathcal{U}\,\theta)\\
\psi\,\mathcal{U}(\sim\!\bot \wedge \theta) &\mapsto \sim\!\bot \wedge (\psi\,\mathcal{U}\,\theta)\\
(\sim\!\bot \wedge \psi)\,\mathcal{U}\,\theta &\mapsto \psi\,\mathcal{U}\,\theta\\
(\sim\!\bot \wedge \psi)\,\mathcal{W}(\sim\!\bot \wedge \theta) &\mapsto \sim\!\bot \wedge (\psi\,\mathcal{W}\,\theta)\\
\psi\,\mathcal{W}(\sim\!\bot \wedge \theta) &\mapsto \psi\,\mathcal{W}\,\theta\\
(\sim\!\bot \wedge \psi)\,\mathcal{W}\,\theta &\mapsto \psi\,\mathcal{W}\,\theta
\end{aligned}
$$

It is straightforward to check that, for any TeamLTL-formulae ψ and θ, the above translation preserves the truth value of formulae over teams of cardinality at most 1. Hence, when using the translation to φ, we obtain a formula φ^+ of the form $\sim\bot \wedge \psi$ or ψ, where $\psi \in$ TeamLTL, such that $\overset{1}{\mathsf{A}}\varphi$ and $\overset{1}{\mathsf{A}}\varphi^+$ are equivalent. Finally note that, by the semantics of $\overset{1}{\mathsf{A}}$, the formulae $\overset{1}{\mathsf{A}}(\sim\bot \wedge \psi)$ and $\overset{1}{\mathsf{A}}\psi$ are equivalent. By defining $\varphi^* := \psi$ we obtain the linear size formula needed in Theorem 6.

D Missing Proofs of Section 5

▶ **Lemma 8.** *Let φ be a formula of* TeamLTL($\sim$) *and $\Phi = \{\pi_1, \dots, \pi_k\}$ a finite set of trace variables. For any team (T, i) with $|T| \leq k$, any set $S \supseteq T$ of traces, and any assignment Π with $\Pi[\Phi] = T$, we have that $(T, i) \models \varphi$ iff $\Pi, i \models_S \varphi^\Phi$. Furthermore, if φ is downward closed and $T \neq \emptyset$, then $(T, i) \models \varphi$ iff $\emptyset, i \models_T \forall\pi_1 \dots \forall\pi_k. \varphi^\Phi$.*

Proof. The second claim of the lemma follows easily from the first claim. We prove the first claim by induction on φ. If $\varphi = p$, then

$$\begin{aligned}
(T,i) \models p &\Leftrightarrow \forall t \in T : p \in t[i] \\
&\Leftrightarrow \forall \pi \in \Phi : p \in \Pi(\pi)[i] && (\because T = \Pi[\Phi]) \\
&\Leftrightarrow \Pi, i \models_S \bigwedge_{\pi \in \Phi} p_\pi && (\because S \supseteq T) \\
&\Leftrightarrow \Pi, i \models_S p^\Phi
\end{aligned}$$

Note that if $T = \emptyset$, then $\Phi = \emptyset$ and by definition $p^\emptyset = \top$. In this special case the above proof still goes through, and in particular $(\emptyset, i) \models p$ and $\Pi, i \models_S \top$.

The case $\varphi = \neg p$ is proved analogously. If $\varphi = \sim\psi$, then

$$\begin{aligned}
(T,i) \models \sim\psi &\Leftrightarrow (T,i) \not\models \psi \\
&\Leftrightarrow (\Pi,i) \not\models_S \psi^\Phi && \text{(by induction hypothesis)} \\
&\Leftrightarrow (\Pi,i) \models_S \neg\psi^\Phi.
\end{aligned}$$

If $\varphi = \psi \vee \chi$, then

$$\begin{aligned}
&(T,i) \models \psi \vee \chi \\
\Leftrightarrow\ & (\Pi[\Phi_0], i) \models \psi \text{ and } (\Pi[\Phi_1], i) \models \chi \text{ for some } \Phi_0, \Phi_1 \\
& \text{ with } \Phi_0 \cup \Phi_1 = \Phi && (\text{since } T = \Pi[\Phi]) \\
\Leftrightarrow\ & \Pi, i \models_S \psi^{\Phi_0} \text{ and } \Pi, i \models_S \chi^{\Phi_1} \text{for some } \Phi_0, \Phi_1 \text{ with} \\
& \Phi_0 \cup \Phi_1 = \Phi \text{ (by IH, } \because S \supseteq T = \Pi[\Phi] \supseteq \Pi[\Phi_0], \Pi[\Phi_1]) \\
\Leftrightarrow\ & \Pi, i \models_S \bigvee_{\Phi_0 \cup \Phi_1 = \Phi} (\psi^{\Phi_0} \wedge \chi^{\Phi_1}) \\
\Leftrightarrow\ & \Pi, i \models_S (\psi \vee \chi)^\Phi.
\end{aligned}$$

If $\varphi = \chi \mathcal{U} \psi$, then

$$\begin{aligned}
&(T,i) \models \chi \mathcal{U} \psi \\
\Leftrightarrow\ & \exists n \geq i : (T,n) \models \psi, \text{ and } \forall m : i \leq m < n \Rightarrow (T,m) \models \chi \\
\Leftrightarrow\ & \exists n \geq i : \Pi, n \models_S \psi^\Phi, \text{ and } \forall m : i \leq m < n \Rightarrow \Pi, m \models_S \chi^\Phi \\
\Leftrightarrow\ & \Pi, i \models_S \chi^\Phi \mathcal{U} \psi^\Phi
\end{aligned}$$

$\Leftrightarrow \Pi, i \models_S (\chi \mathcal{U} \psi)^{\Phi}$.

The other inductive cases are proved by using a routine argument. ◀

▶ **Theorem 9.** *Every k-coherent property that is definable in* TeamLTL($\sim$) *is also definable in* $\forall^k$HyperLTL.

Proof. Let φ be a k-coherent TeamLTL($\sim$)-formula. If $\emptyset, i \not\models \varphi$ for some $i \in \mathbb{N}$, then $(T, j) \not\models \varphi$ for every team (T, j). We then translate φ to $\forall\pi. \bot$. Assume now $\emptyset, i \models \varphi$. For any nonempty team (T, i), we have that

$$
\begin{aligned}
& (T, i) \models \varphi \\
\Leftrightarrow\ & (S, i) \models \varphi \text{ for every } S \subseteq T \text{ with } |S| \leq k && \text{(since } \varphi \text{ is } k\text{-coherent)} \\
\Leftrightarrow\ & \Pi, i \models_T \varphi^{\{\pi_1,\dots,\pi_k\}} \text{ for every } \Pi \text{ s.t } \Pi[\{\pi_1,\dots,\pi_k\}] \subseteq T && \text{(by Lemma 8 \& } \emptyset, i \models \varphi) \\
\Leftrightarrow\ & \emptyset, i \models_T \forall\pi_1 \dots \forall\pi_k. \varphi^{\{\pi_1,\dots,\pi_k\}}.
\end{aligned}
$$

◀

▶ **Theorem 12.** *Checking whether a* TeamLTL($\subseteq, \otimes$)*-formula is* 1*-coherent is undecidable.*

Proof. For a given TeamLTL($\subseteq, \otimes$)-formula φ, let φ^* denote the TeamLTL-formula obtained from φ by first replacing each disjunctions $\otimes$ by $\vee$, and then replacing each inclusion atom of the form $\varphi_1, \dots, \varphi_n \subseteq \psi_1, \dots, \psi_n$ by the conjunction

$$\bigwedge_{i \leq n} \varphi_i \leftrightarrow \psi_i.$$

It is easy to verify, by induction, that over singleton teams φ and φ^* are equivalent. Moreover, since TeamLTL satisfies the singleton equivalence property, we obtain the following equivalences, for every trace t:

$$(\{t\}, i) \models \varphi \Leftrightarrow (\{t\}, i) \models \varphi^* \Leftrightarrow (t, i) \models \varphi^*,$$

where the right-most satisfaction relation is the standard one for LTL.

Next, we show that for any formula $\varphi \in$ TeamLTL($\subseteq, \otimes$)

$$\varphi \text{ is not satisfiable} \quad \Leftrightarrow \quad \varphi \text{ is 1-coherent and } \varphi^* \text{ is not satisfiable in the sense of LTL.}$$

Assume that φ is not satisfiable. Since no team satisfies φ, it is trivially 1-coherent. In particular, for any trace t, we have $(\{t\}, i) \not\models \varphi$, which then implies that $(t, i) \not\models \varphi^*$. Thus φ^* is not satisfiable in the sense of LTL. Conversely, suppose φ is 1-coherent and φ^* is not satisfiable. Then for any trace t, we have $(t, i) \not\models \varphi^*$, which implies that $(\{t\}, i) \not\models \varphi$. It then follows, by 1-coherence, that $(T, i) \not\models \varphi$ for every team T. Thus φ is not satisfiable.

Now, since checking LTL-satisfiability can be done in PSPACE [45] and non-satisfiability for TeamLTL($\subseteq, \otimes$) is Π^0_1-hard by Corollary 4, we conclude that checking 1-coherence is undecidable.

◀

▶ **Theorem 14.** *For every formula φ from the left-flat fragment of* TeamLTL($\otimes, \overset{1}{\mathsf{A}}$)*, we can compute an equivalent* $\overset{u}{\exists}{}^*_p\forall_\pi$ HyperQPTL *formula of size linear in the size of φ.*

The theorem follows from the following lemma.

▶ **Lemma 20.** *Let φ be a* TeamLTL$(\varovee, \overset{1}{\mathsf{A}})$*-formula, and $r_1, \dots, r_n$ free variables occurring in $[\varphi, r]$ but not in φ. Let $i \in \mathbb{N}$, and $s \in (2^{\{r\}})^\omega$ a sequence that has r set exactly at position i. For every team (T, i),*

$$(T,i) \models \varphi \Leftrightarrow \emptyset, 0 \models_{T[r\mapsto s]} \exists r_1 \dots r_n \forall \pi.\, [\varphi, r].$$

Proof. We proceed by induction on φ.
Case for p:

$$\begin{aligned}(T,i) \models p &\Leftrightarrow \forall t \in T : p \in t[i] \\ &\Leftrightarrow \emptyset, 0 \models_{T[r\mapsto s]} \forall \pi. \Box(r_\pi \to p_\pi)\end{aligned}$$

Case for $\neg p$: Similar to the above.
Case for $\overset{1}{\mathsf{A}}\varphi$:

$$\begin{aligned}(T,i) \models \overset{1}{\mathsf{A}}\varphi &\Leftrightarrow \forall t \in T : (t,i) \models \varphi \\ &\Leftrightarrow \emptyset, 0 \models_{T} \forall \pi. \Box(r_\pi \to \hat{\varphi}) && \text{(by Theorem 9, as } \overset{1}{\mathsf{A}}\varphi \text{ is 1-coherent)}\end{aligned}$$

Case for $\bigcirc\varphi$:

$$\begin{aligned}&(T,i) \models \bigcirc\varphi \\ \Leftrightarrow\ &(T,i+1) \models \varphi \\ \overset{\text{IH}}{\Leftrightarrow}\ &\emptyset, 0 \models_{T[r^\varphi \mapsto s^\varphi]} \exists r_1 \dots r_n. \forall \pi. [\varphi, r^\varphi] \\ \Leftrightarrow\ &\emptyset, 0 \models_{T[r\mapsto s]} \exists r^\varphi. \exists r_1 \dots r_n. \forall \pi. \Box(r_\pi \leftrightarrow \bigcirc r^\varphi_\pi) \wedge [\varphi, r^\varphi]\end{aligned}$$

Case for $\varphi \wedge \psi$: Easy, by induction hypothesis.
Case for $\varphi \vee \psi$:

$$\begin{aligned}&(T,i) \models \varphi \vee \psi \\ \Leftrightarrow\ &\exists T_1, T_2 \text{ s.t. } T = T_1 \cup T_2 \text{ and} \\ &\qquad (T_1, i) \models \varphi \text{ and } (T_2, i) \models \psi \\ \overset{\text{IH}}{\Leftrightarrow}\ &\emptyset, 0 \models_{T_1[r\mapsto s]} \exists r^1_1 \dots r^1_n. \forall \pi. [\varphi, r] \text{ and } \emptyset, 0 \models_{T_2[r\mapsto s]} \exists r^2_1 \dots r^2_m. \forall \pi. [\psi, r] \\ \Leftrightarrow\ &\emptyset, 0 \models_{T[r\mapsto s]} \exists r^1_1 \dots r^1_n r^2_1 \dots r^2_m. \forall \pi. [\varphi, r] \vee [\psi, r] \\ &\hfill \text{(since } T_1[r\mapsto s] \cup T_2[r\mapsto s] = T[r\mapsto s])\end{aligned}$$

Case for $\varphi \varovee \psi$:

$$\begin{aligned}&(T,i) \models \varphi \varovee \psi \\ \Leftrightarrow\ &(T,i) \models \varphi \text{ or } (T,i) \models \psi \\ \overset{\text{IH}}{\Leftrightarrow}\ &\emptyset, 0 \models_{T[r\mapsto s]} \exists r^1_1 \dots r^1_n. \forall \pi. [\varphi, r] \text{ or } \emptyset, 0 \models_{T[r\mapsto s]} \exists r^2_1 \dots r^2_m. \forall \pi. [\psi, r] \\ \Leftrightarrow\ &\emptyset, 0 \models_{T[r\mapsto s]} \exists d^{\varphi\varovee\psi}. \exists r^1_1 \dots r^1_n r^2_1 \dots r^2_m. \forall \pi. (d^{\varphi\varovee\psi}_\pi \to [\varphi, r]) \wedge (\neg d^{\varphi\varovee\psi}_\pi \to [\psi, r])\end{aligned}$$

Case for $\varphi \mathcal{U} \psi$:

$$\begin{aligned}&(T,i) \models \varphi \mathcal{U} \psi \\ &\Leftrightarrow \exists i' \geq i. (T, i') \models \psi \text{ and } \forall i \leq i'' < i'. (T, i'') \models \varphi \\ &\Leftrightarrow \exists i' \geq i. (T, i') \models \psi \text{ and } \forall i \leq i'' < i'. \emptyset, i'' \models_T \forall \pi. \hat{\varphi} \\ &\hfill \text{(by Theorem 9, since } \varphi \text{ is 1-coherent)}\end{aligned}$$

$$\overset{\text{IH}}{\Leftrightarrow} \exists i' \geq i.\, \emptyset, 0 \models_{T[r^\psi \mapsto s^\psi]} \exists r_1 \ldots r_n.\, \forall \pi.\, [\psi, r^\psi] \text{ and } \forall i \leq i'' < i'.\, \emptyset, i'' \models_T \forall \pi.\, \hat{\varphi}$$

(where r^ψ is set in s^ψ exactly at position i')

$$\Leftrightarrow \exists i' \geq i.\, \emptyset, 0 \models_{T[r \mapsto s, r^\psi \mapsto s^\psi, r^\varphi \mapsto s^\varphi]} \exists r_1 \ldots r_n.\, \forall \pi.\, \Box(r^\varphi_\pi \to \hat{\varphi}) \wedge [\psi, r^\psi]$$

(where r^ψ is set in s^ψ exactly at position i' and r^φ is set in s^φ exactly at all positions between i and i')

$$\Leftrightarrow \emptyset, 0 \models_{T[r \mapsto s]} \models \exists r^\psi.\, \exists r^\varphi.\, \exists r_1 \ldots r_n.\, \forall \pi.\, \Box(r_\pi \to r^\varphi_\pi \,\mathcal{U} (r^\psi_\pi \wedge \bigcirc \Box \neg r^\psi_\pi)) \wedge \Box(r^\varphi_\pi \to \hat{\varphi}) \wedge [\psi, r^\psi]$$

Case for $\varphi \,\mathcal{W}\, \psi$. Similar to $\varphi \,\mathcal{U}\, \psi$. ◀