


The Complexity of Learning LTL, CTL and ATL Formulas

Benjamin Bordais 

TU Dortmund University, Center for Trustworthy Data Science and Security,
University Alliance Ruhr, Dortmund, Germany

Daniel Neider 

TU Dortmund University, Center for Trustworthy Data Science and Security,
University Alliance Ruhr, Dortmund, Germany

Rajarshi Roy 

Department of Computer Science, University of Oxford, UK

Abstract

We consider the problem of learning temporal logic formulas from examples of system behavior. Learning temporal properties has crystallized as an effective means to explain complex temporal behaviors. Several efficient algorithms have been designed for learning temporal formulas. However, the theoretical understanding of the complexity of the learning decision problems remains largely unexplored. To address this, we study the complexity of the passive learning problems of three prominent temporal logics, Linear Temporal Logic (LTL), Computation Tree Logic (CTL) and Alternating-time Temporal Logic (ATL) and several of their fragments. We show that learning formulas with unbounded occurrences of binary operators is NP-complete for all of these logics. On the other hand, when investigating the complexity of learning formulas with bounded occurrences of binary operators, we exhibit discrepancies between the complexity of learning LTL, CTL and ATL formulas (with a varying number of agents).

2012 ACM Subject Classification Theory of computation

Keywords and phrases Temporal logic, passive learning, complexity

Digital Object Identifier 10.4230/LIPIcs.STACS.2025.19

Related Version *Full Version:* <https://arxiv.org/abs/2408.04486> [8]

Funding Rajarshi Roy acknowledges partial funding by the ERC under the European Union's Horizon 2020 research and innovation programme (grant agreement No.834115, FUN2MODEL).

1 Introduction

Temporal logics are the de-facto standard for expressing temporal properties for software and cyber-physical systems. Originally introduced in the context of program verification [33, 15], temporal logics are now well-established in numerous areas, including reinforcement learning [40, 25, 10], motion planning [17, 12], process mining [13], and countless others. The popularity of temporal logics can be attributed to their unique blend of mathematical rigor and resemblance to natural language.

Until recently, formulating properties in temporal logics has been a manual task, requiring human intuition and expertise [6, 39]. To circumvent this step, in the past ten years, there have been numerous works to automatically learn (i.e., generate) properties in temporal logic. Among them, a substantial number of works [29, 11, 35, 26, 41] target Linear Temporal Logic (LTL) [33]. There is now a growing interest in learning formulas [16, 34, 9] in Computation Tree Logic (CTL) [15] and Alternating-time Temporal Logic (ATL) [1] due to their ability to express branching-time properties, including for multi-agent systems.



© Benjamin Bordais, Daniel Neider, and Rajarshi Roy;
licensed under Creative Commons License CC-BY 4.0

42nd International Symposium on Theoretical Aspects of Computer Science (STACS 2025).

Editors: Olaf Beyersdorff, Michał Pilipczuk, Elaine Pimentel, and Nguyễn Kim Thăng;

Article No. 19; pp. 19:1–19:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Table 1** The complexity results for learning LTL, CTL and ATL formulas. The notation ATL^k refers to ATL-formulas with k agents. $U^t \subseteq \{\neg, \mathbf{X}, \mathbf{F}, \mathbf{G}\}$ refers to the set of unary temporal operators.

	Unbounded use of binary operators	Bounded use of binary operators		
		$\mathbf{X} \in U^t$	$\mathbf{X} \notin U^t$	
			$\{\mathbf{F}, \mathbf{G}\} \subseteq U^t$	$U^t = \{\mathbf{F}\}, \{\mathbf{G}\}$
LTL	NP-c	L		
CTL		NP-c	NL-c	
ATL^2		NP-c		P-c
ATL^k		NP-c		

While existing approaches for learning temporal properties demonstrate impressive empirical performance, detailed comparisons of computational complexity across different temporal logics remain underexplored. Most related works focus on LTL, either in the verification domain [18, 27] or the database domain [19, 24]. These studies primarily report complexity results, often highlighting NP-completeness for learning LTL-formulas and their fragments. In contrast, the computational complexity of learning CTL- and ATL-formulas has not yet been thoroughly examined.

In this work, we extend the study of learning temporal properties to include CTL- and ATL-formulas. Additionally, we broaden existing results for LTL to cover a more comprehensive set of operators, specifically addressing all binary operators (temporal or not).

To elaborate on our contributions, let us precisely describe the problem that we consider, the *passive learning* problem for temporal logic [29, 11]. Its decision version asks the following question: given two sets \mathcal{P} , \mathcal{N} of positive and negative examples of a system’s behavior and a size bound B , does there exist a “separating” formula of size at most B , which is satisfied by the positive examples and violated by the negative ones.

Our instantiation of the above problem depends on the considered logic, following related literature [29, 34, 9]: LTL-formulas express linear-time properties, CTL-formulas express branching-time properties, and ATL-formulas express properties on multi-agent systems. Accordingly, the input examples for learning LTL, CTL and ATL are linear structures (or equivalently infinite words), Kripke structures and concurrent game structures, respectively. We refer to Section 2 for formal definitions and other prerequisites.

We summarize our contributions in Table 1. Our first result, illustrated in the left column, shows that allowing formulas with unrestricted use of at least one binary operator makes the corresponding learning decision problem NP-complete for all considered logics. Some of these NP-hardness results are (closely) inspired by [27], involving reductions from the hitting set problem – one of Karp’s 21 NP-complete problem; some others require novel proof techniques, e.g. one involves a reduction from an NP-complete modulo-2 calculus problem. We describe the outline of the proofs in Section 3.

All of the above NP-hardness proofs rely on separating formulas with linearly many (in the size of the input) occurrences of binary operators. Thus, in the search of expressive temporal logic fragments with lower complexities, we focus on formulas with a bounded occurrences of binary logical operators such as \wedge (and), \vee (or), etc. and no binary temporal operators such as \mathbf{U} (until), \mathbf{R} (release), etc. This choice of formulas is motivated by the fact that such formulas can still express interesting properties (e.g., GR(1) [32] formulas, mode-target formulas [4], etc.) and are used in several practical applications (see Section 4.1 for details). We explore several fragments with different unary temporal operators, \mathbf{X} (next),

F (eventually) and **G** (globally), and present the results in the rightmost column of Table 1. We notice that, in this case, the complexity of the learning problems varies considerably between different logics and unary operators. Importantly, we exhibit fragments where the learning problem is below NP. We prove the three NP-hardness results using a reduction from the hitting set problem; we give key insights on all of these results in Section 4.

All details can be found in the extended version [8].

Related Works. The most closely related works are [18] and [27], which operate within a similar framework to ours. Both works consider learning problems in several fragments of LTL, especially involving boolean operators such as \vee and \wedge , and temporal operators such as **X**, **F** and **G** and prove their NP-completeness. We extend part of their work by categorizing fragments based on the arity of the operators and studying which type of operators contribute to the hardness. Moreover, there are several differences in the parameters considered for the learning problem. The most important one is the following: the above works consider the size upper bound B to be in binary, while we assume B given in unary. Although, in complexity problems, integers are most often assumed to be written in binary, we believe that considering size bound in unary is justified since one may want to not only decide the existence of a formula but also effectively output one, which will require to explicitly write it. The other differences with the setting of the above works are mostly due to the fact that we do not only consider LTL learning, but CTL and ATL learning as well. A thorough discussion of these differences can be found in the extended version of this paper [8].

In the past, complexity analysis of passive learning has been studied for formalisms other than temporal logics. For instance, [21] and [2] proved NP-completeness of the passive learning problems of deterministic finite automata (DFAs) and regular expressions (REs).

When it comes to temporal logics, most related works focus on developing efficient algorithms for learning temporal logic formulas. Among these, the emphasis has predominantly been on learning LTL (or its significant fragments), which has been discussed in detail in a recent survey summarizing various learning techniques [30]. Broadly, the techniques can be categorized into three main types: constraint solving [29, 11, 37, 20, 22], enumerative search [35, 41], and neuro-symbolic techniques [26, 42].

For learning CTL, some approaches rely on handcrafted templates [14, 43] for simple enumerative search, while others employ constraint-solving methods to learn formulas with arbitrary structures [34]. The constraint-solving methods are extended to learn ATL-formulas as well [9]. There are also works on learning other logics such as Signal Temporal Logic [7, 28], Metric Temporal Logic [36], Past LTL [3], Property Specification Language [38], etc.

2 Preliminaries and Definitions

We let \mathbb{N} denote the set of all integers and \mathbb{N}_1 denote the set of all positive integers. For all $i \leq j \in \mathbb{N}$, we let $[i, \dots, j] \subseteq \mathbb{N}$ denote the set of integers $\{i, i+1, \dots, j\}$.

Given any non-empty set Q , we let Q^* , Q^+ and Q^ω denote the sets of finite, non-empty finite and infinite sequences of elements in Q , respectively. For all $\rho \in Q^+$, we denote by $|\rho| \in \mathbb{N}$ the number of elements of ρ . For all $\bullet \in \{+, \omega\}$, $\rho \in Q^\bullet$ and $i \in \mathbb{N}_1$, if ρ has at least i elements, we let: $\rho[i] \in Q$ denote the i -th element in ρ , in particular $\rho[1] \in Q$ is the first element of ρ ; $\rho[: i] \in Q^+$ denotes the non-empty finite sequence $\rho_1 \cdots \rho_i \in Q^+$; $\rho[i :] \in Q^\bullet$ denotes the non-empty sequence $\rho_i \cdot \rho_{i+1} \cdots \in Q^\bullet$, in particular we have $\rho[1 :] = \rho$.

For the remainder of this section, we fix a non-empty set of propositions Prop .

2.1 Structures

Usually, ATL-formulas are interpreted on concurrent game structures, i.e. games where, at each state, the concurrent actions of several agents have an impact on the next state reached. A special kind of concurrent game structures are turn-based game structures, where each state belongs to a specific agent who decides what the next state is. Here, we introduce only this special kind of games mainly due to a lack of space, but also because all of our hardness results, presented in Table 1, hold even when only considering turn-based game structures.

► **Definition 1.** A turn-based game structure (TGS for short) $T = \langle Q, I, \text{Succ}, \text{Ag}, \alpha, \text{Prop}, \pi \rangle$ is a tuple where: Q is a finite set of states; $I \subseteq Q$ is the set of initial states; $\text{Succ} : Q \rightarrow 2^Q \setminus \emptyset$ maps each state to its set of successors; $\text{Ag} \subseteq \mathbb{N}$ denotes the set of agents; $\alpha : Q \rightarrow \text{Ag}$ maps each state to the agent owning it; and $\pi : Q \mapsto 2^{\text{Prop}}$ maps each state $q \in Q$ to the set of propositions that hold in q . A state q is said to be self-looping if $q \in \text{Succ}(q)$. A structure is self-looping if all of its states are self-looping.

For all coalitions of agents $A \subseteq \text{Ag}$, a strategy s_A for the coalition A is a function $s_A : Q^+ \rightarrow Q$ such that, for all $\rho = \rho_1 \cdots \rho_n \in Q^+$, if $\alpha(\rho_n) \in A$, then $s_A(\rho) \in \text{Succ}(\rho_n)$. We denote by S_A the set of strategies for the coalition A . Then, from any state $q \in Q$, we define the set $\text{Out}(q, s_A)$ of infinite paths compatible with the strategy s_A from q : $\text{Out}(q, s_A) := \{\rho \in Q \cdot Q^\omega \mid \forall i \in \mathbb{N}_1 : \alpha(\rho[i]) \in A \implies \rho[i+1] = s_A(\rho[:i])\}$.

Finally, the size $|T|$ of the turn-based structure T is equal to: $|T| = |Q| + |\text{Ag}| + |\text{Prop}|$.

Unless otherwise stated, a turn-based structure T will always refer to the tuple $T = \langle Q, I, \text{Succ}, A, \alpha, \text{Prop}, \pi \rangle$.

There are also special kinds of turn-based structures of interest for us, introduced below.

► **Definition 2.** A Kripke structure is a turn-based structure with only one agent. A linear structure is a Kripke structure such that: $|I| = 1$, and for all $q \in Q$, we have $|\text{Succ}(q)| = 1$. Finally, a turn-based structure is size-1 if $|Q| = 1$.

Unless otherwise stated, a Kripke structure K will always refer to a tuple $\langle Q, I, \text{Succ}, \text{Prop}, \pi \rangle^1$.

We have introduced the notion of linear structures as we are going to interpret LTL-formulas on them. In the literature, they are usually interpreted on ultimately periodic words. However, both models are equivalent and can be encoded into each other straightforwardly.

2.2 ATL, CTL and LTL formulas

The LTL, CTL and ATL-formulas that we consider throughout this paper use the following temporal operators: **X** (neXt), **F** (Future), **G** (Globally), **U** (Until), **R** (Release), **W** (Weak until), **M** (Mighty release). We group these operators into the sets of unary and binary operators: $\text{Op}_{\text{Un}} := \{\neg, \mathbf{X}, \mathbf{F}, \mathbf{G}\}$ and $\text{Op}_{\text{Bin}}^{\text{tp}} := \{\mathbf{U}, \mathbf{R}, \mathbf{W}, \mathbf{M}\}$. We also let $\text{Op}_{\text{Bin}}^{\text{lg}}$ be the set of all logical binary operators, i.e. classical logical operators, along with their negations: $\text{Op}_{\text{Bin}}^{\text{lg}} := \{\vee, \wedge, \implies, \iff, \nabla, \neg\vee, \neg\wedge, \neg\implies, \neg\iff, \neg\iff\}$ (we have $|\text{Op}_{\text{Bin}}^{\text{lg}}| = 10$).

To define ATL-formulas, we consider two types of formulas: state formulas – where strategic operators occur, denoted with the Greek letter ϕ – and path formulas – where temporal operators occur, denoted with the Greek letter ψ . Consider some $\text{U}^t \subseteq \text{Op}_{\text{Un}}$, $\text{B}^t \subseteq \text{Op}_{\text{Bin}}^{\text{tp}}$, and $\text{B}^l \subseteq \text{Op}_{\text{Bin}}^{\text{lg}}$. For all $k \in \mathbb{N}_1$, we denote by $\text{ATL}^k(\text{Prop}, \text{U}^t, \text{B}^t, \text{B}^l)$ the set of ATL^k -state formulas defined by the grammar:

$$\phi ::= p \mid \neg\phi \mid \phi * \phi \mid \langle\langle A \rangle\rangle\psi \qquad \psi ::= *_1\phi \mid \phi *_2\phi$$

¹ In Kripke structures, there is only one agent, thus Ag and α are irrelevant.

where ϕ is a state-formula, ψ is a path formula, $p \in \text{Prop}$, $* \in \mathbf{B}^l$, $A \subseteq [1, \dots, k]$ is a subset of agents, $*_1 \in \mathbf{U}^t \setminus \{\neg\}$, and $*_2 \in \mathbf{B}^t$. We denote by ATL^k the set of all ATL^k -state formulas ϕ . Note that CTL-formulas are $\text{ATL}^1(\text{Prop}, \mathbf{U}^t, \mathbf{B}^t, \mathbf{B}^l)$ -formulas. Hence, there are only two possible strategic operator: $\langle\langle \emptyset \rangle\rangle$, usually denoted \forall , and $\langle\langle \{1\} \rangle\rangle$ usually denoted \exists . We define LTL-formulas as ATL^1 -formulas using only the quantifier \exists . Since LTL-formulas are interpreted on linear structures, where each state has exactly one successor, the strategic operators used have no impact on the satisfaction of the formula. For readability, we will depict LTL-formulas without the \exists quantifier.

The set of sub-formulas $\text{SubF}(\phi)$ of a formula ϕ is then defined inductively as follows: $\text{SubF}(\phi) := \{\phi\} \cup S$ where $S := \emptyset$ if $\phi = p \in \text{Prop}$, $S := \text{SubF}(\phi')$ if $\phi \in \{\neg\phi', \langle\langle A \rangle\rangle *_1 \phi'\}$ and $S := \text{SubF}(\phi_1) \cup \text{SubF}(\phi_2)$ if $\phi \in \{\phi_1 * \phi_2, \langle\langle A \rangle\rangle(\phi_1 *_2 \phi_2)\}$. The size $|\phi|$ of a formula is then defined as its number of sub-formulas: $|\phi| := |\text{SubF}(\phi)|$. We also denote by $|\phi|_{\text{bin}}$ the number of sub-formulas of ϕ using a binary operator, $|\phi|_{\text{bin}} := |\text{SubBin}(\phi)|$ with: $\text{SubBin}(\phi) := \{\phi_1 * \phi_2 \in \text{SubF}(\phi) \mid \phi_1, \phi_2 \in \text{SubF}(\phi), * \in \text{Op}_{\text{Bin}}^{\text{tp}} \cup \text{Op}_{\text{Bin}}^{\text{lg}}\}$.

We interpret ATL-formulas over TGS using the standard definitions [1]. That is, given a state q and a state formula ϕ , the fact q satisfies ϕ , denoted $q \models \phi$, is defined inductively:

$$\begin{aligned} q \models p & \quad \text{iff } p \in \pi(q) & q \models \phi_1 * \phi_2 & \quad \text{iff } (q \models \phi_1) * (q \models \phi_2) = \text{True} \\ q \models \neg\phi & \quad \text{iff } q \not\models \phi & q \models \langle\langle A \rangle\rangle\psi & \quad \text{iff } \exists s_A \in \mathbf{S}_A, \forall \pi \in \text{Out}(q, s), \pi \models \psi \end{aligned}$$

where $* \in \text{Op}_{\text{Bin}}^{\text{lg}}$ is a binary operator seen as a boolean function $* : \mathbb{B} \times \mathbb{B} \rightarrow \mathbb{B}$ with $\mathbb{B} := \{\text{True}, \text{False}\}$. Furthermore, given a path $\pi \in Q^\omega$ and a path formula ψ , the fact that ψ holds for the path π , also denoted $\pi \models \psi$, is defined inductively as follows:

$$\begin{aligned} \pi \models \mathbf{X}\phi & \quad \text{iff } \pi[2:] \models \phi; & \pi \models \phi_1 \mathbf{U}\phi_2 & \quad \text{iff } \exists i \in \mathbb{N}_1, \pi[i:] \models \phi_2 \text{ and} \\ \pi \models \mathbf{F}\phi & \quad \text{iff } \exists i \in \mathbb{N}_1, \pi[i:] \models \phi; & & \quad \forall 1 \leq j \leq i-1, \pi[j:] \models \phi_1 \\ \pi \models \mathbf{G}\phi & \quad \text{iff } \forall i \in \mathbb{N}_1, \pi[i:] \models \phi & \pi \models \phi_1 \mathbf{R}\phi_2 & \quad \text{iff } \pi \models \neg(\neg\phi_1 \mathbf{U}\neg\phi_2) \\ \pi \models \phi_1 \mathbf{W}\phi_2 & \quad \text{iff } \pi \models (\phi_1 \mathbf{U}\phi_2) \vee \mathbf{G}\phi_1 & \pi \models \phi_1 \mathbf{M}\phi_2 & \quad \text{iff } \pi \models (\phi_1 \mathbf{R}\phi_2) \wedge \mathbf{F}\phi_1 \end{aligned}$$

An ATL-formula ϕ accepts a TGS T , denoted by $T \models \phi$, if $q \models \phi$ for all initial states $q \in I$, otherwise it rejects it. Given two formulas ϕ, ϕ' , we write $\phi \implies \phi'$ if, for all TGS T , if $T \models \phi$, then $T \models \phi'$. We write $\phi \equiv \phi'$ when $\phi \implies \phi'$ and $\phi' \implies \phi$.

2.3 Learning decision problem

We define the LTL, CTL and ATL learning problems below, where models for LTL, CTL, and ATL are linear structures, Kripke structures and turn-based game structures, respectively.

► **Definition 3.** Let $\text{TL} \in \{\text{LTL}, \text{CTL}, \text{ATL}^k \mid k \in \mathbb{N}_1\}$ and consider some sets of operators $\mathbf{U}^t \subseteq \text{Op}_{\text{Un}}$, $\mathbf{B}^t \subseteq \text{Op}_{\text{Bin}}^{\text{tp}}$ and $\mathbf{B}^l \subseteq \text{Op}_{\text{Bin}}^{\text{lg}}$. For all $n \in \mathbb{N} \cup \{\infty\}$, we denote by $\text{TL}_{\text{Learn}}(\mathbf{U}^t, \mathbf{B}^t, \mathbf{B}^l, n)$ the decision problem:

- *Input:* $(\text{Prop}, \mathcal{P}, \mathcal{N}, B)$ where Prop is a set of propositions, \mathcal{P}, \mathcal{N} are two finite sets of models for TL, and $B \in \mathbb{N}$.
- *Output:* yes if and only if there exists a TL-formula $\varphi \in \text{TL}(\text{Prop}, \mathbf{U}^t, \mathbf{B}^t, \mathbf{B}^l)$ such that $|\varphi| \leq B$, $|\varphi|_{\text{bin}} \leq n$, and φ is separating, i.e. such that : for all $X \in \mathcal{P}$ (resp. $X \in \mathcal{N}$), we have $X \models \varphi$ (resp. $X \not\models \varphi$).

The size of the input is equal to $|\text{Prop}| + |\mathcal{P}| + |\mathcal{N}| + B$ (i.e. B is written in unary).

As the model checking problems for LTL, CTL, ATL are in P [1], it follows that the learning problems for all these logics are in NP, with a straightforward guess-and-check subroutine.

► **Proposition 4.** For all $U^t \subseteq \text{Op}_{U^n}$, $B^t \subseteq \text{Op}_{\text{Bin}}^{\text{tp}}$, $B^l \subseteq \text{Op}_{\text{Bin}}^{\text{lg}}$, $n \in \mathbb{N} \cup \{\infty\}$, and $\text{TL} \in \{\text{LTL}, \text{CTL}, \text{ATL}^k \mid k \in \mathbb{N}_1\}$, the decision problem $\text{TLearn}(U^t, B^t, B^l, n)$ is in NP.

2.4 Hitting set problem

We recall below the NP-complete problem from which we will establish almost all of our (NP-hardness) reductions.

- **Definition 5 (Hitting set problem).** We denote by *Hit* the following decision problem:
- *Input:* a triple (l, C, k) where $l \in \mathbb{N}_1$, $C = C_1, \dots, C_n$ are non-empty subsets of $[1, \dots, l]$
 - *Output:* yes iff there is a subset $H \subseteq [1, \dots, l]$ of size at most k such that, we have $H \cap C_i \neq \emptyset$ for all $1 \leq i \leq n$. In such a case, the set H is called a hitting set.

In the following, if (l, C, k) is an instance of the hitting set problem, then C refers to C_1, \dots, C_n for some $n \in \mathbb{N}_1$.

3 Learning with unbounded use of binary operators

First, we consider the case of learning a formula with arbitrarily many occurrences of binary operators. The main result of this section is stated in Theorem 6 below.

► **Theorem 6.** Let $B^t \subseteq \text{Op}_{\text{Bin}}^{\text{tp}}$ and $B^l \subseteq \text{Op}_{\text{Bin}}^{\text{lg}}$ such that $B^t \cup B^l \neq \emptyset$. Then, for all $U^t \subseteq \text{Op}_{U^n}$, the decision problem $\text{LTL}_{\text{Learn}}(U^t, B^t, B^l, \infty)$ is NP-hard.

In the passive learning setting that we consider, the size of the formulas is crucial due to the upper bound B . Therefore, although it is possible to express e.g. disjunctions with conjunctions and negations, since doing so affects the size of the formulas involved, if we have proved that a learning problem is NP-hard with the operators \vee, \neg , it does not imply a priori that it is also NP-hard with the operator \wedge . Hence, for the sake of completeness, we consider all those fourteen binary operators (ten logical, four temporal), although it seems that some of these binary operators (like \vee or \wedge) make much more sense to consider than others (like \Leftrightarrow or $\neg \Leftrightarrow$). Since these operators behave differently, we cannot do a single reduction working for all these operators at once. However, we do partition these operators in different groups and exhibit a reduction per group of operators.

Most of the reductions use only size-1 structures, that are (almost) entirely defined by the subset of propositions labeling their only state. In addition, most of the reductions are done from the hitting set problem. In that case, how we extract a hitting set from a (small enough) separating formula relies only on the variables that need to occur in a formula separating the positive and negative structures, regardless of the operators involved.

We start with the operators $\vee, \Rightarrow, \Leftarrow$, i.e. we assume that $B^l \cap \{\vee, \Rightarrow, \Leftarrow\} \neq \emptyset$. The reduction for this case is actually a straightforward adaptation of the proof of [27, Theorem 2]. We describe it here. Given an instance (l, C, k) of the hitting set problem, we let $\text{Prop} := \{a_j, b_j \mid 1 \leq j \leq l\}$. Furthermore, for all subsets $T \subseteq [1, \dots, l]$, we let $\mathcal{L}(T)$ denote a size-1 (linear) structure whose only state is labeled by the set $\{a_j, b_{j'} \mid j \in T, j' \notin T\}$. Then, we let $\text{In}^{\vee, \Rightarrow, \Leftarrow} := (\text{Prop}, \mathcal{P}, \mathcal{N}, B)$ for $\mathcal{P} := \{\mathcal{L}(C_i) \mid 1 \leq i \leq n\}$, $\mathcal{N} := \{\mathcal{L}(\emptyset)\}$, and $B := 2k - 1$. Let us illustrate this reduction on a simple example. Assume that $l = 4$, $C = (\{1, 2, 3\}, \{2, 4\}, \{1, 4\})$, and $k = 2$. Then, the sets labeling the only state of the positive structures are $\{a_1, a_2, a_3, b_4\}$, $\{b_1, a_2, b_3, a_4\}$, and $\{a_1, b_2, b_3, a_4\}$ while the set labeling the only state of the negative structure is $\{b_1, b_2, b_3, b_4\}$. Furthermore, $B = 3$. Then, $H := \{1, 4\}$ is the a hitting set with $|H| \leq 2$, while $\varphi_{\vee} := a_1 \vee a_4$, $\varphi_{\Rightarrow} := b_1 \Rightarrow a_4$, and $\varphi_{\Leftarrow} := a_1 \Leftarrow b_4$ are all separating formulas with $|\varphi_{\vee}| = |\varphi_{\Rightarrow}| = |\varphi_{\Leftarrow}| \leq 3$.

We claim that (l, C, k) is a positive instance of **Hit** iff $\text{In}^{\vee, \Rightarrow, \Leftarrow}$ is a positive instance of $\text{LTL}_{\text{Learn}}(\mathbf{U}^t, \mathbf{B}^t, \mathbf{B}^l, \infty)$. Indeed, given a hitting set $H = \{i_1, \dots, i_r\}$ with $r \leq k$, one can check that the LTL-formula $\varphi_{\vee} := \bigvee_{1 \leq x \leq r} a_{i_x}$ of size $2r - 1 \leq B$ accepts \mathcal{P} and rejects \mathcal{N} . Note that, the LTL-formulas $\varphi_{\Rightarrow} := b_{j_1} \Rightarrow (b_{j_2} \Rightarrow (\dots \Rightarrow a_{j_r}))$ and $\varphi_{\Leftarrow} := ((a_{j_1} \Leftarrow b_{j_2}) \Leftarrow \dots) \Leftarrow b_{j_r}$ of size $2r + 1 \leq B$ also accept \mathcal{P} and reject \mathcal{N} . On the other hand, consider an LTL-formula φ of size at most B that accepts \mathcal{P} and rejects \mathcal{N} . We let $H := \{1 \leq j \leq l \mid a_j \text{ or } b_j \text{ occurs in } \varphi\}$. Since $|\varphi| \leq B$, we have $|H| \leq k$. Furthermore, consider any $1 \leq i \leq n$. Let us consider the set S_i (resp. S) labeling the only state of the structure $\mathcal{L}(C_i)$ (resp. $\mathcal{L}(\emptyset)$). We have $\Delta(S_i, S) := S_i \setminus S \cup S \setminus S_i = \{a_j, b_j \mid j \in C_i\}$. One can then show (rather straightforwardly, by induction on LTL-formulas) that, since φ accepts $\mathcal{L}(C_i)$ and rejects $\mathcal{L}(\emptyset)$, at least one variable in $\Delta(S_i, S)$ occurs in φ . That is, $C_i \cap H \neq \emptyset$ and H is a hitting set of size at most k .

In fact, the reduction for the operators $\wedge, \neg \Rightarrow, \neg \Leftarrow$ is obtained from the above one by reversing the positive and negative sets (the arguments are almost identical).

We then handle the operators $\neg \vee, \neg \wedge$. The above reductions cannot be used since, when the operator $\neg \wedge$ (or the operator $\neg \vee$) is used successively, the formula obtained is semantically equivalent to an alternation of conjunctions and disjunctions. For instance, consider six variables $r_1, r_2, r_3, r_4, x_1, x_2$ to use in a single LTL-formula using only the $\neg \wedge$ operator, e.g.: $\varphi := r_1 \neg \wedge (x_1 \neg \wedge (r_2 \neg \wedge (x_2 \neg \wedge (r_3 \neg \wedge r_4))))$. It is semantically equivalent to: $\varphi \equiv \neg r_1 \vee (x_1 \wedge (\neg r_2 \vee (x_2 \wedge (\neg r_3 \vee \neg r_4))))$. This is in sharp contrast with the above-formulas $\varphi_{\vee}, \varphi_{\Leftarrow}$ and φ_{\Rightarrow} . To circumvent this difficulty, we change the reduction by adding propositions labeling the only state of all the positive size-1 linear structures (but not the only state of all the negative ones). We can then place these propositions where x_2 and x_4 were in the above formula. That way, we semantically obtain a disjunction on relevant variables r_1, r_2, r_3, r_4 . The obtained reduction is slightly more subtle than the previous ones.

Before considering the last two logical operators $\Leftrightarrow, \neg \Leftrightarrow$, we handle the temporal operators **W, M**. The two previous reductions only use size-1 structures. On such structures, the temporal operators **W, M** are actually equivalent to \vee and \wedge respectively. Hence, the reductions for \vee and \wedge can also be used as is for the operators **W** and **M** respectively.

We then handle the final two logical operators $\Leftrightarrow, \neg \Leftrightarrow$. These operators are unlike the other operators. Let us give an intuition of how the learning problems with these operators behave. Consider an LTL-formula φ using only the operators \neg, \Rightarrow and $\neg \Leftrightarrow$ and a size-1 structure \mathcal{L} . Let S denote the set of propositions labeling the only state of \mathcal{L} . We let $\text{Neg}(\varphi)$ denote the number of occurrences of the operators $\neg, \neg \Leftrightarrow$ in φ . We also let $\text{NbOc}_{\bar{S}}(\varphi)$ denote the number of occurrences of the propositions not in S in φ . Then, one can realize that $\mathcal{L} \models \varphi$ if and only if $\text{Neg}(\varphi)$ and $\text{NbOc}_{\bar{S}}(\varphi)$ have the same parity. This simple observation suggests that the learning problem with the operators $\Leftrightarrow, \neg \Leftrightarrow$ is linked to modulo-2 calculus. The reduction for these operators is established from an NP-complete problem dealing with modulo-2 calculus, known as the Coset Weight problem [5].

Finally, we handle the temporal operators **U** and **R**. On size-1 structures, for all LTL-formulas φ_1, φ_2 , we have the following equivalences: $\varphi_1 \mathbf{U} \varphi_2 \equiv \varphi_1 \mathbf{R} \varphi_2 \equiv \varphi_2$. That is, contrary to the temporal operators **W** and **M**, on size-1 structures, **U** and **R** are equivalent to unary operators. Hence, the reduction that we consider does not involve only size-1 structures. It is once again established from the hitting set problem, though the construction and the correctness proof are more involved than for the above cases.

On top of that, for all sets of operators, ATL learning is at least as hard as CTL learning, which is itself at least as hard as LTL learning. Thus, from Theorem 6, we obtain that CTL and ATL learning with unbounded use of binary operators are NP-hard. This justifies the leftmost column of Table 1.

4 Learning with a bounded amount of binary operators

Since, with unbounded use of binary operators, all the learning problems are NP-hard, we focus on learning formulas where the number of occurrences of binary operators is bounded. Note that the bound n parameterizes the decision problem itself, and therefore is independent of the input. For simplicity, we restrict ourselves to formulas that do not use at all binary temporal operators. Before we dive into the details of our results as summarized in the rightmost column of Table 1, let us first argue why this fragment is interesting to focus on.

4.1 Expressivity

The passive learning problem that we consider in this paper bounds the size of the formulas considered. This is because we want a separating formula not to overfit the input (i.e. not to simply describe the positive and negative models). However, another benefit is that the smaller the formulas, the more understandable they are for users. Similarly, using too many binary operators could make the formulas hard to grasp, regardless of their size.

In addition, there are examples of interesting specifications that can be expressed with a bounded amount of binary operators. We give three examples below with LTL-formulas.

Consider first so-called “mode-target” formulas of the shape $\bigwedge_j (\mathbf{F} \mathbf{G} M_j \Rightarrow \bigvee_i \mathbf{F} \mathbf{G} T_{i,j})$, where all $M_j, T_{i,j}$ are propositions. These types of formulas were introduced in [4] and exhibit two interesting features: the corresponding LTL-synthesis problem is tractable, and these formulas express an interesting property, which can be summarized as follows: if a model eventually settles in a mode M_j , then it should eventually settle in one of the target $T_{i,j}$. Interestingly, when the number of different modes and targets that a system can have is fixed, then the number of binary operators sufficient to express such specification is also bounded.

Similarly, there are also interesting specifications related to “generalized reactivity” (from [32] for LTL-formulas). Such specifications are of the shape $\bigwedge_i \mathbf{G} \mathbf{F} \psi_i \Rightarrow \bigwedge_i \mathbf{G} \mathbf{F} \psi'_i$, where all formulas ψ_i and ψ'_i do not feature at all temporal operators. As such, up to introducing additional propositions, these could be expressible with few binary operators. These formulas can be read as an implication between assumptions and guarantees. As above, when the number of assumptions and guarantees is bounded, then the number of binary operators sufficient to express such formulas also is.

Finally, one of the popular LTL learning tools, Scarlet [35], relies on a fragment of LTL, directed LTL and its dual, which uses unary temporal operators and binary logical operators only. In these fragments, formulas of a fixed length (a search parameter they define) can use several of \mathbf{F} , \mathbf{G} and \mathbf{X} operators while using only bounded occurrences of \wedge and \vee operators.

4.2 Abstract recipes

The six decision problems captured in the rightmost column of Table 1 are of two kinds: three are NP-complete, while three others are below NP. In fact, the proofs of all three results of the same kind will follow the same abstract recipes. We present them below.

Recipe for the membership-below-NP proofs. Let TL denote either LTL formulas, or CTL formulas without the operator \mathbf{X} , or ATL^2 formulas with only one unary operator \mathbf{F} or \mathbf{G} (i.e. one of the three logical fragment for which the corresponding decision problem is below NP). Then, we follow the two steps below:

- A) First, we show that given the set of propositions Prop and the bound B , there is a set of relevant TL-formulas $\text{RelForm}(\text{Prop}, B)$ such that: 1) For all TL(Prop)-formulas ϕ of size at most B , there is a formula $\phi' \in \text{RelForm}(\text{Prop}, B)$ such that $\phi \equiv \phi'$; and 2) the size of $\text{RelForm}(\text{Prop}, B)$ is polynomial in $|\text{Prop}|$ and B .

- B) Second, we show that for all TL-formulas $\phi \in \text{RelForm}(\text{Prop}, B)$, deciding if ϕ satisfies a TL-model M can be done, depending on $|\text{Prop}|, B, |M|$, within the resources allowed, i.e. logarithmic space for the LTL case, non-deterministic logarithmic space for the CTL case, and polynomial time for the ATL² case.

Due to a lack of space, in this paper we will only present these two steps in the context of formulas that do not use any binary operator. Since the occurrences of binary operators is bounded in any case, the arguments are essentially the same for the general case. For instance, for the first step, from the result established for formulas without binary operators, we can straightforwardly deduce the result for all formulas, by induction on the bound n . That way, we obtain that $|\text{RelForm}(\text{Prop}, B)|$ could be exponential in the bound n , but this does not have an impact complexity-wise, since n is fixed.

Recipe for the NP-hardness proofs. The formulas that we consider only use a bounded amount of binary operators. Thus, contrary to the NP-hardness reductions of Section 3, here, our NP-hardness proofs do not rely on binary operators. In fact, these binary operators make it harder to argue about how the permitted unary operators interact. For this reason, our proof of NP-hardness is decomposed into two steps. We first exhibit reductions for the learning problems without binary operators. Then, from these reductions, we devise reductions for the learning problems with bounded occurrences of binary operators. We present in details the former reductions in this paper and give intuition behind the later reductions below.

Let $n \in \mathbb{N}$ and $*$ $\in \text{Op}_{\text{Bin}}^{\text{lg}}$ be a binary (non-temporal) operator. We consider n propositions $\{p_1, \dots, p_n\}$ and we define multiple size-1 structures using the propositions $\{p_1, \dots, p_n\}$ forming two sets $\mathcal{A}_{n,*}$ and $\mathcal{B}_{n,*}$. The idea is that to distinguish these two sets, a separating formula will necessarily feature all the propositions $\{p_1, \dots, p_n\}$. In fact, from a positive and a negative sets of structures \mathcal{P} and \mathcal{N} on the set of proposition $\{p\}$ ² (which is the only proposition that unary formulas can use in our reductions), we can show the following: if a formula of size at most $B + 2n$, with at most n occurrences of binary operators, separates both \mathcal{P} and \mathcal{N} , and $\mathcal{A}_{n,*}$ and $\mathcal{B}_{n,*}$, then there is a unary formula of size at most B that separates \mathcal{P} and \mathcal{N} .³ That way, a reduction for the learning problem without binary operators can be translated (in logspace) into a reduction for the learning with bounded occurrences of binary operators. Note that the arguments presented in this paragraph are not straightforward to formally state and prove (this is handled in Theorem “Proving NP-hardness without binary operators is sufficient” in the extended version [8]).

Let us now consider how we handle the reduction without binary operators. From an instance (l, C, k) of the hitting problem, we proceed as follows. We define a sample of structures (and a bound B) such that all separating formulas have a specific shape, and there is a bijection between subsets $H \subseteq [1, \dots, l]$ and formulas $\varphi(l, H)$ of that specific shape. This correspondence allows us to extract a hitting set. More specifically, we follow the abstract recipe below:

- (a) We define the bound B and positive and negative structures that “eliminate” certain operators or pattern of operators from any potential separating formula. This way we ensure that any separating formula will be of the form $\varphi(l, H)$, for some $H \subseteq [1, \dots, l]$.
- (b) We define a negative structure satisfied by a formula $\varphi(l, H)$ if and only if $|H| \geq k + 1$.
- (c) For all $1 \leq i \leq n$, we define a positive structure that a formula $\varphi(l, H)$ accepts if and only if $H \cap C_i \neq \emptyset$.

² In fact, for technical reason, in [8], we use two propositions $\{p, \bar{p}\}$.

³ Actually, we can also show the converse (which is important for us to prove that the reduction is correct).

19:10 Learning LTL, CTL and ATL Formulas

By construction, the instance of the learning decision problem that we obtain is a positive instance if and only if the hitting set instance (l, C, k) also is. Furthermore, note that in all three cases, this reduction can be computed in logspace.

4.3 LTL learning

We start with LTL learning. We have the proposition below.

► **Proposition 7.** *For all sets of unary operators $U^t \subseteq \text{Op}_{U_n}$, sets of binary (non-temporal) operators $B^l \subseteq \text{Op}_{B_{\text{in}}}^{\text{lg}}$, and $n \in \mathbb{N}$, the decision problem $\text{LTL}_{\text{Learn}}(U^t, \emptyset, B^l, n)$ is in L.*

We present Steps A and B of Section 4.2 in the case $n = 0$. Toward Step A, we have the equivalences below (see e.g. [27, Prop. 8]), which imply the corollary that follows.

► **Observation 8.** *For all LTL-formulas φ and $k \in \mathbb{N}$, we have: 1) $\mathbf{F} \mathbf{X}^k \varphi \equiv \mathbf{X}^k \mathbf{F} \varphi$, $\mathbf{G} \mathbf{X}^k \varphi \equiv \mathbf{X}^k \mathbf{G} \varphi$; 2) $\mathbf{F} \mathbf{F} \varphi \equiv \mathbf{F} \varphi$, $\mathbf{G} \mathbf{G} \varphi \equiv \mathbf{G} \varphi$; 3) $\mathbf{F} \mathbf{G} \mathbf{F} \varphi \equiv \mathbf{G} \mathbf{F} \varphi$, $\mathbf{G} \mathbf{F} \mathbf{G} \varphi \equiv \mathbf{F} \mathbf{G} \varphi$.*

► **Corollary 9.** *Consider a set of propositions Prop . We let $\text{Lit}(\text{Prop}) := \{x, \neg x \mid x \in \text{Prop}\}$ and $\text{LTL}_{U_n}(\text{Prop}) := \{\mathbf{X}^k x, \mathbf{X}^k \mathbf{F} x, \mathbf{X}^k \mathbf{G} x, \mathbf{X}^k \mathbf{F} \mathbf{G} x, \mathbf{X}^k \mathbf{G} \mathbf{F} x \mid k \in \mathbb{N}, x \in \text{Lit}(\text{Prop})\}$.*

Then, for any LTL-formula $\varphi \in \text{LTL}(\text{Prop}, \text{Op}_{U_n}, \emptyset, B^l, 0)$, there is an LTL-formula $\varphi' \in \text{LTL}_{U_n}(\text{Prop}) \cap \text{LTL}(\text{Prop}, \text{Op}_{U_n}, \emptyset, B^l, 0)$ such that $\varphi \equiv \varphi'$ and $|\varphi'| \leq |\varphi|$.

Proof sketch. With the equivalences 1) from Observation 8, we can push the \mathbf{X} operators in φ at the beginning of the formula. The equivalences 2) and 3) from Observation 8 ensure that it is possible to have at most two nested \mathbf{F}, \mathbf{G} operators in the resulting formula φ' . ◀

The set of relevant formulas $\text{RelForm}(\text{Prop}, B)$ is then obtained directly from the set of formulas $\text{LTL}_{U_n}(\text{Prop})$. Note that however, how it is obtained depends on the exact operators in U^t . For instance, if $\mathbf{G} \notin U^t$ while $\neg, \mathbf{F} \in U^t$, we should replace the occurrences of \mathbf{G} in formulas in $\text{RelForm}(\text{Prop}, B)$ by $\neg \mathbf{F} \neg$. Nonetheless, in any case, we obtain a set $\text{RelForm}(\text{Prop}, B)$ of relevant formulas whose number of elements is linear in $|\text{Prop}| \cdot B$. This concludes the arguments for Step A. As for Step B, one can realize that since there are at most two nested \mathbf{F}, \mathbf{G} operators in formulas in $\text{RelForm}(\text{Prop}, B)$, then checking that they hold on a linear structure can be done in logarithmic space (because it suffices to have a constant number of pointers browsing the structure).

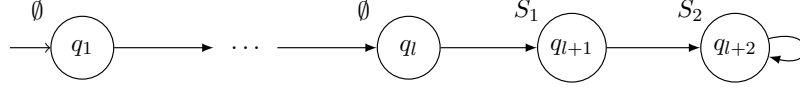
4.4 CTL learning

Consider now the more involved case of CTL learning. As can be seen in Table 1, we distinguish two cases: with and without the operator \mathbf{X} .

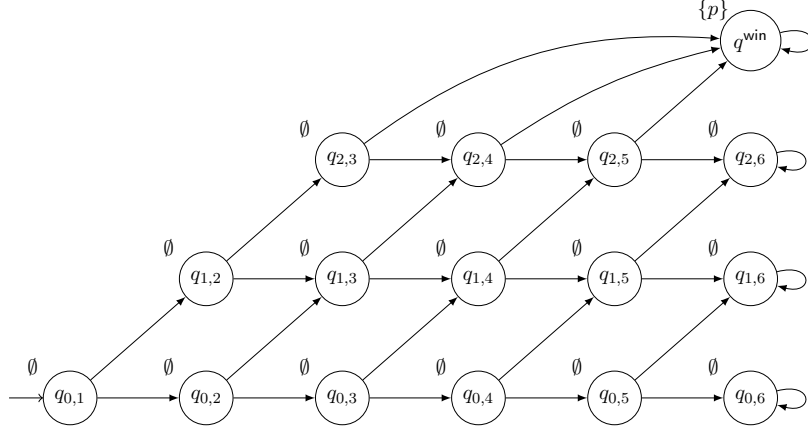
Assume that $\mathbf{X} \in U^t$. The goal is to show the theorem below.

► **Theorem 10.** *For all sets $U^t \subseteq \text{Op}_{U_n}$, $B^l \subseteq \text{Op}_{B_{\text{in}}}^{\text{lg}}$, and bound $n \in \mathbb{N}$, if $\mathbf{X} \in U^t$, then the decision problem $\text{CTL}_{\text{Learn}}(U^t, \emptyset, B^l, n)$ is NP-hard.*

As stated in Section 4.2, we argue the theorem in the case $n = 0$. Recall that in that case we consider a single proposition $\{p\}$. Consider an instance (l, C, k) of the hitting set problem Hit. We follow the three Steps a, b, and c. Toward Step a, we define Kripke structures that prevent the use of the operators $\mathbf{F}, \mathbf{G}, \neg$. To do so, we let $B := l + 1$ and for two sets $S_1, S_2 \subseteq \{p\}$, we consider the Kripke structure K_{l, S_1, S_2} that is depicted in Figure 1. These structures satisfy the lemma below.



■ **Figure 1** The structure K_{l,S_1,S_2} where $S_1 \subseteq \{p\}$ (resp. $S_2 \subseteq \{p\}$) labels q_{l+1} (resp. q_{l+2}).



■ **Figure 2** The Kripke structure $K_{\exists > 2}^5$.

► **Lemma 11.** *A formula $\phi \in \text{CTL}(\{p\}, \text{U}^t, \emptyset, \text{B}^l, 0)$ of size at most $l + 1$ accepting $K_{l,\{p\},\emptyset}$ and rejecting $K_{l,\emptyset,\{p\}}$ and $K_{l,\emptyset,\emptyset}$ cannot use the operators \mathbf{F} , \mathbf{G} , \neg .*

Proof sketch. Consider an equivalent CTL-formula ϕ' with $|\phi'| \leq |\phi|$ where negations, if any, occur right before the proposition p . Then, if ϕ' uses the operator \mathbf{G} , it cannot distinguish the structures $K_{l,\{p\},\emptyset}$ and $K_{l,\emptyset,\emptyset}$. Otherwise, if it uses the operator \mathbf{F} , it cannot distinguish the structures $K_{l,\{p\},\emptyset}$ and $K_{l,\emptyset,\{p\}}$. Otherwise, since $K_{l,\{p\},\emptyset}$, $K_{l,\emptyset,\{p\}}$, and $K_{l,\emptyset,\emptyset}$ coincide on the first l states, ϕ' has to use at least l operators \mathbf{X} . Since $|\phi'| \leq l + 1$, it cannot use a negation. Thus ϕ' does not use \mathbf{F} , \mathbf{G} , \neg , and neither does ϕ . ◀

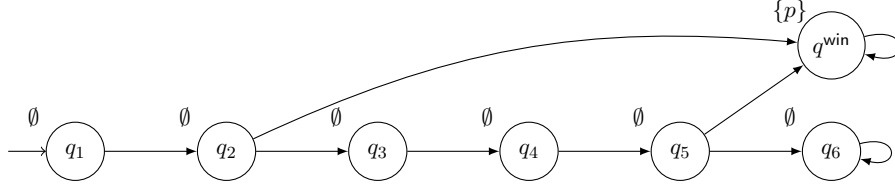
In fact, a CTL-formula $\phi \in \text{CTL}(\{p\}, \text{U}^t, \emptyset, \text{B}^l, 0)$ of size at most $l + 1$ accepting $K_{l,\{p\},\emptyset}$ and rejecting $K_{l,\emptyset,\{p\}}$, $K_{l,\emptyset,\emptyset}$ necessarily uses exactly l operators \mathbf{X} followed by the proposition p . Such a formula is therefore entirely defined by the \mathbf{X} operators before which it uses the \exists quantifier. This suggests the definition below of the CTL-formula $\phi(l, H)$ induced by a subset $H \subseteq [1, \dots, l]$.

► **Definition 12.** *For all $H \subseteq [1, \dots, l]$, we let $\phi(l, H) \in \text{CTL}(\{p\}, \text{U}^t, \emptyset, \text{B}^l, 0)$ denote the CTL-formula defined by $\phi(l, H) := Q_1 \mathbf{X} \dots Q_l \mathbf{X} p$ where, for all $1 \leq i \leq l$, we have $Q_i \in \{\exists, \forall\}$ and $Q_i = \exists$ if and only if $i \in H$.*

For $1 \leq i \leq l + 1$, we let $\phi_i(l, H) := Q_i \mathbf{X} \dots Q_l \mathbf{X} p$ (with $\phi_{l+1}(l, H) := p$).

Let us now turn toward Step b. We define a structure $K_{\exists > k}^l$ with $k + 2$ different levels, where: the single starting state $q_{0,1}$ is at the bottommost level; the proposition p only labels the state q^{win} at the topmost level; and every state of the bottom $k + 1$ levels has a successor at the same level and one level higher. That way, going from $q_{0,1}$ to q^{win} is equivalent to leveling up $k + 1$ times. Furthermore, the top most level can be reached in at most l . An example is depicted in Figure 2. This structure satisfies the lemma below.

► **Lemma 13.** *For all $H \subseteq [1, \dots, l]$, we have $K_{\exists > k}^l \models \phi(l, H)$ if and only if $|H| > k$.*



■ **Figure 3** The Kripke structure $K_{5, \{2,5\}}$.

Consider now Step c. For $C \subseteq [1, \dots, l]$, we define the Kripke structure $K(l, C)$ with $\{q_1, \dots, q_l, q_{l+1}, q^{\text{win}}\}$ as set of states where q^{win} is the only state labeled with p ; and for all $1 \leq j \leq l$, q_j branches to q_{j+1} and, if (and only if) $j \in C$, q_j also branches to q^{win} , as exemplified in Figure 3. Such structures satisfy the lemma below.

► **Lemma 14.** *For all $C, H \subseteq [1, \dots, l]$, we have $K_{(l,C)} \models \phi(l, H)$ if and only if $C \cap H \neq \emptyset$.*

Proof sketch. We can show by induction on $l + 1 \geq j \geq 1$ the property $\mathcal{P}(j)$: $q_j \models \phi_j(l, H)$ if and only if $H \cap [j, \dots, l] \cap C \neq \emptyset$. The lemma is then given by $\mathcal{P}(1)$. ◀

We can finally define the reduction that we consider. We let $\text{In}^{\text{CTL}} := (\{p\}, \mathcal{P}, \mathcal{N}, B)$, with $B := l + 1$, $\mathcal{P} := \{K_{l, \{p\}, \emptyset}, K_i \mid 1 \leq i \leq n\}$ and $\mathcal{N} := \{K_{l, \emptyset, \emptyset}, K_{l, \emptyset, \{p\}}, K_{\exists > k}^l\}$, be an input of the decision problem $\text{CTL}_{\text{Learn}}(\text{U}^t, \emptyset, \text{B}^l, 0)$. By Lemmas 11, 13, 14, In^{CTL} is a positive instance of the decision problem $\text{CTL}_{\text{Learn}}(\text{U}^t, \emptyset, \text{B}^l, 0)$ if and only if (l, C, k) is a positive instance of the decision problem Hit. Theorem 10 follows (in the case $n = 0$).

Assume that $X \notin \text{U}^t$. In that case, the CTL learning problem is now in NL.

► **Theorem 15.** *For all sets of operators $\text{U}^t \subseteq \{\mathbf{F}, \mathbf{G}, \neg\}$, $\text{B}^l \subseteq \text{Op}_{\text{Bin}}^{\text{lg}}$, and bounds $n \in \mathbb{N}$, the decision problem $\text{CTL}_{\text{Learn}}(\text{U}^t, \emptyset, \text{B}^l, n)$ is in NL.*

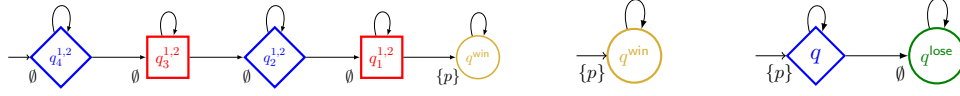
Toward Step A, a crucial observation is that using the operators \mathbf{F} or \mathbf{G} twice in a row is useless. This is stated in the lemma below in the context of ATL-formula because this lemma will be used again in the next subsection.

► **Lemma 16.** *Let $I \subseteq J \subseteq \mathbb{N}$, and ϕ be an ATL-formula. We have:*

$$\langle\langle J \rangle\rangle \mathbf{F} \phi \equiv \langle\langle I \rangle\rangle \mathbf{F} \langle\langle J \rangle\rangle \mathbf{F} \phi \equiv \langle\langle J \rangle\rangle \mathbf{F} \langle\langle I \rangle\rangle \mathbf{F} \phi \quad \langle\langle I \rangle\rangle \mathbf{G} \phi \equiv \langle\langle I \rangle\rangle \mathbf{G} \langle\langle J \rangle\rangle \mathbf{G} \phi \equiv \langle\langle J \rangle\rangle \mathbf{G} \langle\langle I \rangle\rangle \mathbf{G} \phi$$

Proof sketch. We argue the result for \mathbf{F} , the case of \mathbf{G} is dual. We have $\langle\langle J \rangle\rangle \mathbf{F} \phi \implies \langle\langle I \rangle\rangle \mathbf{F} \langle\langle J \rangle\rangle \mathbf{F} \phi$ by definition of \mathbf{F} . Furthermore, if a state q satisfies $\langle\langle I \rangle\rangle \mathbf{F} \langle\langle J \rangle\rangle \mathbf{F} \phi$ then there is a strategy s_I for the coalition I such that eventually a state satisfying $\langle\langle J \rangle\rangle \mathbf{F} \phi$ is surely reached. For all such states q , we consider a strategy s_J^q for the coalition J ensuring to eventually visit a state satisfying ϕ . Then, consider a strategy s'_J for the coalition J that mimics s_I (which is possible since $I \subseteq J$) until a state q satisfying $\langle\langle J \rangle\rangle \mathbf{F} \phi$ is reached, and then switches to the strategy s_J^q . That strategy ensures eventually reaching a state satisfying ϕ . Therefore, $\langle\langle I \rangle\rangle \mathbf{F} \langle\langle J \rangle\rangle \mathbf{F} \phi \implies \langle\langle J \rangle\rangle \mathbf{F} \phi$. This is similar for $\langle\langle J \rangle\rangle \mathbf{F} \langle\langle I \rangle\rangle \mathbf{F} \phi$. ◀

From this, we can actually deduce (this is not direct) that there is a bound $M \in \mathbb{N}$ such that, for any set of propositions Prop and for all $\text{U}^t \subseteq \{\mathbf{F}, \mathbf{G}, \neg\}$, given any $\text{CTL}(\text{Prop}, \text{U}^t, \emptyset, \text{B}^l, 0)$ -formula ϕ , there is an equivalent $\text{CTL}(\text{Prop}, \text{U}^t, \emptyset, \text{B}^l, 0)$ -formula ϕ' , with $|\phi'| \leq M$. Thus, the number of CTL-formulas to consider is linear in $|\text{Prop}|$. As for Step B, consider any such formula ϕ . Since the number of quantifiers it uses is bounded by M and $\text{NL} = \text{coNL}$, we deduce that checking that it satisfies a Kripke structure can be done in NL.



■ **Figure 4** The turn-based structure $T_{4:1,2}$. ■ **Figure 5** On the left T_p , on the right $T_{no\ 2\ G}$.

Proof of NL-hardness. In Table 1, we do not state only that CTL learning without the operator \mathbf{X} is in NL, but also that it is NL-hard. Proving this result is actually straightforward. We exhibit a reduction from the problem of reachability in a graph (which is NL-complete [23]). Given an input (\mathcal{G}, s, t) of that problem, with \mathcal{G} a graph, s the source state and t the target state, we define a positive Kripke structure K that is obtained from \mathcal{G} by making s its only initial state, and t the only state labeled by the proposition p . Additionally, we consider $B := 2$ as the bound, and with an additional structure, we ensure that if there is a separating formula, then the formula $\phi := \exists \mathbf{F} p$ is separating.

4.5 ATL learning

We have seen that CTL learning with the operator \mathbf{X} is NP-hard, which implies that it is also the case for ATL learning. Here, we consider the case of ATL learning without the operator \mathbf{X} . First, let us informally explain why the NP-hardness reduction that we have described above for CTL cannot possibly work without the operator \mathbf{X} . A central aspect of the proof of Lemma 14 is to be able to associate a specific operator in a prospective formula with a specific state in a Kripke structure. That is intrinsically not possible with the operator \mathbf{F} since this operator looks at arbitrarily distant horizons. At least, this is true with CTL-formulas interpreted on Kripke structures. However, with ATL-formulas interpreted on turn-based structures, it is possible to “block the horizon” of \mathbf{F} operators. Indeed, consider the structure of Figure 4, where blue lozenge-shaped states are Agent-1 states, and red square-shaped states are Agent-2’s. Here, one can see that $q_2^{1,2} \not\models \langle\langle 1 \rangle\rangle \mathbf{F} p$ because Agent 2 can enforce to loop on the Agent-2 state $q_1^{1,2}$ and not see the state q^{win} , labeled by p .

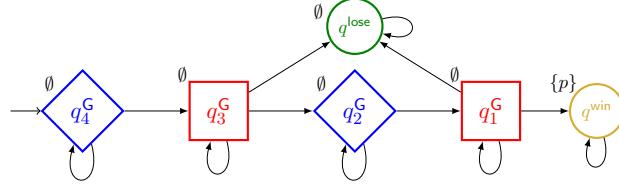
These kinds of turn-based games will be extensively used in the following. In all generality, there are defined as follows: given a pair of agents $i \neq j$ and $l \in \mathbb{N}$, in the turn-based structure $T_{l:i,j}$, there are $l + 1$ self-looping states, alternatively belonging to Agents i and j , that can get closer and closer to the self-looping sink q^{win} , the only state labeled by p . In fact, such structures are linked to alternating-formulas, defined below.

► **Definition 17.** An ATL-formula is positive if it does not use any negation. For a pair of agents $i \neq j$ and $l \in \mathbb{N}$, a positive ATL-formula ϕ is (i, j) -free if it does not use an operator $\langle\langle A \rangle\rangle \mathbf{F}$ with $i, j \in A$. It is (i, j, l) -alternating if it is (i, j) -free and if there are at least l alternating occurrences of operators $\langle\langle A_i \rangle\rangle \mathbf{F}$ with $i \in A_i$ and $\langle\langle A_j \rangle\rangle \mathbf{F}$ with $j \in A_j$.

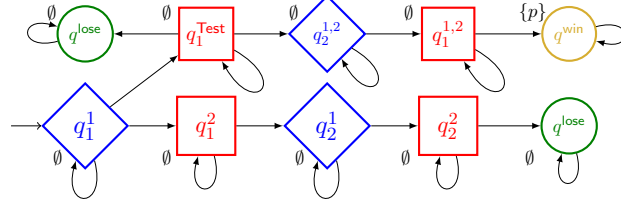
► **Lemma 18.** Consider two agents $i \neq j$, $l \in \mathbb{N}$, and a positive ATL-formula ϕ that is (i, j) -free. The formula ϕ accepts the structure $T_{l:i,j}$ if and only if it is (i, j, l) -alternating.

ATL² learning with $\{\mathbf{F}, \mathbf{G}\} \subseteq \mathbf{U}^t$. Here, all the turn-based structures that we consider use the set of agents $\text{Ag} = \{1, 2\}$. The goal is to show the theorem below.

► **Theorem 19.** For all sets $\mathbf{U}^t \subseteq \text{Op}_{\mathbf{U}^t}$, $\mathbf{B}^l \subseteq \text{Op}_{\mathbf{B}^l}^{\text{lg}}$, and bound $n \in \mathbb{N}$, if $\{\mathbf{F}, \mathbf{G}\} \subseteq \mathbf{U}^t$ and $\mathbf{X} \notin \mathbf{U}^t$, then the decision problem $\text{ATL}_{\text{Learn}}^2(\mathbf{U}^t, \emptyset, \mathbf{B}^l, n)$ is NP-hard.



■ **Figure 6** The structure $T_{no 1 G \geq 2}$.



■ **Figure 7** The structure $T_{2, \{1\}, 2}$.

In the following, to ease the notations, the strategic operators $\langle\langle \emptyset \rangle\rangle$, $\langle\langle \{1\} \rangle\rangle$, $\langle\langle \{2\} \rangle\rangle$, $\langle\langle \{1, 2\} \rangle\rangle$ will simply be denoted \emptyset , 1, 2 and 1, 2 respectively. Consider an instance (l, C, k) of the hitting set problem. We follow the recipe of Subsection 4.2. Here, we want separating formulas to be promising, i.e. to only use the operators $1\mathbf{F}$, $2\mathbf{F}$ and $1\mathbf{G}$. To this end, all the structures we use are self-looping, thus making the operators $\emptyset\mathbf{F}$ and $1, 2\mathbf{G}$ useless.

► **Lemma 20.** *For all ATL-formulas ϕ and self-looping states q , we have: $q \models \phi$ if and only if $q \models \emptyset\mathbf{F}\phi$ if and only if $q \models 1, 2\mathbf{G}\phi$*

Proof. Since q is self-looping the coalition of agents $\{1, 2\}$ has a strategy s such that $\text{Out}(q, s) = \{q^\omega\}$. The lemma follows from the definition of the operators \mathbf{F} and \mathbf{G} . ◀

We also consider the two structures $T_p, T_{no 2\mathbf{G}}$, of Figure 5 satisfying the lemma below.

► **Lemma 21.** *For all ATL-formulas $\phi \in \text{ATL}(\{p\}, \mathbf{U}^t, \emptyset, \mathbf{B}^1, 0)$ accepting $T_p, T_{no 2\mathbf{G}}$ and rejecting $T_{2l+1;1,2}$, there is a promising formula $\phi' \in \text{ATL}(\{p\}, \mathbf{U}^t, \emptyset, \mathbf{B}^1, 0)$ with $|\phi'| \leq |\phi|$ that is equivalent to ϕ on self-looping structures.*

Proof sketch. Consider an ATL-formula ϕ' equivalent to ϕ with $|\phi'| \leq |\phi|$ and with at most one negation occurring before the proposition p . Since ϕ' accepts T_p , it follows that it is positive. By Lemma 20, we can remove the operators $1, 2\mathbf{G}$ and $\emptyset\mathbf{F}$ from ϕ' . Furthermore: ϕ' cannot use $\emptyset\mathbf{G}, 2\mathbf{G}$, since it accepts $T_{no 2\mathbf{G}}$, and it cannot use $1, 2\mathbf{F}$ since it rejects $T_{2l+1;1,2}$. It is therefore promising. ◀

We will also consider $T_{2l;1,2}$ as a positive structure, thus allowing us to focus on $(1, 2, 2l)$ -alternating formulas (recall Lemma 18). Then, we want to associate to a subset $H \subseteq [1, \dots, l]$ a promising $(1, 2, 2l)$ -alternating ATL-formula. To get an intuition, let us consider the turn-based structure $T_{no 1\mathbf{G} \geq t}$ for $t = 2$ of Figure 6. This structure $T_{no 1\mathbf{G} \geq t}$ is analogous to the structure $T_{2t;1,2}$ except that all Agent-2 states have an additional successor: the state q^{lose} that does not satisfy any positive formula. Back to the structure of Figure 6, because Agent 2 owns the states q_3^G, q_1^G , these states do not accept any positive ATL-formula of the shape $1\mathbf{G}\phi$. Therefore, for all $q \in \{q_4^G, q_2^G\}$ and positive ATL-formulas ϕ , we have $q \models 1\mathbf{F}1\mathbf{G}2\mathbf{F}\phi$ if and only if $q \models \phi$. This actually implies that a $(1, 2, 2l)$ -alternating formula ϕ accepts

$T_{\text{no } 1\mathbf{G} \geq 2}$ if and only if the sequence of operators $1\mathbf{F}2\mathbf{F}$ (without $1\mathbf{G}$ in between) occurs at least twice in ϕ (to go from $q_4^{\mathbf{G}}$ to $q_2^{\mathbf{G}}$ and then from $q_2^{\mathbf{G}}$ to q^{win}). In fact, we consider formulas that only use $1\mathbf{G}$ operators after $1\mathbf{F}$ and before $2\mathbf{F}$, as defined below. Such formulas satisfy the lemma that follows.

► **Definition 22.** For all $H \subseteq [1, \dots, l]$, we let $\phi(l, H, 2) \in \text{ATL}^2(\{p\}, \mathbf{U}^t, \emptyset, \mathbf{B}^l, 0)$ denote the ATL-formula defined by: $\phi(l, H, 2) := 1\mathbf{F}Q_12\mathbf{F} \cdots 1\mathbf{F}Q_l2\mathbf{F}p$ where, for all $1 \leq i \leq l$, we have $Q_i \in \{\epsilon, 1\mathbf{G}\}$ and $Q_i = 1\mathbf{G}$ iff $i \notin H$.

For all $1 \leq i \leq l+1$, we let $\phi_i(l, H, 2) := 1\mathbf{F}Q_i2\mathbf{F} \cdots 1\mathbf{F}Q_l2\mathbf{F}p$ (with $\phi_{l+1}(l, H, 2) := p$).

► **Lemma 23.** A promising $(1, 2, 2l)$ -alternating formula ϕ with $|\phi| \leq 3l + 1 - k$ rejects $T_{\text{no } 1\mathbf{G} \geq k+1}$ if and only if $\phi = \phi(l, H, 2)$ for some $H \subseteq [1, \dots, l]$ such that $|H| = k$.

Proof sketch. Since ϕ is $(1, 2, 2l)$ -alternating, it uses at least l operators $1\mathbf{F}$ and $2\mathbf{F}$. Thus, it can use at most $l - k$ operators $1\mathbf{G}$. In addition, ϕ accepts $T_{\text{no } 1\mathbf{G} \geq k+1}$ iff there are at least $k + 1$ occurrences of the sequence $1\mathbf{F}2\mathbf{F}$ in ϕ . Thus, ϕ uses each $l - k$ remaining operators $1\mathbf{G}$ between a different pair of successive $1\mathbf{F}, 2\mathbf{F}$ iff it rejects $T_{\text{no } 1\mathbf{G} \geq k+1}$. ◀

With $T_p, T_{\text{no } 2\mathbf{G}}, T_{2l:1,2}$ as positive structures and $T_{2l+1:1,2}, T_{\text{no } 1\mathbf{G} \geq k+1}$ as negative structures, we have achieved both Steps a and b. Let us turn to Step c. For all $C \subseteq [1, \dots, l]$, we define a turn-based structure $T_{l,C,2}$. An example is depicted in Figure 7 for $l = 2$. The structure $T_{l,C,2}$ features a sequence of states $q_1^1, q_1^2, \dots, q_l^1, q_l^2$ alternating between Agent-1 and Agent-2 states ending in a self-looping sink q^{lose} not labeled by p . However, the Agent-1 states q_i^1 for which $i \in C$ have a “testing state” q_i^{Test} as successor. That state is self-looping, and may branch to the self-looping sink q^{lose} or to the structure $T_{2(l-i):1,2}$. That state is such that $q_i^{\text{Test}} \models Q_i2\mathbf{F}\phi_{i+1}(l, H, 2)$ iff $Q_i = \epsilon$ (iff $i \in H$). Furthermore, note that it is useless to “wait” at the state q_i^1 before branching to q_i^{Test} . Indeed, if for instance $Q_i = 1\mathbf{G}$ but $Q_{i+1} = \epsilon$, then it may seem that $q_i^{\text{Test}} \models \varphi'$, for $\varphi' := Q_{i+1}2\mathbf{F}\phi_{i+2}(l, H, 2)$ and therefore $q_i^1 \models \phi_i(l, H, 2) = 1\mathbf{F}Q_i2\mathbf{F}1\mathbf{F}\varphi'$. However, it is not the case because we do not have $q_i^{\text{Test}} \models \varphi'$, since $\phi_{i+2}(l, H, 2)$ is not $(1, 2, 2(l-i))$ -alternating, and thus it does not satisfy the structure $T_{2(l-i):1,2}$. Overall, we have the lemma below.

► **Lemma 24.** For all $C, H \subseteq [1, \dots, l]$, we have $T_{(l,C)} \models \phi(l, H, 2)$ if and only if $C \cap H = \emptyset$.

We have achieved Step c. Then, we let $\text{In}^{\text{ATL}(2)} := (\{p\}, \mathcal{P}, \mathcal{N}, B)$ be an input of the decision problem $\text{ATL}_{\text{Learn}}^2(\mathbf{U}^t, \emptyset, \mathbf{B}^l, 0)$ where $\mathcal{P} := \{T_p, T_{\text{no } 2\mathbf{G}}, T_{2l:1,2}, T_{(l,C_i,2)} \mid 1 \leq i \leq n\}$, $\mathcal{N} := \{T_{2l+1:1,2}, T_{\text{no } 1\mathbf{G} \geq k+1}\}$, and $B := 3l + 1 - k$. By Lemmas 21, 23 and 24, $\text{In}^{\text{ATL}(2)}$ is a positive instance of $\text{ATL}_{\text{Learn}}^2(\mathbf{U}^t, \emptyset, \mathbf{B}^l, 0)$ iff (l, C, k) is a positive instance of Hit.

ATL² learning with $\mathbf{U}^t = \{\mathbf{F}\}$ or $\mathbf{U}^t = \{\mathbf{G}\}$. The ATL² learning problem is now in P.

► **Theorem 25.** For all sets of operators $\mathbf{U}^t \in \{\{\mathbf{F}\}, \{\mathbf{G}\}\}$, $\mathbf{B}^l \subseteq \text{Op}_{\text{Bin}}^{\text{lg}}$, and bounds $n \in \mathbb{N}$, the decision problem $\text{ATL}_{\text{Learn}}^2(\mathbf{U}^t, \emptyset, \mathbf{B}^l, n)$ is in P.

We focus on the case $\mathbf{U}^t = \{\mathbf{F}\}$, the other is analogous. Towards Step A, consider a formula $\phi \in \text{ATL}^2(\text{Prop}, \{\mathbf{F}\}, \emptyset, \mathbf{B}^l, 0)$ and the only proposition $p \in \text{Prop}$ occurring in ϕ . By Lemma 16, we can make the following observations: 1) If the operator $1, 2\mathbf{F}$ occurs in ϕ , then $\phi \equiv 1, 2\mathbf{F}p$; 2) Otherwise, if the only operator occurring in ϕ is $\emptyset\mathbf{F}$ then $\phi \equiv \emptyset\mathbf{F}p$; 3) Otherwise, ϕ is equivalent to a formula ϕ' alternating between the operators $1\mathbf{F}$ and $2\mathbf{F}$, with $|\phi'| \leq |\phi|$. These observations suggest the definition below, which satisfies the lemma that follows.

► **Definition 26.** For a set of propositions Prop , we define the set $\text{ATL}_{\mathbf{F}}^2(\text{Prop}) := \{\text{Qt} \cdot p \mid p \in \text{Prop}, \text{Qt} \in \text{Quant}_{\text{Alt}}^{\mathbf{F}}\}$ where $\text{Quant}_{\text{Alt}}^{\mathbf{F}} := \{\epsilon, \emptyset \mathbf{F}, 1, 2 \mathbf{F}, (1 \mathbf{F} \cdot 2 \mathbf{F})^*, (1 \mathbf{F} \cdot 2 \mathbf{F})^* \cdot 1 \mathbf{F}, (2 \mathbf{F} \cdot 1 \mathbf{F})^*, (2 \mathbf{F} \cdot 1 \mathbf{F})^* \cdot 2 \mathbf{F}\}$.

► **Lemma 27.** For a set of propositions Prop , and $\phi \in \text{ATL}^2(\text{Prop}, \{\mathbf{F}\}, \emptyset, \mathbf{B}^1, 0)$, there is an ATL-formula $\phi' \in \text{ATL}_{\mathbf{F}}^2(\text{Prop})$ such that $\phi \equiv \phi'$ and $|\phi'| \leq |\phi|$.

This concludes Step A since the number of formulas of size at most B in $\text{ATL}_{\mathbf{F}}^2(\text{Prop})$ is polynomial in B and $|\text{Prop}|$. As for Step B, in this case it is trivial since checking that an ATL-formula satisfies a structure can always be done in polynomial time.

Proof of P-hardness. In Table 1, we additionally state only that ATL^2 learning with $\mathbf{U}^t \in \{\{\mathbf{F}\}, \{\mathbf{G}\}\}$ is P-hard. The proof of this fact is actually very similar to the proof that CTL learning without the operator \mathbf{X} is NL-hard, except that the reduction is made from the problem of reachability in a turn-based game (which is P-complete [31]).

ATL³ learning with $\mathbf{U}^t \in \{\{\mathbf{F}\}, \{\mathbf{G}\}\}$. Let us consider ATL learning with one more agent, i.e. ATL^3 learning, still with $\mathbf{U}^t \in \{\{\mathbf{F}\}, \{\mathbf{G}\}\}$. The turn-based structures that we consider now use the set of agents $\text{Ag} = \{1, 2, 3\}$. The goal is to show the theorem below.

► **Theorem 28.** For all sets $\mathbf{U}^t \in \{\{\mathbf{F}\}, \{\mathbf{G}\}\}$, $\mathbf{B}^1 \subseteq \text{Op}_{\text{Bin}}^{\text{lg}}$, and bound $n \in \mathbb{N}$, the decision problem $\text{ATL}_{\text{Learn}}^3(\mathbf{U}^t, \emptyset, \mathbf{B}^1, n)$ is NP-hard.

We focus on the case $\mathbf{U}^t = \{\mathbf{F}\}$ (the case $\mathbf{U}^t = \{\mathbf{G}\}$ is analogous since the operators \mathbf{F} and \mathbf{G} have a dual behavior). Once again, let us consider an instance (l, C, k) of the problem Hit. We start right away by defining the ATL^3 -formula associated to a subset $H \subseteq [1, \dots, l]$.

► **Definition 29.** For $H \subseteq [1, \dots, l]$, we let $\phi(l, H, 3)$ denote the ATL^3 -formula defined by $\phi(l, H, 3) := 1 \mathbf{F} \langle \langle A_1 \rangle \rangle \mathbf{F} \cdots 1 \mathbf{F} \langle \langle A_l \rangle \rangle \mathbf{F} p$ where, for all $1 \leq i \leq l$, we have $A_i \in \{\{2\}, \{2, 3\}\}$ and $A_i = \{2, 3\} \mathbf{F}$ if and only if $i \in H$.

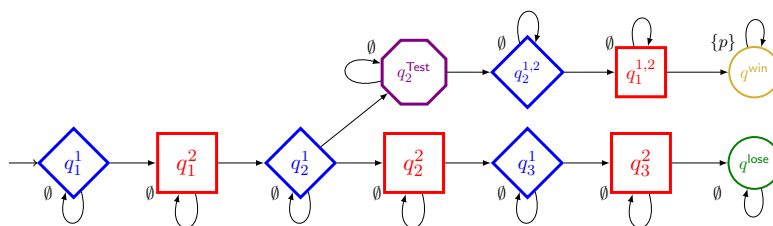
For $1 \leq i \leq l + 1$, we let $\phi_i(l, H, 3) := 1 \mathbf{F} \langle \langle A_i \rangle \rangle \mathbf{F} \cdots 1 \mathbf{F} \langle \langle A_l \rangle \rangle \mathbf{F} p$ (with $\phi_{l+1}(l, H, 3) = p$).

Toward Step a, we define $T_{2l+1:1,2}, T_{2(k+1):1,3}$ as negative structures, thus ensuring that a separating formula does not use an operator $\langle \langle A \rangle \rangle \mathbf{F}$ with $1, 2 \in A$, or $1, 3 \in A$. We also define $T_{2l:1,2}$ as a positive structure with the bound $B := 2l + 1$. That way, a separating formula is necessarily $(1, 2, 2l)$ -alternating and only uses the operators $1 \mathbf{F}, 2 \mathbf{F}$, and $2, 3 \mathbf{F}$.

► **Lemma 30.** If a formula $\phi \in \text{ATL}^3(\{\{p\}, \{\mathbf{F}\}, \emptyset, \mathbf{B}^1, n)$ with $|\phi| \leq 2l + 1$ accepts $T_{2l:1,2}$ and rejects $T_{2l+1:1,2}, T_{2(k+1):1,3}$, then there is some $H \subseteq [1, \dots, l]$ such that $\phi = \phi(l, H, 3)$.

Note that, if $|H| \geq k + 1$, then $\phi(l, H, 3)$ is $(1, 3, 2(k + 1))$ -alternating. Therefore, since $T_{2(k+1):1,3}$ is a negative structure, if $\phi(l, H, 3)$ is separating, then $|H| \leq k$, i.e. we have also achieved Step b. Let us now turn to Step c. For all $C \subseteq [1, \dots, l]$, we define the structure $T_{l,C,3}$. An example is given in Figure 8 with $l = 3$. This structure $T_{l,C,3}$ features a sequence of states $q_1^1, q_1^2, \dots, q_l^1, q_l^2$ alternating between Agent-1 and Agent-2 states ending in a self-looping sink q^{lose} . However, the Agent-1 states q_i^1 for which $i \in C$ have an Agent-3 “testing state” q_i^{test} as successor. That state is self-looping and also branches to the structure $T_{(l-i):1,2}$. Note that, given $r \geq i + 1$, the sub-formula $\phi_r(l, H, 3)$ is $(1, 2, l - r + 1)$ -alternating, and therefore satisfies the structure $T_{(l-i):1,2}$, iff $r = i + 1$. Thus, since q_i^{test} is an Agent-3 state, $q_i^{\text{test}} \models \langle \langle A_i \rangle \rangle \mathbf{F} \phi_{i+1}(l, H, 3)$ iff $3 \in A_i$ iff $i \in H$. Thus, we have the following lemma.

► **Lemma 31.** For all $C, H \subseteq [1, \dots, l]$, we have $T_{(l,C,3)} \models \phi(l, H, 3)$ if and only if $C \cap H \neq \emptyset$.



■ **Figure 8** The turn-based structure $T_{3, \{2\}, 3}$.

This concludes Step c. Overall, we let $\text{In}^{\text{ATL}^{(3), \mathbf{F}}} := (\{p\}, \mathcal{P}, \mathcal{N}, B)$ be an input of the decision problem $\text{ATL}_{\text{Learn}}^3(\{\mathbf{F}\}, \emptyset, \mathbf{B}^1, 0)$ where $\mathcal{P} := \{T_{2l:1,2}, T_{(l, C_i, 3)} \mid 1 \leq i \leq n\}$, $\mathcal{N} := \{T_{2l+1:1,2}, T_{2(k+1):1,3}\}$, and $B := 2l + 1$. We have that $\text{In}^{\text{ATL}^{(3), \mathbf{F}}}$ is a positive instance of $\text{ATL}_{\text{Learn}}^3(\{\mathbf{F}\}, \emptyset, \mathbf{B}^1, 0)$ if and only if (l, C, k) is a positive instance of Hit.

5 Future Work

Within our setting, we have covered many cases, as can be seen in Table 1. That is why the complete version of this work [8] is already quite long. However, there are still some cases that we have not tackled. First, there is the case of ATL^2 learning with $\mathbf{U}^t \in \{\{\mathbf{F}, \neg\}, \{\mathbf{G}, \neg\}\}$. We believe that it behaves like the case $\mathbf{F}, \mathbf{G} \in \mathbf{U}^t$, but the proofs would entail many additional technical details, since replacing \mathbf{F} with $\neg \mathbf{G} \neg$ increases the size of the formulas.

More importantly, when considering a bounded amount of binary operators, we have not allowed binary temporal operators ($\mathbf{U}, \mathbf{R}, \mathbf{W}, \mathbf{M}$). Doing so would enhance the expressivity of the fragment that we consider, and we conjecture that we would obtain the same result as in this paper, with proofs that should be only moderately more involved.

On a more high level perspective, in this paper we have focused solely on solving exactly the learning problems and although we have found some relevant tractable cases, many are untractable. A promising research direction would be to look for tractable approximation algorithms, similarly to what is done in [27].

References

- 1 Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, September 2002. doi:10.1145/585265.585270.
- 2 Dana Angluin. On the complexity of minimum inference of regular sets. *Inf. Control.*, 39(3):337–350, 1978. doi:10.1016/S0019-9958(78)90683-6.
- 3 M. Fareed Arif, Daniel Larraz, Mitziu Echeverria, Andrew Reynolds, Omar Chowdhury, and Cesare Tinelli. SYSLITE: syntax-guided synthesis of PLTL formulas from finite traces. In *FMCAD*, pages 93–103. IEEE, 2020. doi:10.34727/2020/ISBN.978-3-85448-042-6_16.
- 4 Ayca Balkan, Moshe Y. Vardi, and Paulo Tabuada. Mode-target games: Reactive synthesis for control applications. *IEEE Trans. Autom. Control.*, 63(1):196–202, 2018. doi:10.1109/TAC.2017.2722960.
- 5 Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Trans. Inf. Theory*, 24(3):384–386, 1978. doi:10.1109/TIT.1978.1055873.
- 6 Dines Bjørner and Klaus Havelund. 40 years of formal methods - some obstacles and some possibilities? In *FM*, volume 8442 of *Lecture Notes in Computer Science*, pages 42–61. Springer, 2014. doi:10.1007/978-3-319-06410-9_4.

- 7 Giuseppe Bombara, Cristian-Ioan Vasile, Francisco Penedo, Hirotohi Yasuoka, and Calin Belta. A decision tree approach to data classification using signal temporal logic. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control, HSCC '16*, pages 1–10, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2883817.2883843.
- 8 Benjamin Bordais, Daniel Neider, and Rajarshi Roy. The complexity of learning temporal properties. *CoRR*, abs/2408.04486, 2024. doi:10.48550/arXiv.2408.04486.
- 9 Benjamin Bordais, Daniel Neider, and Rajarshi Roy. Learning branching-time properties in CTL and ATL via constraint solving. In André Platzer, Kristin Yvonne Rozier, Matteo Pradella, and Matteo Rossi, editors, *Formal Methods - 26th International Symposium, FM 2024, Milan, Italy, September 9-13, 2024, Proceedings, Part I*, volume 14933 of *Lecture Notes in Computer Science*, pages 304–323. Springer, 2024. doi:10.1007/978-3-031-71162-6_16.
- 10 Alberto Camacho, Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Anthony Valenzano, and Sheila A. McIlraith. LTL and beyond: Formal languages for reward function specification in reinforcement learning. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 6065–6073. ijcai.org, 2019. doi:10.24963/IJCAI.2019/840.
- 11 Alberto Camacho and Sheila A. McIlraith. Learning interpretable models expressed in linear temporal logic. In *ICAPS*, pages 621–630. AAAI Press, 2019. URL: <https://ojs.aaai.org/index.php/ICAPS/article/view/3529>.
- 12 Alberto Camacho, Eleni Triantafillou, Christian J. Muise, Jorge A. Baier, and Sheila A. McIlraith. Non-deterministic planning with temporally extended goals: LTL over finite and infinite traces. In *AAAI*, pages 3716–3724. AAAI Press, 2017. doi:10.1609/AAAI.V31I1.11058.
- 13 Alessio Cecconi, Giuseppe De Giacomo, Claudio Di Ciccio, Fabrizio Maria Maggi, and Jan Mendling. Measuring the interestingness of temporal logic behavioral specifications in process mining. *Inf. Syst.*, 107:101920, 2022. doi:10.1016/J.IS.2021.101920.
- 14 William Chan. Temporal-logic queries. In *CAV*, volume 1855 of *Lecture Notes in Computer Science*, pages 450–463. Springer, 2000.
- 15 Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In Dexter Kozen, editor, *Logics of Programs, Workshop, Yorktown Heights, New York, USA, May 1981*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer, 1981. doi:10.1007/BFB0025774.
- 16 Rüdiger Ehlers, Ivan Gavran, and Daniel Neider. Learning properties in LTL \cap ACTL from positive examples only. In *2020 Formal Methods in Computer Aided Design, FMCAD 2020, Haifa, Israel, September 21-24, 2020*, pages 104–112. IEEE, 2020. doi:10.34727/2020/ISBN.978-3-85448-042-6_17.
- 17 Georgios E. Fainekos, Hadas Kress-Gazit, and George J. Pappas. Temporal logic motion planning for mobile robots. In *ICRA*, pages 2020–2025. IEEE, 2005. doi:10.1109/ROBOT.2005.1570410.
- 18 Nathanaël Fijalkow and Guillaume Lagarde. The complexity of learning linear temporal formulas from examples. In Jane Chandlee, Rémi Eyraud, Jeff Heinz, Adam Jardine, and Menno van Zaanen, editors, *Proceedings of the 15th International Conference on Grammatical Inference, 23-27 August 2021, Virtual Event*, volume 153 of *Proceedings of Machine Learning Research*, pages 237–250. PMLR, 2021. URL: <https://proceedings.mlr.press/v153/fijalkow21a.html>.
- 19 Marie Fortin, Boris Konev, Vladislav Ryzhikov, Yury Savateev, Frank Wolter, and Michael Zakharyashev. Reverse engineering of temporal queries mediated by LTL ontologies. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 3230–3238. ijcai.org, 2023. doi:10.24963/IJCAI.2023/360.

- 20 Jean-Raphaël Gaglione, Daniel Neider, Rajarshi Roy, Ufuk Topcu, and Zhe Xu. Maxsat-based temporal logic inference from noisy data. *Innov. Syst. Softw. Eng.*, 18(3):427–442, 2022. doi:10.1007/S11334-022-00444-8.
- 21 E. Mark Gold. Complexity of automaton identification from given data. *Inf. Control.*, 37(3):302–320, 1978. doi:10.1016/S0019-9958(78)90562-4.
- 22 Antonio Ielo, Mark Law, Valeria Fionda, Francesco Ricca, Giuseppe De Giacomo, and Alessandra Russo. Towards ilp-based ltlf passive learning. In *Inductive Logic Programming: 32nd International Conference, ILP 2023, Bari, Italy, November 13–15, 2023, Proceedings*, pages 30–45, Berlin, Heidelberg, 2023. Springer-Verlag. doi:10.1007/978-3-031-49299-0_3.
- 23 Neil Immerman. Number of quantifiers is better than number of tape cells. *J. Comput. Syst. Sci.*, 22(3):384–406, 1981. doi:10.1016/0022-0000(81)90039-8.
- 24 Jean Christoph Jung, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyashev. Extremal separation problems for temporal instance queries. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024*, pages 3448–3456. ijcai.org, 2024. URL: <https://www.ijcai.org/proceedings/2024/382>.
- 25 Xiao Li, Cristian Ioan Vasile, and Calin Belta. Reinforcement learning with temporal logic rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, pages 3834–3839. IEEE, 2017. doi:10.1109/IROS.2017.8206234.
- 26 Weilin Luo, Pingjia Liang, Jianfeng Du, Hai Wan, Bo Peng, and Delong Zhang. Bridging ltlf inference to GNN inference for learning ltlf formulae. In *AAAI*, pages 9849–9857. AAAI Press, 2022. doi:10.1609/AAAI.V36I9.21221.
- 27 Corto Mascle, Nathanaël Fijalkow, and Guillaume Lagarde. Learning temporal formulas from examples is hard. *CoRR*, abs/2312.16336, 2023. doi:10.48550/arXiv.2312.16336.
- 28 Sara Mohammadinejad, Jyotirmoy V. Deshmukh, Aniruddh Gopinath Puranic, Marcell Vazquez-Chanlatte, and Alexandre Donzé. Interpretable classification of time-series data using efficient enumerative techniques. In *HSCC '20: 23rd ACM International Conference on Hybrid Systems: Computation and Control, Sydney, New South Wales, Australia, April 21-24, 2020*, pages 9:1–9:10. ACM, 2020. doi:10.1145/3365365.3382218.
- 29 Daniel Neider and Ivan Gavran. Learning linear temporal properties. In Nikolaj S. Bjørner and Arie Gurfinkel, editors, *2018 Formal Methods in Computer Aided Design, FMCAD 2018, Austin, TX, USA, October 30 - November 2, 2018*, pages 1–10. IEEE, 2018. doi:10.23919/FMCAD.2018.8603016.
- 30 Daniel Neider and Rajarshi Roy. *What Is Formal Verification Without Specifications? A Survey on Mining LTL Specifications*, pages 109–125. Springer Nature Switzerland, Cham, 2025. doi:10.1007/978-3-031-75778-5_6.
- 31 C.H. Papadimitriou. *Computational Complexity*. Theoretical computer science. Addison-Wesley, 1994. URL: <https://books.google.de/books?id=JogZAQAIAAJ>.
- 32 Nir Piterman, Amir Pnueli, and Yaniv Sa’ar. Synthesis of reactive(1) designs. In E. Allen Emerson and Kedar S. Namjoshi, editors, *Verification, Model Checking, and Abstract Interpretation, 7th International Conference, VMCAI 2006, Charleston, SC, USA, January 8-10, 2006, Proceedings*, volume 3855 of *Lecture Notes in Computer Science*, pages 364–380. Springer, 2006. doi:10.1007/11609773_24.
- 33 Amir Pnueli. The temporal logic of programs. In *Proc. 18th Annu. Symp. Found. Computer Sci.*, pages 46–57, 1977. doi:10.1109/SFCS.1977.32.
- 34 Adrien Pommellet, Daniel Stan, and Simon Scatton. Sat-based learning of computation tree logic. In Christoph Benzmüller, Marijn J. H. Heule, and Renate A. Schmidt, editors, *Automated Reasoning - 12th International Joint Conference, IJCAR 2024, Nancy, France, July 3-6, 2024, Proceedings, Part I*, volume 14739 of *Lecture Notes in Computer Science*, pages 366–385. Springer, 2024. doi:10.1007/978-3-031-63498-7_22.

- 35 Ritam Raha, Rajarshi Roy, Nathanaël Fijalkow, and Daniel Neider. Scalable anytime algorithms for learning fragments of linear temporal logic. In Dana Fisman and Grigore Rosu, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 263–280, Cham, 2022. Springer International Publishing. doi:10.1007/978-3-030-99524-9_14.
- 36 Ritam Raha, Rajarshi Roy, Nathanaël Fijalkow, Daniel Neider, and Guillermo A. Pérez. Synthesizing efficiently monitorable formulas in metric temporal logic. In *VMCAI (2)*, volume 14500 of *Lecture Notes in Computer Science*, pages 264–288. Springer, 2024. doi:10.1007/978-3-031-50521-8_13.
- 37 Heinz Riener. Exact synthesis of LTL properties from traces. In *FDL*, pages 1–6. IEEE, 2019. doi:10.1109/FDL.2019.8876900.
- 38 Rajarshi Roy, Dana Fisman, and Daniel Neider. Learning interpretable models in the property specification language. In *IJCAI*, pages 2213–2219. ijcai.org, 2020. doi:10.24963/IJCAI.2020/306.
- 39 Kristin Yvonne Rozier. Specification: The biggest bottleneck in formal methods and autonomy. In *VSTTE*, volume 9971 of *Lecture Notes in Computer Science*, pages 8–26, 2016. doi:10.1007/978-3-319-48869-1_2.
- 40 Dorsa Sadigh, Eric S. Kim, Samuel Coogan, S. Shankar Sastry, and Sanjit A. Seshia. A learning based approach to control synthesis of markov decision processes for linear temporal logic specifications. In *53rd IEEE Conference on Decision and Control, CDC 2014, Los Angeles, CA, USA, December 15-17, 2014*, pages 1091–1096. IEEE, 2014. doi:10.1109/CDC.2014.7039527.
- 41 Mojtaba Valizadeh, Nathanaël Fijalkow, and Martin Berger. LTL learning on gpus. In Arie Gurfinkel and Vijay Ganesh, editors, *Computer Aided Verification - 36th International Conference, CAV 2024, Montreal, QC, Canada, July 24-27, 2024, Proceedings, Part III*, volume 14683 of *Lecture Notes in Computer Science*, pages 209–231. Springer, 2024. doi:10.1007/978-3-031-65633-0_10.
- 42 Hai Wan, Pingjia Liang, Jianfeng Du, Weilin Luo, Rongzhen Ye, and Bo Peng. End-to-end learning of ltlf formulae by faithful ltlf encoding. In *AAAI*, pages 9071–9079. AAAI Press, 2024. doi:10.1609/AAAI.V38I8.28757.
- 43 Andrzej Wasylkowski and Andreas Zeller. Mining temporal specifications from object usage. *Autom. Softw. Eng.*, 18(3-4):263–292, 2011. doi:10.1007/S10515-011-0084-1.