Deciding the Common Fragment of CTL with Past and LTL

Massimo Benerecetti
Università degli Studi di Napoli "Federico II", Italy

Dario Della Monica
Università degli Studi di Udine, Italy

Angelo Matteo
Università degli Studi di Udine, Italy

Fabio Mogavero
Università degli Studi di Napoli "Federico II", Italy

Gabriele Puppis
Università degli Studi di Udine, Italy

Abstract

A central goal of language theory is to compare formalisms by understanding both their expressive overlaps and their relative expressive power. One particularly challenging question in this direction is the problem of determining the *common fragment* of two formalisms F_1 and F_2, that is, effectively characterise the class $F_1 \cap F_2$ of properties that can be expressed in both formalisms. This question can be equally phrased as a decision problem: given a property expressed in F_1 or F_2, decide whether the same property can be also expressed in $F_1 \cap F_2$. A question closely related to this is the *membership problem*, denoted $F_1 \mapsto F_2$, which asks whether a property expressed in F_1 can be also expressed in F_2. These problems become particularly difficult when *branching-time* formalisms are involved, in general due to the lack of equivalent algebraic characterizations.

In this work, we prove that LTL$\cap$PCTL is decidable, where PCTL denotes CTL extended with *past operators*. We do this by showing that both membership problems, LTL $\mapsto$ PCTL and PCTL $\mapsto$ LTL, are decidable. The direction PCTL $\mapsto$ LTL follows from suitable combinations of known results. The converse direction, LTL $\mapsto$ PCTL, requires an automata-theoretic characterisation of PCTL. Specifically, we introduce a new class of automata, called *counter-free hesitant weak tree automata* (HWT_{cf}) that capture precisely the expressiveness of PCTL, and that are obtained by combining two orthogonal restrictions on alternating parity tree automata, namely, *counter-free hesitancy* and *weakness*. We then prove that, for every word language L defined by an LTL formula, the associated tree language $\triangle[L]$ is recognisable by an HWT_{cf} if and only if L is recognized by a deterministic Büchi word automaton. Since the latter recognisability problem is known to be decidable, so is the former.

This result advances the longstanding open problem of deciding LTL $\cap$ CTL. Indeed, that problem can now be reduced to PCTL $\mapsto$ CTL, that is, the question of when past operators can be eliminated.

2012 ACM Subject Classification Theory of computation → Tree languages; Theory of computation → Modal and temporal logics; Theory of computation → Logic and verification; Theory of computation → Automata over infinite objects

Keywords and phrases Membership problems, tree languages, tree logics, tree automata, Monadic Path Logic, CTL*, CTL with past

Digital Object Identifier 10.4230/LIPIcs.MFCS.2026.24

Related Version *Full Version*: https://arxiv.org/abs/2606.30405

Funding This work was partially supported by INdAM - GNCS Project (CUP: E53C25002010001).

1 Introduction

The notion of a *common fragment* is central to expressiveness theory: given two formalisms F_1 and F_2, the aim is to effectively characterise the class $F_1 \cap F_2$ of properties that are definable in both. Closely related is the associated *membership problem* $F_1 \mapsto F_2$: given a specification in F_1, determine whether it can be transformed into an equivalent specification in F_2. Of course, deciding the common fragment $F_1 \cap F_2$ reduces to solving two membership problems, i.e. $F_1 \mapsto F_2$ and $F_2 \mapsto F_1$.

These questions have a long history in language theory, tracing back at least to Schützenberger's theorem on *star-free word languages* and *aperiodic monoids* [45] and to Kamp's theorem connecting Prior's *tense logic* [41] with *first-order logic* over *linearly ordered structures* [20]. From these seminal works, a rich line of research has emerged, yielding a remarkably complete picture for languages of finite and infinite words, where predicate logics [8, 9, 10, 27], temporal logics [39, 40, 49], automata [29, 30, 12, 25], and algebraic structures [36] are tied together by deep and elegant correspondences [37].

By contrast, for regular tree languages the situation is considerably more difficult and far more fragmented. A milestone is Rabin's work on the monadic second-order theory of infinite trees, which established a decisive connection between *Monadic Second-Order Logic* (MSO) and tree automata [44]. This connection made it possible to study logical questions about decidability and definability by means of automata. Within this framework, full MSO corresponds to automata with parity conditions, whereas weaker logics correspond to weaker acceptance mechanisms. In particular, *Weak* MSO (WMSO), where second-order quantification ranges only over finite sets, admits an automaton-based characterisation in terms of *alternating weak tree automata* (AWT) [35], a subclass of alternating parity tree automata where each strongly connected component of the transition graph has a single priority. Thanks to this characterisation, the membership problem MSO $\mapsto$ WMSO can be viewed automata-theoretically as the problem of deciding when a parity tree automaton can be replaced by an equivalent weak one. The same parity-versus-weakness boundary also appears with bisimulation-invariant properies, where it governs the comparison between the *modal μ-calculus* (L_μ) [21] and its *alternation-free fragment* (AFL_μ). More precisely, thanks to the celebrated characterisation theorem of Janin and Walukiewicz [19], L_μ captures precisely the bisimulation-invariant fragment of MSO, and admits a standard presentation in terms of alternating parity tree automata [48]. Its alternation-free fragment, in turn, corresponds to weak alternating automata, as well as to the bisimulation-invariant fragment of WMSO. Thus, the question whether a L_μ property is definable in AFL_μ can again be reduced to a membership problem between classes of automata, namely, APT $\mapsto$ AWT; in other words, deciding when parity can be replaced by weakness. This viewpoint has become central in verification: many logical formalisms can be analysed via their corresponding automata models, and suitable restrictions can often be identified, both at the logic level and at the automaton level, that support a number of effective verification procedures [22].

Within this landscape, the two formalisms most relevant to us are AFL_μ and CTL^*. Over infinite binary trees, these formalisms correspond to natural fragments of MSO. Indeed, we have already seen that AFL_μ corresponds to WMSO (note that the bisimulation relation is trivial over binary trees). As for CTL^*, this corresponds to the *path-quantification fragment* of MSO (MPL) [17, 31, 32]. Moreover, recent automata-theoretic characterisations of CTL^* have made this picture even sharper, by identifying *counter-free hesitancy* as the structural constraint on alternating parity tree automata that achieves the same expressive power as CTL^* [3]. It is therefore natural to ask for a formalism L that is as expressive as

WMSO $\cap$ MPL, or, equally, as $\mathrm{AFL}_\mu \cap \mathrm{CTL}^*$. At the same time, this question appears to be surprisingly open. Identifying such a logic would have immediate consequences. First, analogous to Rabin's results for MSO $\mapsto$ WMSO [42], this approach would yield a reduction from the membership problem MPL $\mapsto$ WMSO $\cap$ MPL to the problem APT $\mapsto$ AWT. Second, and more importantly for us, it would provide a new perspective on the closely related and longstanding open problem of determining the common fragment LTL$\cap$CTL [16], where LTL here is interpreted over trees using an implicit universal path-quantification. In other words, the problem of characterising LTL $\cap$ CTL amounts to understanding the overlap between a linear-time and a branching-time formalism.

The history of the common fragment LTL $\cap$ CTL spans almost four decades. The first major result is the decidability of $\mathrm{CTL}^* \mapsto \mathrm{LTL}$, due to Clarke and Draghicescu, which also settles CTL $\mapsto$ LTL [13]. A possible route towards $\mathrm{CTL}^* \mapsto \mathrm{CTL}$, conjectured in the same work, was later ruled out by Boker and Shaulian [7]. The inverse membership problem, i.e., LTL $\mapsto$ CTL, proved substantially more difficult. Maidl gave a necessary and sufficient condition for LTL $\mapsto$ ACTL [28], where ACTL is the syntactic fragment of CTL in which the existential path quantifier E is disallowed, and Bojańczyk later proved this problem decidable, while also showing the strict inclusion $\mathrm{LTL} \cap \mathrm{ACTL} \subsetneq \mathrm{LTL} \cap \mathrm{CTL}$. This is conceptually striking: although LTL expresses only universal properties, existential path quantification is still needed on the CTL side to capture the full common fragment [5]. A further key development is the result of Kupferman and Vardi showing that $\mathrm{LTL} \mapsto \mathrm{AFL}_\mu$ is decidable, with an ExpSpace upper bound and a PSpace lower bound [23].

The present paper develops this automata-theoretic perspective. We introduce a new class of automata, called *counter-free hesitant weak tree automata* (HWT_{cf}). The definition combines two orthogonal restrictions: *counter-free hesitancy*, which comes from the automata model for CTL^*, and *weakness*, which underlies the automata model for AFL_μ. This combination makes HWT_{cf} a natural candidate for capturing the common fragment $\mathrm{CTL}^* \cap \mathrm{AFL}_\mu$, and it leads to the following conjecture:

▶ **Conjecture 1.** HWT_{cf} *are as expressive as* $\mathrm{CTL}^* \cap \mathrm{AFL}_\mu$*, and, equivalently, as* MPL $\cap$ WMSO*.*

This conjecture provides the conceptual starting point of the paper. Even without resolving it in full, the automata class HWT_{cf} already reveals substantial structural information about the target intersection and leads to a sequence of results that we outline below.

Our first main result is a logical characterisation of HWT_{cf} in terms of PCTL, namely, CTL extended with *past operators*. This identifies PCTL as a natural temporal-logic candidate for the common fragment under investigation. The characterisation is supported by a normal form for PCTL and by the identification of an equivalent fragment of CTL^* in which path formulae are restricted to define safety and co-safety properties. The correspondence also preserves the polarity of path quantification. More precisely, HWT_{cf} that only use universal non-determinism (we call them *universal* HWT_{cf} for short) translate into APCTL, i.e. the fragment of PCTL that uses only universal path quantification. Symmetrically, *existential* HWT_{cf} translate into EPCTL, i.e. the fragment of PCTL that uses only existential path quantification.

▶ **Theorem 2.** HWT_{cf} *and* PCTL *are effectively equivalent formalisms. Moreover, universal* HWT_{cf} *(resp., existential* HWT_{cf}*) admits translations into* APCTL *(resp.,* EPCTL*).*

A first consequence of the above characterisation is conditional on the conjectured equivalence between HWT_{cf} and $\mathrm{CTL}^* \cap \mathrm{AFL}_\mu$. Indeed, under the assumption that

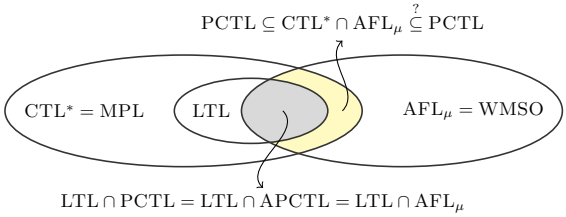


Figure 1 Main relationships between classes of tree languages; new results concern shaded regions.

$\mathrm{HWT}_{cf} = \mathrm{CTL}^* \cap \mathrm{AFL}_\mu$, the above characterization would make PCTL the natural logical presentation for the common fragment $\mathrm{CTL}^* \cap \mathrm{AFL}_\mu$. Moreover, the membership problem $\mathrm{CTL}^* \mapsto \mathrm{PCTL}$ would admit a Rabin-style automata-theoretic reduction: indeed, the problem would become equivalent to asking whether a CTL^*-definable language is also definable in AFL_μ ($= \mathrm{CTL}^* \cap \mathrm{AFL}_\mu$), which in turn is reduced to asking whether a given parity tree automaton (capturing a specification in CTL^*) is equivalent to some weak alternating tree automaton (capturing a specification in AFL_μ). This gives the following conditional result:

▶ **Theorem 3.** *Assuming Conjecture 1, the membership problem* $\mathrm{CTL}^* \mapsto \mathrm{PCTL}$ *(or, equally,* $\mathrm{MPL} \mapsto \mathrm{PCTL}$*) reduces to the automata-theoretic problem* $\mathrm{APT} \mapsto \mathrm{AWT}$*.*

More importantly, the characterisation of HWT_{cf} in terms of PCTL yields unconditional consequences for the linear/branching-time interface. Specifically, it enables an exact identification of the intersection of LTL with several branching-time formalisms: PCTL, APCTL, and AFL_μ. Indeed, all these common fragments coincide. This is particularly noteworthy because, unlike the situation for CTL, the presence of past operators makes existential path quantification unnecessary: in this respect, PCTL behaves more like CTL^* than like CTL.

▶ **Theorem 4.** $\mathrm{LTL} \cap \mathrm{APCTL} = \mathrm{LTL} \cap \mathrm{PCTL} = \mathrm{LTL} \cap \mathrm{AFL}_\mu$.

This equivalence can in turn be converted into decidability of membership problems beetween LTL and PCTL. More precisely, for an LTL-definable word language L, we prove that L is recognisable by a deterministic Büchi automaton (DBW) if and only if the associated tree language $\triangle[L]$ is recognisable by an HWT_{cf} automaton, equivalently definable in PCTL. Combining this correspondence with the known decidability of DBW-recognisability for LTL [23] yields decidability of $\mathrm{LTL} \mapsto \mathrm{PCTL}$, with an ExpSpace upper bound and a PSpace lower bound. The converse problem, namely $\mathrm{PCTL} \mapsto \mathrm{LTL}$, is also decidable, by means of the equivalent CTL^* fragment identified above and the corresponding characterisation of safety and co-safety path languages.

▶ **Theorem 5.** *The membership problem* $\mathrm{LTL} \mapsto \mathrm{PCTL}$ *is decidable in* ExpSpace *and is* PSpace*-hard. Moreover,* $\mathrm{PCTL} \mapsto \mathrm{LTL}$ *is decidable.*

These results have a direct bearing on the longstanding open problem of deciding $\mathrm{LTL} \mapsto \mathrm{CTL}$. Indeed, once the common fragment of LTL with PCTL has been identified and its membership problem shown decidable, the original question can be reformulated in terms of *past elimination*. More precisely, deciding whether an LTL formula is equivalent to some CTL formula reduces to deciding whether an equivalent PCTL formula exists and can be translated into CTL. In this way, the classical problem of understanding $\mathrm{LTL} \cap \mathrm{CTL}$ is reduced to the branching-time problem $\mathrm{PCTL} \mapsto \mathrm{CTL}$, thereby opening an alternative route towards its solution through the study of elimination of past operators.

▶ **Corollary 6.** *Decidability of* PCTL $\mapsto$ CTL *implies decidability of* LTL $\mapsto$ CTL.

Finally, we investigate whether PCTL admits a matching characterisation in terms of known monadic predicate logics. Here the answer is negative. On the one hand, WMPL, which coincides with FO, is known to be strictly less expressive than PCTL. On the other hand, the recently introduced tree fragment of WMSO, namely WMTL [2], turns out to be strictly more expressive, and in fact incomparable with CTL*. This shows that the currently known fragments of WMSO do not capture the expressive power of PCTL, and therefore do not provide a purely monadic characterisation of the target intersection. Conversely, looking beyond the expressive power of WMSO is unnecessary; the inclusion PCTL $\subseteq$ WMSO $\cap$ MPL immediately implies that PCTL is already a fragment of WMSO.

▶ **Theorem 7.** WMTL *is incomparable to both* CTL* *and* MPL. *Consequently, no known fragment of* WMSO *is expressively equivalent to* PCTL.

2 Preliminaries

Trees. We mainly work with trees, precisely with unordered, full binary trees. Formally, a tree is a pair $t = (Dom(t), E)$, where $Dom(t)$ is the set of nodes and $E \subseteq Dom(t) \times Dom(t)$ is the directed edge relation, such that:

1. there is a unique node r, called the root of t, from which all nodes can be reached, i.e., $(r, u) \in E^*$ for all $u \in Dom(t)$;
2. for every $u \neq r$, there is a unique v, called the parent of u, such that $(v, u) \in E$;
3. the root has no parent, that is, there is no $v \in Dom(t)$ with $(v, r) \in E$;
4. for every node u, there are exactly two nodes v_1, v_2 such that $(u, v_1), (u, v_2) \in E$; these are the children of u (note that there is no particular order imposed on v_1, v_2).

Unless otherwise stated, all trees are infinite and without leaves. We use binary trees rather than unranked trees, where nodes may have an arbitrary finite number of children, in order to keep a direct correspondence between temporal and monadic logics. As a matter of fact, on unranked trees, temporal logics correspond only to bisimulation-invariant fragments of monadic logics, unless extended with counting operators (see, e.g., [32]). Restricting the branching degree allows us to bypass these technicalities, though it is understood that our results readily extend to the unranked case.

A Σ-*tree* is a tree t expanded with a labelling $\lambda : Dom(t) \to \Sigma$, associating a letter from the finite alphabet Σ with each node. An ω-*tree language* is a set of Σ-trees. We denote by ϵ_t the root of a tree (or a Σ-tree) t. A *path* of t is a finite or infinite sequence $\pi = v_1 v_2 \dots$ of nodes that are two-by-two connected by the child relation, i.e., $(v_i, v_{i+1}) \in E$ for all $i \geq 1$. A *maximal path* is a path that cannot be extended further to the right, so it must be infinite unless the tree has some leaves. Note that path and maximal paths may start at arbitrary nodes of trees. A *subtree* of t is a finite or infinite set $D \subseteq Dom(t)$ such that E restricted to D defines a tree, not necessarily full or leafless.

Monadic logics. We introduce *monadic second-order logic* (MSO) on trees. Fix a finite set AP of atomic propositions, and let $\Sigma = 2^{AP}$. MSO formulae are defined by the grammar

$$\varphi \ ::= \ x = x' \ \mid \ x \leq x' \ \mid \ x \in X \ \mid \ p(x) \ \mid \ \neg\varphi \ \mid \ \varphi \vee \varphi \ \mid \ \exists x.\, \varphi \ \mid \ \exists X.\, \varphi$$

where $p \in AP$ and x, x' (resp., X) are first-order (resp., monadic) variables. First-order variables are interpreted as single nodes of a tree, while monadic variables are interpreted as sets of nodes. The atom $x \leq x'$ states that the node assigned to x' is a descendant of the

node assigned to x; the atom $x \in X$ states that the node assigned to x belongs to the set assigned to X; and $p(x)$ states that the node assigned to x carries the proposition p, that is, p belongs to its label. The rest of the semantics is standard.

We consider the usual notion of MSO *sentence*. Since an MSO sentence has no free variables, its truth value on a Σ-tree is independent of any variable assignment. We write $t \models \varphi$ when t is a Σ-tree that satisfies the sentence φ. An ω-tree language T is MSO-*definable* if there is a sentence φ such that $T = \{t \mid t \models \varphi\}$. Due to the correspondence between MSO and several classes of automata on Σ-trees and due to the closure properties satisfied by these classes [43, 33, 34, 47], MSO-definable ω-tree languages are also often called *regular*.

Finally, we introduce the following fragments of MSO:

- *Weak monadic second-order logic* (WMSO) is the fragment in which monadic variables can only be assigned *finite sets* of nodes of a tree.
- *Monadic tree logic* (MTL), resp., *weak* MTL (WMTL) are the fragments in which monadic variables can only be assigned *subtrees*, resp., *finite subtrees*.
- *Monadic path logic* (MPL), resp., *Weak* MPL (WMPL) are the fragments in which monadic variables can only be assigned *paths*, resp., *finite paths*.
- *First order logic* (FO) is the fragment in which monadic variables are disallowed.

Temporal Logics. We consider here *full computation tree logic with past*, denoted PCTL*. Its formulae are generated in negation normal form according to the following grammar:

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{E}\psi \mid \mathsf{A}\psi$$
$$\psi ::= \varphi \mid \psi \vee \psi \mid \psi \wedge \psi \mid \mathsf{X}\psi \mid \mathsf{Y}\psi \mid \tilde{\mathsf{Y}}\psi \mid \psi \mathsf{U} \psi \mid \psi \mathsf{R} \psi \mid \psi \mathsf{S} \psi$$

Formulae generated from φ are called *state formulae*, while those generated from ψ are called *path formulae*. The operators E and A quantify existentially and universally over paths. The operators X, U, and R are future temporal operators: X is the usual "next" operator, and U and R are "until" and "release". Dually, Y, $\tilde{\mathsf{Y}}$, and S are past temporal operators: Y is "yesterday", $\tilde{\mathsf{Y}}$ is "weak yesterday", and S is "since". The difference between Y and $\tilde{\mathsf{Y}}$ is that Y requires the existence of the predecessor (i.e., parent) along the current path, whereas $\tilde{\mathsf{Y}}$ is also satisfied at the beginning of the path. Unless explicitly stated otherwise, by a "PCTL* formula" we mean a *state* PCTL* formula.

State formulae are interpreted over *pointed trees*, i.e., pairs (t, u) consisting of a Σ-tree and a node u in it, where $\Sigma = 2^{AP}$. Similarly, path formulae are interpreted over *path-pointed trees*, i.e., triples (t, π, i) consisting of a Σ-tree, an infinite path π in it, and a position $i \geq 0$ on that path. We give the semantics only for the most interesting cases:

- $(t, u) \models \mathsf{E}\psi$ iff there exists an infinite path π starting at u such that $(t, \pi, 0) \models \psi$,
- $(t, \pi, i) \models \psi_1 \mathsf{U} \psi_2$ iff $\exists j \geq i$ such that $(t, \pi, j) \models \psi_2$ and $\forall k.(i \leq k < j), (t, \pi, k) \models \psi_1$,
- $(t, \pi, i) \models \psi_1 \mathsf{S} \psi_2$ iff $\exists j \leq i$ such that $(t, \pi, j) \models \psi_2$ and $\forall k.(i \geq k > j), (t, \pi, k) \models \psi_1$,

We write $t \models \varphi$ to mean $(t, \epsilon_t) \models \varphi$ — in this case, we also say that t is a *model* of φ. We denote by $\mathscr{L}(\varphi)$ the set of all models of φ, and declare two formulae φ, φ' *equivalent* if $\mathscr{L}(\varphi) = \mathscr{L}(\varphi')$. We will also consider the following fragments of PCTL*: CTL*, which is just PCTL* without past operators; PCTL, i.e., the variant of PCTL* in which every future temporal operator (i.e., X, U and R) must be immediately preceded by a path quantifier; CTL, i.e., PCTL without past operators. By [17, Theorem 3.6], we have that MPL, PCTL* and CTL* are equivalent formalisms. We spell out the syntax of PCTL formulas in negation

normal form, since this is the formalism that is mostly used throughout the paper:

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{EX}\psi \mid \mathsf{E}\psi\mathsf{U}\psi \mid \mathsf{E}\psi\mathsf{R}\psi \mid \mathsf{AX}\psi \mid \mathsf{A}\psi\mathsf{U}\psi \mid \mathsf{A}\psi\mathsf{R}\psi$$
$$\psi ::= \varphi \mid \psi \vee \psi \mid \psi \wedge \psi \mid \mathsf{Y}\psi \mid \tilde{\mathsf{Y}}\psi \mid \psi\mathsf{S}\psi.$$

We also consider *linear-time formalisms*. In particular, we consider the well-known *linear temporal logic with past* (PLTL), here defined as the set of path formulae of PCTL* in which we forbid the use of path quantifiers, and some of its fragments: LTL is PLTL without past operators; coSafeLTL (resp., SafeLTL) is PLTL in which past operators and the future operator R (resp., U) are disallowed; coSafePLTL (resp., SafePLTL) consists of PLTL formulae of the form $\mathsf{F}\alpha$ (resp., $\mathsf{G}\alpha$), where $\mathsf{F}\alpha = \top\mathsf{U}\alpha$ (resp., $\mathsf{G}\alpha = \bot\mathsf{R}\alpha$) and α contains only past operators.

▶ **Theorem 8** ([11, Theorems 1 and 8]). SafePLTL *is as expressive as* SafeLTL*; similarly,* coSafePLTL *is as expressive as* coSafeLTL.

Theorem 8 can be strenghtened to show that safety and co-safety properties can be equally captured by slightly more relaxed formalisms than SafePLTL and coSafePLTL.

▶ **Proposition 9.** *The fragment of* PLTL *that consists of formulae in negation normal form and forbids the use of* R *(resp.,* U*) is as expressive as* coSafePLTL *(resp.,* SafePLTL*).*

Finally, we interpret linear-time formalisms, in particular LTL, over trees using the standard universal-path semantics. Thus, for an LTL formula φ, we write $t \models \varphi$ iff every path of t starting at the root satisfies φ, or equivalently iff $t \models \mathsf{A}\varphi$.

Word automata. An *ω-word* over Σ is an infinite sequence $\sigma_1\sigma_2\ldots$ of letters from Σ. An *ω-word language* is a set of ω-words. We assume the reader to be familiar with the theory of regular ω-word languages [9].

An ω-word automaton is a tuple $\mathcal{A} = \langle Q, \Sigma, \delta, q_I, F\rangle$, where Q is a finite set of states, Σ is the alphabet, $q_I \in Q$ is an initial state, and $F \subseteq Q$ is a set of final states. The transition function is $\delta : Q \times \Sigma \to Q$ for a deterministic automaton, and $\delta : Q \times \Sigma \to 2^Q$ for a non-deterministic one. Here non-determinism is structural: a state and an input letter may have several successors. This branching can be interpreted either existentially —i.e., the input is accepted if some initial run on it is accepting— or universally —i.e., the input is accepted if all initial runs on it are accepting. We use three-letter acronyms for classes of ω-word automata. The first letter indicates the branching mode: deterministic (D), existential non-deterministic (N), or universal non-deterministic (U). The second letter indicates the acceptance condition: Büchi (B), coBüchi (C), or weak (W), where weak means that every strongly connected component is either contained in F or disjoint from F. The third letter is W for words, to disambiguate from tree automata. For example, UCW denotes universal coBüchi automata. We denote by $\mathscr{L}(\mathcal{A})$ the language recognized by the automaton $\mathcal{A}$.

An automaton $\mathcal{A}$ is *counter-free* if for all states q, all finite words w, and all natural numbers n, if $\mathcal{A}$ admits a path on w^n that starts and ends at q, then it also admits a path on w that starts and ends at q. For a given automata class ABC, we denote by ABC_{cf} its counter-free restriction.

Membership problems. Given two formalisms F and F', we will denote by $F \mapsto F'$ the membership problem from F to F', that is, the problem of deciding, given an expression in F, if there is a language-equivalent expression in F'. For example, MSO $\mapsto$ WMSO

asks whether a given MSO sentence φ can be transformed to an equivalent sentence φ' of WMSO. We will also denote by $F \cap F'$ the set of languages definable by both F and F'.

We recall that APT stands for *alternating parity tree automata*, which are as expressive as MSO and the *modal μ-calculus* (L_μ) over full binary trees. AWT is the subclass of *alternating weak tree automata*, which captures WMSO and the *alternation-free μ-calculus* (AFL_μ). For the definitions of APT, L_μ, and AFL_μ, we refer the reader to, e.g., [48], while for AWT and other variants we will recall their definitions at the beginning of Section 3.

Below is a rephrasing of the celebrated Rabin's reduction theorem.

▶ **Theorem 10** (Rabin's reduction theorem [42, Theorem 29]). *Decidability of* $\mathrm{APT} \mapsto \mathrm{AWT}$ *implies decidability of* $\mathrm{MSO} \mapsto \mathrm{WMSO}$ *and* $\mathrm{L}_\mu \mapsto \mathrm{AFL}_\mu$.

Although direct attacks on the latter logical membership problems have seen limited progress, the corresponding automata-theoretic problem has recently witnessed major advances [15, 14, 18]. Through Rabin's reduction, these advances feed back into the logical setting, yielding progress on $\mathrm{MSO} \mapsto \mathrm{WMSO}$ and $\mathrm{L}_\mu \mapsto \mathrm{AFL}_\mu$.

Our goal is to apply Rabin's reduction scheme within smaller source logics. Rather than taking the input specification from MSO or L_μ, we consider specifications given in MPL or CTL^*. Rabin's reduction then immediately implies that decidability of $\mathrm{APT} \mapsto \mathrm{AWT}$ yields decidability of $\mathrm{MPL} \mapsto \mathrm{MPL} \cap \mathrm{WMSO}$ and $\mathrm{CTL}^* \mapsto \mathrm{CTL}^* \cap \mathrm{AFL}_\mu$. The key question is therefore to identify the target intersection: what is $\mathrm{MPL} \cap \mathrm{WMSO}$, or equivalently $\mathrm{CTL}^* \cap \mathrm{AFL}_\mu$? In the sequel we mostly use the former notation, leaving the equivalence with the latter implicit. As argued in the introduction, this common fragment is a natural object of study, and the present paper makes progress towards its characterisation.

3 The automaton and the conjecture

We introduce the new class of *counter-free hesitant weak ω-tree automata*, by imposing a series of restrictions to alternating Büchi ω-tree automata. We then explain why this model is a natural candidate for capturing the common fragment $\mathrm{MPL} \cap \mathrm{WMSO}$. Finally, we discuss the perhaps surprising fact that $\mathrm{MPL} \cap \mathrm{WMSO}$ is strictly more expressive than FO.

Alternating Büchi ω-tree automata. Given a set X, we think of its elements as *atoms* and we denote by $\mathcal{B}^+(X)$ the set of positive Boolean formulae over X. An *alternating Büchi ω-tree automaton* (ABT) is a tuple $\mathcal{A} = \langle Q, \Sigma, \delta, q_I, F \rangle$, where Q, Σ, q_I, F are as usual and δ is a transition function from $Q \times \Sigma$ to $\mathcal{B}^+(\{\Diamond, \Box\} \times Q)$. Let t be a Σ-tree. A run of $\mathcal{A}$ on t is an *unranked*, *unordered* tree, i.e., a tree in which every node has an arbitrary but finite set of children. It is also labelled over the infinite alphabet $Q \times \mathit{Dom}(t)$, thus associating with each node a state of $\mathcal{A}$ and a corresponding node in t. Intuitively, a labelling (q, v) of a node of a run represents a copy of the automaton that reads an input node v with state q. Let r be an unranked tree labeled over $Q \times \mathit{Dom}(t)$. We define the satisfaction relation between a (q, v)-labelled node u of r and a positive Boolean formula in $\mathcal{B}^+(\{\Diamond, \Box\} \times Q)$ as follows:

- $u \models (\Diamond, q')$ if v has a child v' in t and u has a child u' labelled by (q', v');
- $u \models (\Box, q')$ if for every child v' of v in t, there is a child u' of u labelled by (q', v').

The satisfaction relation $\models$ is naturally extended to the Boolean constants $\top$ and $\bot$ and to every positive Boolean combination of atoms. We then say that r is a run on t if:

1. the root of r is labelled by (q_I, ϵ_t), namely, by the initial state of $\mathcal{A}$ and by the root of t;
2. for every (q, v)-labelled node u of r, $u \models \delta(q, \sigma)$, where σ is the label of v in t.

Concerning the above condition (2), we observe that the definition of the satisfaction relation $\models$ between nodes and positive Boolean formulae requires the existence of a minimal set of children of each node u of a run. On the other hand, it also allows a node u to contain spurious children that are not strictly needed to satisfy condition (2). In particular, the property of being a run of t is preserved under the addition of subtrees that are themselves runs on t, possibly starting at different nodes. This turns out to be useful in order to avoid dealing with runs with leaves: indeed, even in the presence of a transition of the form $\delta(q, \sigma) = \top$, we can still assume that there is an infinite path from u, e.g., one that repeats copies of u itself (note that every such copy of u would still satisfy condition (2) above). Finally, a run r is *accepting* if every maximal path in it satisfies the Büchi condition, i.e., every infinite path has infinitely many nodes labeled by some state in F. We say that a Σ-tree t is accepted by $\mathcal{A}$ if $\mathcal{A}$ admits an accepting run on t.

Weakness. An ABT $\mathcal{A} = \langle Q, \Sigma, \delta, q_I, F\rangle$ is *weak* (AWT) if its state space can be partitioned into a sequence of components $Q_1, ..., Q_k$ in such a way that (1) for every $q \in Q_i$ and every $\sigma \in \Sigma$, $\delta(q, \sigma)$ can only contain atoms with states in $\bigcup_{j \leq i} Q_j$, and (2) for every i, either $Q_i \subseteq F$ or $Q_i \cap F = \emptyset$. We call *accepting* (resp., *rejecting*) the components that are contained in F (resp., disjoint from F).

Hesitancy. An AWT $\mathcal{A} = \langle Q, \Sigma, \delta, q_I, F\rangle$ is *hesitant* (HWT) if every component Q_i of the partition witnessing weakness satisfies one of the following additional conditions, for every $q, q' \in Q_i$ and $\sigma \in \Sigma$:

- *transient component*: q' does not occur in any atom of $\delta(q, \sigma)$;
- *existential component*: q' can occur in the disjunctive normal form of $\delta(q, \sigma)$, but only in atoms of the form $(\Diamond, q')$ and only disjunctively w.r.t. to atoms with other states in Q_i;
- *universal component*: q' can occur in the conjunctive normal form of $\delta(q, \sigma)$, but only in atoms of the form $(\Box, q')$ and only conjunctively w.r.t. to atoms with other states in Q_i.

Linearisation. Towards introducing the remaining restrictions, we need to describe a linearisation operation that transforms any non-transient component of an HWT $\mathcal{A}$ into an ω-word automaton that reads annotated inputs. The idea is to simulate the path of a run of $\mathcal{A}$ that stays inside a given component by a suitable ω-word automaton, with either Büchi or coBüchi acceptance conditions and non-determinism used either existentially or universally depending on the type of component, and at the same time encode the subruns of $\mathcal{A}$ that branch off the path by a suitable annotation of the input.

Let $\mathcal{A} = \langle Q, \Sigma, \delta, q_I, F\rangle$ be an HWT and let q be a state that belongs to a non-transient component Q_i, according to some partition witnessing hesitancy of $\mathcal{A}$. Further let B be the set of atoms from $\{\Diamond, \Box\} \times (Q \setminus Q_i)$, namely, with states outside the component Q_i. The *linearisation* of $\mathcal{A}$ from q is defined as the ω-word automaton $\mathcal{A}^q_{\text{exit}}$ obtained from $\mathcal{A}$ by restricting the state set to Q_i, adding a fresh state q_{exit}, declaring q to be the new initial state, annotating the input letters with subsets C of B, and redefining the transition function and the acceptance condition by a case distinction, as follows.

- If Q_i is an existential component, then
  1. for all $q' \in Q_i \cup \{q_{\text{exit}}\}$, $\delta'(q', (\sigma, C))$ contains q_{exit} iff $q' = q_{\text{exit}}$, or $\delta(q', \sigma) = \bot$, or the disjunctive normal form of $\delta(q', \sigma)$ contains a clause of the form $\bigwedge C$;
  2. for all $q', q'' \in Q_i$, $\delta'(q', (\sigma, C))$ contains q'' iff the disjunctive normal form of $\delta(q', \sigma)$ contains a clause of the form $(\Diamond, q'') \wedge \bigwedge C$.

In addition, if Q_i is accepting (resp., rejecting), then $\mathcal{A}^q_{\text{exit}}$ is used as an NBW (resp., NCW), with Q_i as set of final states.

- If Q_i is a universal component, then
  1. for all $q' \in Q_i \cup \{q_{\text{exit}}\}$, $\delta'(q', (\sigma, C))$ contains q_{exit} iff $q' = q_{\text{exit}}$, or $\delta(q', \sigma) = \top$, or the conjunctive normal form of $\delta(q', \sigma)$ contains a clause of the form $\bigvee C$;
  2. for all $q', q'' \in Q_i$, $\delta'(q', (\sigma, C))$ contains q'' iff the conjunctive normal form of $\delta(q', \sigma)$ contains a clause of the form $(\Box, q'') \vee \bigvee C$.

  In addition, if Q_i is accepting (resp., rejecting), then $\mathcal{A}^q_{\text{exit}}$ is used as an UBW (resp., UCW), with Q_i as set of final states.

Visibility. Let $\mathcal{A} = \langle Q, \Sigma, \delta, q_I, F \rangle$ be an HWT, and let $\theta \in \{\Diamond, \Box\} \times Q$. We write $\mathcal{A}^\theta$ for the automaton obtained from $\mathcal{A}$ by introducing a fresh initial state and by forcing the first transition to be θ, independently of the label of the root of the input tree. We say that a non-transient component Q_i of $\mathcal{A}$ is *visible* if for every pair of distinct annotations $C, C' \subseteq B$ that appear in some transitions of the linearisation $\mathcal{A}^q_{\text{exit}}$, for any $q \in Q_i$, there are atoms $\theta \in C$ and $\theta' \in C'$ for which the ω-tree languages $\mathscr{L}(\mathcal{A}^\theta)$ and $\mathscr{L}(\mathcal{A}^{\theta'})$ are complement of each other. Accordingly, we say that $\mathcal{A}$ is *visible* if every non-transient component is visible.

Counter-freeness. Recall the definition of counter-free ω-word automaton given in the preliminaries. The last restriction that we enforce requires the linearisation of every non-transient component of a visible HWT to be counter-free. Accordingly, we call *counter-free hesitant weak ω-tree automata* (HWT_{cf} for short) a Büchi ω-tree automaton that is weak, hesitant, visible, and counter-free.

We remark that counter-freeness only makes sense when the automaton is visible. Intuitively, this is because visibility guarantees that the annotations used in the linearisation of a component reflect genuine branching alternatives. In other words, distinct annotations are distinguishable by incompatible behaviours outside the component; hence they cannot serve as hidden markers that encode counting information while leaving the linearised ω-word language counter-free. On the other hand, it can be shown that a counter-free HWT which is not visible can recognize regular ω-tree languages that are not even definable in MPL, see, e.g., [3, Example 5.1].

The conjecture. In [35, Theorem 1.1] WMSO was shown to be as expressive as AWT, and in [3, Theorems 5.8 and 5.9] MPL was shown to be as expressive as counter-free, visible, hesitant ω-tree automata (HT_{cf} for short). Because HT_{cf} employ an acceptance condition that is stronger than Büchi but weaker than Parity, we know that $\text{HWT}_{cf} \subseteq \text{HT}_{cf} \cap \text{AWT} \subseteq \text{MPL} \cap \text{WMSO}$. Our Conjecture 1 claims that HWT_{cf} are in fact as expressive as $\text{MPL} \cap \text{WMSO}$.

The findings in the subsequent sections should provide evidence for the conjecture. The first intuition for its validity relies on the fact that MPL restricts the *structure* of quantified sets, while WMSO restricts their *cardinality*, and the two constraints seem to be orthogonal. More precisely, orthogonality should be witnessed at the syntactic level, when we translate MPL and WMSO to their automata counterparts, HT_{cf} and AWT. If our conjecture were false, there would exist, e.g., an AWT that recognises an MPL-definable language, yet structurally violates the constraints required by HT_{cf}. Given the independent nature of the syntactic constraints posed by the two classes of automata, i.e., hesitancy and counter-freeness on one side and weakness on the other side, we belive this is unlikely to happen.

FO is WMPL but not MPL ∩ WMSO. We conclude the section by showing that the common fragment MPL ∩ WMSO is non-trivial. We first note that WMPL, the fragment of MSO in which monadic quantification is restricted to finite paths, is as expressive as FO: indeed, every finite path in a tree is uniquely determined by its endpoints. One might also expect WMPL to coincide with MPL ∩ WMSO, since, as argued earlier, MPL and WMSO impose orthogonal restrictions on monadic quantification. However, these restrictions interact semantically in a subtler way, and the expectation is in fact false:

▶ **Lemma 11.** *There is an ω-tree language T definable in* MPL ∩ WMSO *but not in* FO.

Proof. Consider the language T of ω-trees where every infinite path from the root contains a node carrying the monadic predicate p. The underlying alphabet is $\Sigma = 2^{\{p\}}$. This language is definable in MPL, by the formula $\neg\exists X\forall y(y \in X \to \neg p(y))$. It is equally definable in WMSO, by the formula $\exists X(\text{root} \in X \wedge \forall y(y \in X \to (p(y) \vee \forall z(\text{child}(z,y) \to z \in X))))$, where $\text{root} \in X = \exists x\forall y(x \leq y \wedge x \in X)$ and $\text{child}(z,y) = z > y \wedge \neg\exists w(z > w > y)$. Finally, it was shown in [2, Theorem 3], that this language is not definable in FO. ◀

This shows that MPL ∩ WMSO is a genuine common fragment, not another presentation of FO. It also suggests that the membership problem MPL ↦ FO cannot be settled by a Rabin-style reduction to APT ↦ AWT. Although MPL = HT_{cf} ⊆ APT, reducing a given HT_{cf} to an equivalent AWT would only show that the corresponding MPL formula is equivalent to some WMSO formula; it would not give FO-definability. Thus, even for a logic close to FO such as MPL, FO-definability remains an elusive question, not even reducible to the (still open) problem APT ↦ AWT.

4 PCTL is equivalent to HWT_{cf}

Recall that the witness language of Lemma 11 is over the alphabet $\Sigma = 2^{\{p\}}$ and of the form $T = \{t \mid \text{every infinite path from the root of } t \text{ carries } p\}$. The following lemma shows that T is recognised by an HWT_{cf}, and also serves as a worked example of an HWT_{cf}. Knowing that FO is equivalent to a subclass of HWT_{cf}, as proved in [4], this also implies that FO is strictly less expressive than HWT_{cf}. In particular, T cannot be used to disprove our conjecture.

▶ **Lemma 12.** *The language T from Lemma 11 is recognized by an* HWT_{cf}.

Proof. Let $\mathcal{A}_T = \langle\{q_I\}, 2^{\{p\}}, \delta, q_I, \emptyset\rangle$, where $\delta(q_I, \{p\}) = \top$ and $\delta(q_I, \emptyset) = (\Box, q_I)$. This is an APT that clearly recognises T, since it accepts a tree iff it finds a $\{p\}$-labelled node along every infinite path from the root. We claim that $\mathcal{A}_T$ is also an HWT_{cf}. Observe that $\mathcal{A}_T$ is weak, with only one component $Q_1 = \{q_I\}$ that is rejecting and universal. The linearisation of $\mathcal{A}_T$ from q_I is an UCW of the form $\mathcal{A}^{q_I}_{\text{exit}} = \langle\{q_I, q_{exit}\}, \Sigma', \delta', q_I, \{q_I\}\rangle$. Note that we can assume the annotated alphabet to be $\Sigma' = \Sigma \times \{\emptyset\}$, since no atom outside the component occurs in the image of δ. The transition function $\delta' : \{q_I, q_{\text{exit}}\} \times \Sigma' \to 2^{\{q_I, q_{\text{exit}}\}}$ is such that $\delta'(q_I, (\emptyset, \emptyset)) = \{q_I\}$ and $\delta'(q, (\sigma, \emptyset)) = \{q_{\text{exit}}\}$ in all remaining cases, i.e., when $q = q_{\text{exit}}$ or $\sigma = \{p\}$. It is easy to see that this UCW is visible and counter-free: indeed, there is only one possible annotation that appears in the transitions, i.e., $C = \emptyset$, and the transition graph consists solely of self-loops and a transition from q_I to q_{exit}. $\mathcal{A}_T$ is therefore an HWT_{cf}. ◀

We now turn to a logical characterisation of HWT_{cf} in terms of PCTL. Towards the end of the section, we will see that, conditional on Conjecture 1, this characterisation yields a Rabin-style reduction from the membership problem CTL^* ↦ PCTL to the

automata-theoretic problem APT $\mapsto$ AWT. To prove the characterisation, we introduce two intermediate results: a normal form for PCTL and a syntactic fragment of CTL* that is equivalent to PCTL.

▶ **Lemma 13.** *For every* PCTL *formula, there is an equivalent one generated by:*

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{EF}\psi \mid \mathsf{EG}\psi \mid \mathsf{AF}\psi \mid \mathsf{AG}\psi$$
$$\psi ::= \varphi \mid \psi \vee \psi \mid \psi \wedge \psi \mid \mathsf{Y}\psi \mid \tilde{\mathsf{Y}}\psi \mid \psi \mathsf{S} \psi$$

Proof sketch. The proof is by structural induction. The path formulae of PCTL belong to the fragments of PLTL in negation normal form that forbid the use of either R or U. Thus one can use Proposition 9 to turn those formulae into equivalent ones from SafePLTL and coSafePLTL. This is consistent with the above grammar. ◀

The above normal form clearly shows that PCTL expressive power is deeply linked to the ability to express both *safety* and *co-safety* properties along paths. Moreover, it gives us a way to identify a syntactic fragment of CTL* equivalent to PCTL. This is useful for two reasons: first, it shows how to retain PCTL expressive power while dropping past operators; second, it will be instrumental in the proof that PCTL is expressively equivalent to HWT$_{cf}$.

▶ **Lemma 14.** PCTL *is as expressive as* ObligationCTL**, generated by the grammar*

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{E}\psi \mid \mathsf{E}\chi \mid \mathsf{A}\psi \mid \mathsf{A}\chi$$
$$\psi ::= \varphi \mid \psi \vee \psi \mid \psi \wedge \psi \mid \mathsf{X}\psi \mid \psi \mathsf{U} \psi$$
$$\chi ::= \varphi \mid \chi \vee \chi \mid \chi \wedge \chi \mid \mathsf{X}\chi \mid \chi \mathsf{R} \chi$$

Proof sketch. The proof is again by induction in both directions. For the non-trivial cases, the idea is to exploit the fact that path formulae of the PCTL normal form and those of ObligationCTL* are expressively equivalent, thanks to Theorem 8. ◀

Now, we are ready to establish the equivalence between PCTL and HWT$_{cf}$. We say that an HWT$_{cf}$ is *universal* (resp., *existential*) if its transition function only produces conjunctions (resp., disjunctions) of atoms of the form $(\Box, q)$ (resp., $(\Diamond, q)$). Similarly, we define APCTL (resp., EPCTL) as the fragment of PCTL in which only universal (resp., existential) path quantifiers are allowed.

▶ **Theorem 2.** HWT$_{cf}$ *and* PCTL *are effectively equivalent formalisms. Moreover, universal* HWT$_{cf}$ *(resp., existential* HWT$_{cf}$*) admits translations into* APCTL *(resp.,* EPCTL*).*

Proof sketch. The proof relies on using linearisations of non-transient components of HWT$_{cf}$ in one direction, and ObligationCTL* in the other direction. To translate an HWT$_{cf}$ to an equivalent PCTL formula, we adapt the proof of [4, Theorem 12], proceeding by induction on the components' order. The transient case is straightforward; the case of a universal (resp., existential) component is addressed by linearising the component and translating it into an equivalent SafeLTL (resp., coSafeLTL) formula. Recombining the latter formulas yields an ObligationCTL* formula equivalent to the initial HWT$_{cf}$. Finally, Lemma 14 provides an equivalent PCTL formula. This also proves the second statement of the theorem, since translating a universal (resp., existential) HWT$_{cf}$ never requires existential (resp., universal) path quantification.

For the converse direction, we apply again Lemma 14 to transform the input PCTL formula into an equivalent ObligationCTL* formula. To enforce visibility in the target

automaton, we rely on an expressively complete syntactic variant of ObligationCTL* [3, Section 5.1] designed to isolate path quantifiers, ensuring that every pair of consecutive path quantifiers is explicitly separated. Then, the translation is fairly routine: the only non-trivial case is the path quantifier, which is handled as in, e.g., [3, Theorem 5.9]. ◀

We can now state a Rabin-style reduction theorem, conditional on Conjecture 1:

▶ **Theorem 3.** *Assuming Conjecture 1, the membership problem* CTL* $\mapsto$ PCTL *(or, equally,* MPL $\mapsto$ PCTL*) reduces to the automata-theoretic problem* APT $\mapsto$ AWT*.*

5 The common fragment of PCTL and LTL is decidable

In this section we show another major application of the logical characterization of HWT_{cf} given in Theorem 2: specifically, we establish the decidability of the common fragment of LTL and PCTL. More precisely, we prove that both membership problems PCTL $\mapsto$ LTL and LTL $\mapsto$ PCTL are decidable, and therefore, MSO $\mapsto$ LTL $\cap$ PCTL is decidable too. This result is interesting for at least two reasons: first, it provides decidability results for membership problems of tree logics, which are notoriously among the hardest problems in language theory; second, it offers a new way to attack the almost 40-year-old open problem of deciding the common fragment of LTL and CTL.

Given a regular ω-word language L, let $\triangle[L]$ be the regular ω-tree language of all trees such that every infinite path starting at the root is in L.

▶ **Lemma 15.** *Let $\mathcal{A}$ be a* DBW_{cf}*. One can construct a universal* HWT_{cf} *recognizing* $\triangle[\mathscr{L}(\mathcal{A})]$*.*

Proof sketch. Given a DBW_{cf} $\mathcal{A} = \langle Q, \Sigma, \delta, q_I, F\rangle$, we first construct an equivalent UWW $\mathcal{B}$ by taking two disjoint copies of $\mathcal{A}$. Intuitively, the first copy follows the unique run of $\mathcal{A}$ forever, and it is accepting. Whenever this run enters a non-final state, the universal automaton $\mathcal{B}$ spawns a branch in the second copy, which checks the local obligation that a final state is reached later. Since the second copy is rejecting, such a branch cannot remain there forever; it is discharged exactly when the run reaches a final state. Formally, we let $\mathcal{B} = \langle (Q \times \{1\}) \cup ((Q \setminus F) \times \{2\}), \Sigma, \delta', (q_I, 1), (Q \times \{1\})\rangle$, where δ' is defined as follows:

- if $\delta(q,\sigma) = q' \notin F$, then $\delta'((q,1),\sigma) = \{(q',1),(q',2)\}$ and $\delta'((q,2),\sigma) = \{(q',2)\}$;
- if $\delta(q,\sigma) = q' \in F$, then $\delta'((q,1),\sigma) = \{(q',1)\}$ and $\delta'((q,2),\sigma) = \emptyset$.

Moreover, $\mathcal{B}$ is weak: the first component consists only of accepting states, the second one only of rejecting states, and there are no transitions from the second component back to the first. Counter-freeness is preserved as well, since every cycle in $\mathcal{B}$ projects to a cycle of $\mathcal{A}$.

The second step turns $\mathcal{B}$ into a tree automaton. For each transition of $\mathcal{B}$, every successor state (q,i), for $i \in \{1,2\}$, is replaced by the atom $(\Box,(q,i))$. In this way, the word automaton is simulated uniformly along all branches of the input tree. The resulting automaton is a universal HWT_{cf} and recognises precisely $\triangle[\mathscr{L}(\mathcal{A})]$. ◀

The following theorem gives an automaton-theoretic characterization of a robust class of LTL-definable ω-tree languages, and is crucial for the main decidability result presented in this section (Theorem 5).

▶ **Theorem 16.** *For an* LTL*-definable ω-word language L, the following are equivalent: (i) L is recognized by a* DBW_{cf}*, (ii) $\triangle[L]$ is recognized by a universal* HWT_{cf}*, (iii) $\triangle[L]$ is recognized by an* HWT_{cf}*, (iv) $\triangle[L]$ is recognized by an* AWT*.*

Proof. Assume first that L is recognised by a DBW_{cf}. By Lemma 15, $\triangle[L]$ is recognised by a universal HWT_{cf}, so (i) implies (ii). The implications from (ii) to (iii) and from (iii) to (iv) are immediate, since universal HWT_{cf} are HWT_{cf}, and HWT_{cf} are AWT. It remains to prove the implication from (iv) to (i). Suppose that $\triangle[L]$ is recognised by an AWT. By [35, Theorem 1.1], it is also recognised by a *non-deterministic Büchi tree automaton*. Then, by [24, Theorem 3.1], L is recognised by a DBW. Since L is LTL-definable, it is star-free; hence, after minimising the DBW, we obtain an equivalent DBW_{cf} [38]. ◀

Theorem 16 also yields the following logical equivalences (see again Figure 1):

▶ **Theorem 4.** $\mathrm{LTL} \cap \mathrm{APCTL} = \mathrm{LTL} \cap \mathrm{PCTL} = \mathrm{LTL} \cap \mathrm{AFL}_{\mu}$.

Proof. Let L be an LTL-definable ω-word language. It is enough to show that definability of $\triangle[L]$ in APCTL, in PCTL, and in AFL_{μ} are all equivalent to recognisability of L by a DBW_{cf}. If $\triangle[L]$ is definable in PCTL, then by Theorem 2 it is recognised by an HWT_{cf}, and Theorem 16 yields a DBW_{cf} for L. The case of APCTL follows since $\mathrm{APCTL} \subseteq \mathrm{PCTL}$. If $\triangle[L]$ is definable in AFL_{μ}, then it is recognised by an AWT, thanks to [23]; again Theorem 16 yields a DBW_{cf} for L. Conversely, if L is recognised by a DBW_{cf}, then Theorem 16 gives a universal HWT_{cf} recognising $\triangle[L]$. By Theorem 2, this yields an equivalent APCTL formula. Definability in PCTL follows from $\mathrm{APCTL} \subseteq \mathrm{PCTL}$, and definability in AFL_{μ} follows since HWT_{cf} are AWT, which characterise AFL_{μ} [23]. ◀

The above result might appear surprising, since Bojańczyk proved in [6] that $\mathrm{LTL} \cap \mathrm{ACTL} \subsetneq \mathrm{LTL} \cap \mathrm{CTL}$. Indeed, Theorem 4 implies that it is possible to avoid existential path quantification and capture the common fragment of LTL and PCTL, employing past operators instead. To exemplify this idea, we show how to define in APCTL the language used by Bojańczyk to show the strict inclusion already mentioned.

▶ **Proposition 17.** *The regular ω-tree language $\triangle[(ab)^*a(ab)^*c^{\omega}]$ is definable in* APCTL.

Proof. The formula that defines the $\{a,b,c\}$-tree language $\triangle[(ab)^*a(ab)^*c^{\omega}]$ is as follows, where $\mathsf{P}\psi$ is a shorthand for $\top\mathsf{S}\psi$:

$$a \wedge \mathsf{AF}c \wedge \mathsf{AG}(\neg(b \wedge \mathsf{Y}b) \wedge \neg(a \wedge \mathsf{Y}a \wedge \mathsf{YP}(a \wedge \mathsf{Y}a)) \wedge (\mathsf{Y}c \rightarrow c) \wedge \\ ((c \wedge \mathsf{Y}\neg c) \rightarrow \mathsf{Y}((a \wedge \neg\mathsf{P}(a \wedge \mathsf{Y}a)) \vee (b \wedge \mathsf{P}(a \wedge \mathsf{Y}a)))$$ ◀

In the formula above, past operators are necessary to capture the complex alternation of a, b, and c without employing the existential path quantifier E while respecting the syntactic constraints of CTL. This unveils an unexpected power of past operators in the context of tree temporal logics. We finally conclude with our main decidability results.

▶ **Theorem 5.** *The membership problem* $\mathrm{LTL} \mapsto \mathrm{PCTL}$ *is decidable in* ExpSpace *and is* PSpace*-hard. Moreover,* $\mathrm{PCTL} \mapsto \mathrm{LTL}$ *is decidable.*

Proof. By Theorem 16, given an LTL formula φ, it suffices to decide if $\mathscr{L}(\varphi)$ is recognizable by a DBW (the counter-free requirement is trivially fulfilled, since any LTL definable language must be recognizable by a counter-free automaton). By [23, Theorem 4.1, Theorem 5.1], deciding $\mathrm{LTL} \mapsto \mathrm{DBW}$ is in EXPSPACE and PSPACE-hard.

For $\mathrm{PCTL} \mapsto \mathrm{LTL}$, we can always rewrite a PCTL formula into ObligationCTL* by Lemma 14, and then apply the algorithm from [13] to decide if there is an equivalent LTL formula. ◀

The first statement of the above theorem implies the following.

▶ **Corollary 6.** *Decidability of* PCTL $\mapsto$ CTL *implies decidability of* LTL $\mapsto$ CTL.

Thanks to the above corollary, the 40-year-old open problem can be attacked along a new direction: determine which PCTL formulae are equivalent to some CTL formula. This is a promising new approach, since some subcases are already solved (see, e.g., [26]). To conclude, we can settle that the common fragment LTL $\cap$ PCTL is decidable.

▶ **Corollary 18.** *The membership problem* MSO $\mapsto$ LTL $\cap$ PCTL *is decidable.*

6 No known fragment of WMSO is equivalent to PCTL

In this section we will show that no known fragment of WMSO is equivalent to PCTL. In other words, to understand PCTL expressive power with respect to WMSO, one needs to define a new fragment of WMSO. Indeed, given Theorem 3, one could wonder what is the exact fragment of WMSO equivalent to MPL $\cap$ WMSO. We have shown that, assuming Conjecture 1 to be true, MPL $\cap$ WMSO is PCTL. Nevertheless, it is a natural question to ask for a fragment of WMSO, rather than for a temporal logic. In this section we initiate this work, by isolating the lower and upper bounds of PCTL expressive power with regard to WMSO fragments, to pave the way for future work on this matter.

To the best of our knowledge, there are three defined semantic fragments of WMSO: FO (which is equivalent to WMPL, as already argued), *Weak Chain Logic* (WMCL) [46] and *Weak Tree Logic* (WMTL) [2]. While FO and WMTL have already been defined, we briefly recall the definition of WMCL. In WMCL, second order quantification is restricted to *finite chains*, that is, finite sets of nodes that are pairwise comparable under the transitive closure of the tree's edge relation.

First, while it is already established that FO is strictly less expressive than PCTL [4], this fact also emerges as a direct consequence of Lemma 12 and Theorem 2. Furthermore, WMCL is known to be incomparable with MPL [1]. Because MPL strictly subsumes PCTL, this incomparability necessarily implies that PCTL cannot be equivalent to WMCL. Thus, WMTL remains the only reasonable fragment of WMSO to investigate. Recall that in WMTL set quantification ranges over sets of nodes that are finite subtrees. Although in [2] the authors explored the expressive power of this logic, the precise relationship between WMTL and MPL in terms of expressiveness remained an open problem. Here, we show that these two logics are incomparable, implying also that PCTL is strictly less expressive than WMTL. The following is obtained by an easy translation by structural induction on PCTL formulae.

▶ **Lemma 19.** PCTL *is at most as expressive as* WMTL.

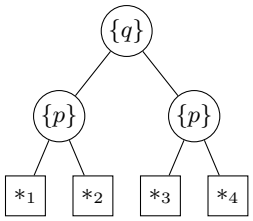
Since there is no defined intermediate WMSO fragment between FO and WMTL, one could believe PCTL to be equivalent to WMTL. However, the following theorem shows this is not the case, while also closing the open question by [2].

▶ **Theorem 7.** WMTL *is incomparable to both* CTL* *and* MPL. *Consequently, no known fragment of* WMSO *is expressively equivalent to* PCTL.

Proof sketch. The difficult direction is WMTL $\not\subseteq$ MPL. Consider the alphabet $2^{\{p,q\}}$. The separating language will be $U =$ "there is a finite subtree containing the root such that: its leaves are labeled $\{p,q\}$, its internal nodes labeled $\{q\}$ have at least one child and its internal nodes not labeled $\{q\}$ have exactly two children". This language is definable in WMTL:

$$\exists X.root \in X \wedge \forall z \in X \rightarrow \wedge \begin{cases} (\neg p(z) \rightarrow \exists y(child(z,y) \wedge y \in X)) \\ (\neg q(z) \rightarrow \forall y(child(z,y) \rightarrow y \in X)) \end{cases}$$

Our claim is that this language is not MPL definable. We show it exploiting two families of trees. First, let a Σ-*multicontext* be a finite or infinite $\Sigma \cup \{*_1, ..., *_k\}$-tree, for $k \in \mathbb{N}$, in which every internal node is labeled by a symbol of Σ and every leaf (if any) is labeled by $*_i$, for some $i \in [1, k]$. Consider the following $2^{\{p,q\}}$-multicontext c:



Let l (resp., r) be the $2^{\{p,q\}}$-tree whose root is labeled by $\{q\}$ (resp., $\{p, q\}$) and all its other nodes are labeled by $\emptyset$. We define t_1 to be c in which the leaf labeled $*_1$ is substituted by l and leaves labeled $*_2, *_3$ and $*_4$ are substituted by r, while t'_1 is defined as c in which leaves labeled $*_1, *_3$ are substituted by l while nodes labeled $*_2, *_4$ are substituted by r. Then, $t_1 \in U$ and $t'_1 \notin U$. We then define infinite such trees as follows: for every $i \geq 2$, t_i is c in which the leaf labeled $*_1$ is substituted by t'_{i-1} and leaves labeled $*_2, *_3$ and $*_4$ are substituted by t_{i-1}, while t'_i is c in which leaves labeled $*_1$ and $*_3$ are substituted by t'_{i-1} while leaves labeled $*_2$ and $*_4$ are substituted by t_{i-1}. Again, for every $i \geq 2$, $t_i \in U$ and $t'_i \notin U$. We show by induction on CTL* formulae that, for every $n \in \mathbb{N}$, CTL* formulae of size at most n, where their size is defined as the length of the strings for representing the formulae, cannot distinguish t_n and t'_n. This implies that CTL* cannot define U, yielding the theorem. ◀

This theorem shows that the branching power of WMTL is in a strong sense more expressive than just path quantification. Our result also shows that WMTL $\neq$ MPL $\cap$ WMSO, further solidifying our conjecture. Indeed, we can now see that if our conjecture is false, there would be not one but even *two* "meaningful" logics strictly in between FO and WMTL, which are already fairly close formalisms with regard to expressiveness. This seems unlikely, since it is arguably already surprising the existence of one such a logic.

7 Conclusions

We have proved a Rabin-style theorem for the MSO fragment called MPL and the temporal logic CTL* (Theorem 3), assuming a conjecture (Conjecture 1) regarding the expressive power of the class of HWT$_{cf}$. We have shown that this class of automata is equivalent to the temporal logic PCTL (Theorem 2) and to the syntactic fragment of CTL* called ObligationCTL* (Lemma 14). Moreover, we have shown that the membership problems LTL $\mapsto$ PCTL and PCTL $\mapsto$ LTL are decidable (Theorem 5), as well as MSO $\mapsto$ LTL $\cap$ PCTL (Corollary 18), implying that the almost 40-year-old open question regarding decidability of LTL $\mapsto$ CTL can be reduced to the arguably easier problem of showing decidability of PCTL $\mapsto$ CTL (Corollary 6). Given the progress already made in, e.g., [26], this seems promising. Finally, we have shown that the expressive power of PCTL is strictly in between that of FO and WMTL ([4] and Theorem 7), implying that no known WMSO fragment is equivalent to PCTL, leaving the definition of such a fragment as a natural open question. The obvious final open question regards the truth of Conjecture 1.

References

1 P. Balbiani and E. Lorini. Ockhamist Propositional Dynamic Logic: a natural link between PDL and CTL. In *International Workshop on Logic, Language, Information, and Computation*, pages 251–265. Springer, 2013.

2 M. Benerecetti, L. Bozzelli, F. Mogavero, and A. Peron. Quantifying over Trees in Monadic Second-Order Logic. In *LICS'23*, pages 1–13. IEEE Computer Society, 2023.

3 M. Benerecetti, L. Bozzelli, F. Mogavero, and A. Peron. Automata-Theoretic Characterisations of Branching-Time Temporal Logics. In *ICALP'24*, LIPIcs 297, pages 128:1–20. Leibniz-Zentrum fuer Informatik, 2024.

4 M. Benerecetti, D. Della Monica, A. Matteo, F. Mogavero, and G. Puppis. An Automaton-Based Characterisation of First-Order Logic over Infinite Trees. In *GANDALF'25*, EPTCS 428, pages 45–61, 2025.

5 M. Bojańczyk. The Common Fragment of ACTL and LTL. In *FOSSACS'08*, LNCS 4962, pages 172–185. Springer, 2008. doi:10.1007/978-3-540-78499-9_13.

6 Mikołaj Bojańczyk. The common fragment of actl and ltl. In *International Conference on Foundations of Software Science and Computational Structures*, pages 172–185. Springer, 2008.

7 U. Boker and Y. Shaulian. Automaton-Based Criteria for Membership in CTL. In *LICS'18*, pages 155–164. Association for Computing Machinery, 2018.

8 J.R. Büchi. Weak Second-Order Arithmetic and Finite Automata. *MLQ*, 6(1-6):66–92, 1960.

9 J.R. Büchi. On a Decision Method in Restricted Second-Order Arithmetic. In *ICLMPS'62*, pages 1–11. Stanford University Press, 1962.

10 J.R. Büchi. On a Decision Method in Restricted Second Order Arithmetic. In *Studies in Logic and the Foundations of Mathematics*, volume 44, pages 1–11. Elsevier, 1966.

11 Edward Chang, Zohar Manna, and Amir Pnueli. Characterization of temporal property classes. In *International Colloquium on Automata, Languages, and Programming*, pages 474–486. Springer, 1992. doi:10.1007/3-540-55719-9_97.

12 Y. Choueka. Theories of Automata on ω-Tapes: A Simplified Approach. *JCSS*, 8(2):117–141, 1974.

13 E.M. Clarke and I.A. Draghicescu. Expressibility Results for Linear-Time and Branching-Time Logics. In *REX'88*, LNCS 354, pages 428–437. Springer, 1988. doi:10.1007/BFb0013029.

14 Thomas Colcombet, Denis Kuperberg, Christof Löding, and Michael Vanden Boom. Deciding the weak definability of büchi definable tree languages. In *CSL*, 2013.

15 Thomas Colcombet and Christof Löding. The non-deterministic mostowski hierarchy and distance-parity automata. In *International Colloquium on Automata, Languages, and Programming*, pages 398–409. Springer, 2008. doi:10.1007/978-3-540-70583-3_33.

16 E.A. Emerson and J.Y. Halpern. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. *JACM*, 33(1):151–178, 1986.

17 T. Hafer and W. Thomas. Computation Tree Logic CTL* and Path Quantifiers in the Monadic Theory of the Binary Tree. In *ICALP'87*, LNCS 267, pages 269–279. Springer, 1987. doi:10.1007/3-540-18088-5_22.

18 Olivier Idir and Karoliina Lehtinen. Using games and universal trees to characterise the nondeterministic index of tree languages. *arXiv preprint arXiv:2504.16819*, 2025.

19 D. Janin and I. Walukiewicz. On the Expressive Completeness of the Propositional mu-Calculus with Respect to Monadic Second Order Logic. In *CONCUR'96*, LNCS 1119, pages 263–277. Springer, 1996. doi:10.1007/3-540-61604-7_60.

20 H.W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, CA, USA, 1968.

21 D. Kozen. Results on the Propositional muCalculus. *TCS*, 27(3):333–354, 1983. doi:10.1016/0304-3975(82)90125-6.

22 O. Kupferman and M.Y. Vardi. Weak Alternating Automata and Tree Automata Emptiness. In *STOC'98*, pages 224–233. Association for Computing Machinery, 1998.

23 O. Kupferman and M.Y. Vardi. From Linear Time to Branching Time. *TOCL*, 6(2):273–294, 2005. doi:10.1145/1055686.1055689.

24 Orna Kupferman, Shmuel Safra, and Moshe Y Vardi. Relating word and tree automata. *Annals of Pure and Applied Logic*, 138(1-3):126–146, 2006. doi:10.1016/j.apal.2005.06.009.

25 R.E. Ladner. Application of Model Theoretic Games to Discrete Linear Orders and Finite Automata. *IC*, 33(4):281–303, 1977. doi:10.1016/S0019-9958(77)90443-0.

26 François Laroussinie and Ph Schnoebelen. A hierarchy of temporal logics with past. *Theoretical Computer Science*, 148(2):303–324, 1995. doi:10.1016/0304-3975(95)00035-U.

27 H. Läuchli. A Decision Procedure for the Weak Second-Order Theory of Linear Order. In *LC'66*, volume 50, pages 189–197. North-Holland, 1968.

28 M. Maidi. The Common Fragment of CTL and LTL. In *FOCS'00*, pages 643–652. IEEE Computer Society, 2000. doi:10.1109/SFCS.2000.892332.

29 R. McNaughton. Testing and Generating Infinite Sequences by a Finite Automaton. *IC*, 9(5):521–530, 1966. doi:10.1016/S0019-9958(66)80013-X.

30 R. McNaughton and S. Papert. *Counter-Free Automata*. MIT Press, 1971.

31 F. Moller and A.M. Rabinovich. On the Expressive Power of CTL*. In *LICS'99*, pages 360–368. IEEE Computer Society, 1999. doi:10.1109/LICS.1999.782631.

32 F. Moller and A.M. Rabinovich. Counting on CTL*: On the Expressive Power of Monadic Path Logic. *IC*, 184(1):147–159, 2003. doi:10.1016/S0890-5401(03)00104-4.

33 A.W. Mostowski. Regular Expressions for Infinite Trees and a Standard Form of Automata. In *SCT'84*, LNCS 208, pages 157–168. Springer, 1984. doi:10.1007/3-540-16066-3_15.

34 David E. Muller and Paul E. Schupp. Alternating automata on infinite trees. *Theoretical Computer Science*, 54:267–276, 1987. doi:10.1016/0304-3975(87)90133-2.

35 D.E. Muller, A. Saoudi, and P.E. Schupp. Alternating Automata, the Weak Monadic Theory of Trees and its Complexity. *TCS*, 97(2):233–244, 1992. doi:10.1016/0304-3975(92)90076-R.

36 D. Perrin and J. Pin. First-Order Logic and Star-Free Sets. *JCSS*, 32(3):393–406, 1986. doi:10.1016/0022-0000(86)90037-1.

37 D. Perrin and J. Pin. *Infinite Words*. Pure and Applied Mathematics. Elsevier, 2004.

38 Dominique Perrin and Jean-Éric Pin. Semigroups and automata on infinite words. *NATO ASI Series C Mathematical and Physical Sciences-Advanced Study Institute*, 466:49–72, 1995.

39 A. Pnueli. The Temporal Logic of Programs. In *FOCS'77*, pages 46–57. IEEE Computer Society, 1977. doi:10.1109/SFCS.1977.32.

40 A. Pnueli. The Temporal Semantics of Concurrent Programs. *TCS*, 13:45–60, 1981. doi:10.1016/0304-3975(81)90110-9.

41 A. Prior. *Past, Present and Future*. Oxford University Press, 1967.

42 Michael O Rabin. Weakly definable relations and special automata. In *Studies in Logic and the Foundations of Mathematics*, volume 59, pages 1–23. Elsevier, 1970.

43 M.O. Rabin. Decidability of Second-Order Theories and Automata on Infinite Trees. *TAMS*, 141:1–35, 1969.

44 M.O. Rabin. Weakly Definable Relations and Special Automata. In *Studies in Logic and the Foundations of Mathematics*, volume 59, pages 1–23. Elsevier, 1970.

45 M.P. Schützenberger. On Finite Monoids Having Only Trivial Subgroups. *IC*, 8(2):190–194, 1965. doi:10.1016/S0019-9958(65)90108-7.

46 W. Thomas. Logical Aspects in the Study of Tree Languages. In *CAAP'84*, pages 31–50. Cambridge University Press, 1984.

47 W. Thomas. Ehrenfeucht Games, the Composition Method, and the Monadic Theory of Ordinal Words. In *Structures in Logic and Computer Science: A Selection of Essays in Honor of A. Ehrenfeucht*, pages 118–143. Springer, 1997. doi:10.1007/3-540-63246-8_8.

48 T. Wilke. Alternating Tree Automata, Parity Games, and Modal muCalculus. *BBMS*, 8(2):359–391, 2001.

49 P. Wolper. Temporal Logic Can Be More Expressive. *IC*, 56(1-2):72–99, 1983. doi:10.1016/S0019-9958(83)80051-5.