Cycle Detection in Computation Tree Logic[☆]

Gaëlle Fontaine[a,1], Fabio Mogavero[b,1], Aniello Murano[c,1], Giuseppe Perelli[b,1], Loredana Sorrentino[c,1]

[a] *Universidad de Chile, Santiago de Chile, Chile.*
[b] *University of Oxford, Oxford, UK*
[c] *Università degli Studi di Napoli "Federico II", Napoli, Italy.*

Abstract

Temporal logic is a powerful formalism widely studied in formal verification. It allows reasoning about the ongoing behavior of a system, without talking explicitly about time. Two logic frameworks have been mainly investigated: (i) linear-time temporal logics, such as LTL, suitable to describe single computations of a system, and (ii) branching-time temporal logics, such as CTL and CTL*, useful to reason about its computational tree structure.

Systems are often made by iterative and recursive structures. Then, reasoning about computational cycles becomes crucial both in their specification and verification. Indeed, this is the case when we use automata and game-theoretic approaches to handle decision problems such as model checking and satisfiability. Surprisingly, no temporal logic has been studied so far with explicit ability of talking about cycles.

In this paper we introduce Cycle-CTL*, an extension of CTL* with cycle quantifications that are able to predicate over cycles. The introduced logic turns out to be very expressive. Indeed, we prove that it strictly extends CTL* and is orthogonal to μCalculus. We also give an evidence of its usefulness by providing few examples involving non-regular properties. We extensively investigate both the model-checking and satisfiability problems for Cycle-CTL* and some of its variants/fragments.

Keywords: Temporal Logic, Model Checking, Satisfiability, Verification

[☆] This paper is an extended version of [15].

1. Introduction

Temporal logic is a suitable framework largely used in formal system verification [27, 8, 11, 10]. It allows to specify and reason in a rigorous manner about the temporal evolution of a system, without talking explicitly about the elapsing of time. Two fundamental decision problems involving temporal logics have been deeply investigated: *model checking* and *satisfiability*. The former, given a mathematical model of the system, such as a *Kripke* structure, asks whether it satisfies a temporal logic formula specifying its desired behavior. The latter, instead, checks whether the temporal logic specification is consistent and, thus, a corresponding system is feasible [10].

In several situations, reasoning about system correctness and, in particular, solving the above decision questions, reduces to detect precise cycle properties over the system model. For example, in the classical automata-theoretic approach there are settings in which the satisfiability question reduces to first build a Büchi automaton accepting all models of the formula and then to check for its non-emptiness [22]. The latter can be solved by looking for a "lasso", that is a path from the initial state to a final state belonging to a cycle [22, 18]. Similarly, if one uses a game-theory approach, solving the model checking or the satisfiability questions reduces to first construct a two-player game, such as a Büchi or a parity game [13, 22, 23, 3, 16, 31], and then check for the existence of a winning strategy for a designated player. The latter can be reduced to check whether it has the ability to confine the evolution of the game (a *play*) over some specific cycle over the arena, no matter how the other player behaves.

Depending on the view of the underlying nature of time, two types of temporal logics are mainly considered. In *linear-time temporal logics*, such as LTL [27], time is treated as if each moment in time has a unique possible future. Conversely, in *branching-time temporal logics* such as CTL [8] and CTL* [12] each moment in time may split into various possible futures. Then, to express properties along one or all the possible futures we make use of existential and universal quantifiers. Noticeably, LTL is suitable to express path properties; CTL is more appropriate to express state-based properties; finally, CTL* has the power to express combinations of path and state properties. In the years, these logics have been extended in a number of ways in order to express very complicated specification properties. Surprisingly, no temporal logic has been introduced so far to reason explicitly about cycles, despite their usefulness. In addition to the technical motivation

mentioned above, there are often cases in which it is useful to distinguish between purely infinite behaviors, like those occurring in infinite-state systems, from regular infinite behaviors [6, 19], which can be detected by looking for cycles. Moreover, also in finite-state systems there are infinite behaviors that are not regular, like the one referred as *prompt* in [20, 26], which also can be detected by looking at cyclic computations.

In this paper we introduce *Cycle*-CTL* (CTL$^\star_{\circlearrowleft}$), an extension of the classical logic CTL* along with the ability to predicate over cycles. For a *cycle* we mean a path that passes through its initial state infinitely often. Syntactically, CTL$^\star_{\circlearrowleft}$ is obtained by enriching CTL* with two novel cycle quantifiers, namely the existential one $\mathtt{E}^{\circlearrowleft}$ and the universal one $\mathtt{A}^{\circlearrowleft}$. Note that CTL$^\star_{\circlearrowleft}$ still uses the classical quantifiers $\mathtt{E}$ and $\mathtt{A}$. Hence, we can use it to specify models whose behaviour results as an opportune combination of standard paths and cycles. In particular, CTL$^\star_{\circlearrowleft}$ can specify the existence of a lasso within a model.

We study the expressiveness of CTL$^\star_{\circlearrowleft}$ and show that it is strictly more expressive than CTL* but orthogonal to μ-calculus. To give an evidence of the power and usefulness of the introduced logic, we provide some examples along the paper. Precisely, we first show how CTL$^\star_{\circlearrowleft}$ can be used to reasoning, in a very natural way, about liveness properties restricted to cycles. Precisely, we show how to specify that some designated properties recurrently occurs in the starting state of a cycle. As another example, we show the ability of the logic to handle non-regular properties such as the "*prompt-parity condition*" [26]. In temporal logic, we can specify properties that will eventually hold, but this gives no bound on the moment they will occur. Prompt temporal logics and games have been deeply investigated in order to restrict reasoning about properties that only occur in bounded time [7, 1, 21, 4, 26].

We investigate both the model checking and the satisfiability questions for CTL$^\star_{\circlearrowleft}$ and provide some automata-based solutions. For the model checking question we provide a PSPACE upper-bound by opportunely extending the classical approach that is used for CTL* [22]. Specifically, we add a machinery consisting of an appropriate Büchi automaton that checks in parallel whether a path is a cycle and satisfies a required formula. Concerning the satisfiability question, we introduce instead a novel approach that makes use of two-way automata [28]. These automata, largely investigate and used in formal verification [5, 14, 19], allow to traverse trees both in forward and backward. The reason why we cannot use and extend the classical approach provided for CTL* (see [22]) resides on the fact that such an approach makes

strongly use of some positive properties that hold for CTL*, among the others the tree- and the finite-model ones. Unluckily and unsurprisingly, due to the ability in CTL$^\star_{\circlearrowleft}$ to force (and even more to forbid) the existence of cycles, we lose in this logic both these properties. This requires the introduction of novel and *ad hoc* definitions of bisimulation and tree-like unwinding to be used along with the automata-based approach. In particular, two-way tree automata are used to collect all tree representations of such tree-like unwinding structures. By means of this machinery we show that the satisfiability question for the full logic is 3ExpTime.

In addition to CTL$^\star_{\circlearrowleft}$, we also introduce Simple-CTL$^\star_{\circlearrowleft}$: a semantic variant of the logic in which the cycle quantifications predicate only on simple cycles. By using a similar reasoning as for CTL$^\star_{\circlearrowleft}$, also Simple-CTL$^\star_{\circlearrowleft}$ strictly includes CTL*. In particular, it is orthogonal to μ-calculus. We investigate both the model-theoretic and the satisfiability problems for Simple-CTL$^\star_{\circlearrowleft}$, showing that their complexities correspond to the ones for CTL*. Finally, we investigate Cycle-CTL and Simple-Cycle-CTL as the natural CTL-like fragments of the introduced logics.

Outline of the paper. The paper is divided into sections as follows. In Section 2, we introduce the syntax and semantics of *Cycle*-CTL* and *Simple-Cycle*-CTL*, as well as the fragments corresponding to CTL. We also introduce some example to make the reader familiar with the new logic. In Section 3, we analyze the model-theoretic properties of these logics. In particular, we first prove that they are not invariant under the classic notion of bisimulation, this showing that they cannot be embedded into either CTL* or μ-calculus. Then, we introduce the notion of *Cycle-bisimulation*, a refinement of the classic bisimulation for which our logic and its fragments are invariant. In Section 4 we analyze the computational complexities of both the model-checking and the satisfiability problem. Finally, in Section 5 we provide some discussion and future work.

2. Computation-Tree Logic with Cycle Detection

In this section we introduce and discuss the syntax and semantics of Cycle-CTL* (CTL$^\star_{\circlearrowleft}$, for short) and Simple-Cycle-CTL* (CTL$^\star_{s\circlearrowleft}$, for short), as well as their fragments CTL$_{\circlearrowleft}$ and CTL$_{s\circlearrowleft}$, respectively. To do this, we first recall the concept of Kripke structure and some related basic notions. Finally, we also discuss some interesting problems that can be expressed in our logic.

Models We first provide the definition of the underlying model for our logics.

Definition 1 (Kripke Structure)**.** *A* Kripke structure *(*KS*, for short) [17] over a finite set of* atomic propositions AP *is a tuple* $\mathcal{K} \triangleq \langle \mathrm{AP}, \mathrm{W}, R, \mathsf{L}, w_{\mathrm{o}} \rangle$*, where* W *is an enumerable non-empty set of* worlds*,* $w_{\mathrm{o}} \in \mathrm{W}$ *is a designated* initial world*,* $R \subseteq \mathrm{W} \times \mathrm{W}$ *is a left-total* transition relation*, and* $\mathsf{L} : \mathrm{W} \mapsto 2^{\mathrm{AP}}$ *is a* labeling function *mapping each world to the set of atomic propositions true in that world.*

A *path* in $\mathcal{K}$ is an infinite sequence of worlds $\pi \in \mathrm{Pth} \subseteq \mathrm{W}^{\omega}$ such that, for all $i \in \mathbb{N}$, it holds that $((\pi)_i, (\pi)_{i+1}) \in R$. We denote by $\mathsf{fst}(\pi) \triangleq \pi_0$ and $(\pi)_i \triangleq \pi_i$ the first and i-th element of π. For a path π, we say that π is a *cycle* if, for all $i \in \mathbb{N}$, there exists $j \in \mathbb{N}$, with $j > i$, such that $(\pi)_j = \mathsf{fst}(\pi)$. We denote by $\mathrm{Cyc} \subseteq \mathrm{Pth}$ the set of cycles. A cycle π is a *simple cycle* if there is a strictly increasing sequence $(n_i)_{i \in \mathbb{N}}$ such that, for all $i \in \mathbb{N}$, *(a)* $\pi_{n_i} = \pi_{\mathrm{o}}$ and *(b)* for all $i \in \mathbb{N}$ and for all $n_i < j < k < n_{i+1}$, we have $\pi_j \neq \pi_k$. We denote by $\mathrm{SCyc} \subseteq \mathrm{Cyc}$ the set of simple cycle. For a given path π, we denote by $\mathsf{L}(\pi)$ the sequence γ in $(2^{\mathrm{AP}})^{\omega}$ such that $(\gamma)_i = \mathsf{L}(\pi_i)$ for all $i \in \mathbb{N}$. Moreover, $(\pi)_{\leq i} \triangleq \pi_0 \cdots \pi_i$ and $(\pi)_{\geq i} \triangleq \pi_i \cdot \pi_{i+1} \cdots$ represent the prefix up to and the suffix from position i of π. Prefixes of a path are also called *tracks* and denoted by $\rho \in \mathrm{Trk} \subseteq \mathrm{W}^{+}$. We also denote by $\mathsf{lst}(\rho)$ the last element occurring in the track ρ. Finally, all the definitions given above for paths naturally apply to tracks.

By $\mathrm{Trk}(w)$ and $\mathrm{Pth}(w)$ we denote the set of tracks and paths starting from w, respectively. By $\mathrm{Cyc}(w)$ and $\mathrm{SCyc}(w)$ we denote the set of cycles and simple cycles starting from w, respectively. Intuitively, tracks and paths of a KS $\mathcal{K}$ are legal sequences, either finite or infinite, of reachable worlds that can be seen as partial or complete descriptions of possible *computations* of the system modelled by $\mathcal{K}$.

For a pair $(w_1, w_2) \in R$, we say that w_2 is an R-successor of w_1. Note that in case R is a function, then each world w has only one R-successor. This implies that, starting from the initial world w_{o}, there is a unique legal path. Such structures are called LTL *models*.

Syntax $\mathrm{CTL}^{\star}_{\circlearrowleft}$ extends $\mathrm{CTL}^{\star}$ [9] by means of two additional path operators, $\mathtt{E}^{\circlearrowleft}\psi$ and $\mathtt{A}^{\circlearrowleft}\psi$, which respectively read as "there exists a cycle path satisfying ψ" and "for all cycle paths ψ holds". As for $\mathrm{CTL}^{\star}$, the syntax

includes path-formulas, expressing properties over sequences of words, and state-formulas, expressing properties over a single word. State and path formulas are defined by mutual induction as follows.

Definition 2 ($\mathrm{CTL}^{\star}_{\circlearrowleft}$ syntax)**.** $\mathrm{CTL}^{\star}_{\circlearrowleft}$ *formulas are inductively built from a set of atomic propositions* AP*, by using the following grammar, where* $p \in \mathrm{AP}$*:*

$$\phi := p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \mathtt{E}\psi \mid \mathtt{A}\psi \mid \mathtt{E}^{\circlearrowleft}\psi \mid \mathtt{A}^{\circlearrowleft}\psi$$
$$\psi := \phi \mid \neg\psi \mid \psi \wedge \psi \mid \psi \vee \psi \mid \mathtt{X}\psi \mid \psi\mathtt{U}\psi$$

Moreover, for $\mathrm{CTL}^{\star}_{\circlearrowleft}$ and all the fragments introduced below, all the other Boolean connectives are derived as usual. In particular, we make use of the *implication*, defined as $\phi_1 \rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2$, and *equivalence*, defined as $\phi_1 \leftrightarrow \phi_2 \equiv (\phi_1 \rightarrow \phi_2) \wedge (\phi_2 \rightarrow \phi_1)$.

All the formulas generated by a ϕ-rule are called *state-formulas*, while the formulas generated by a ψ-rule are called *path-formulas*.

By $\mathsf{sub}(\varphi)$ we denote the set of all subformulas of φ, and by $\mathsf{sub}_s(\varphi)$ we denote the set of state subformulas of φ.

Similarly to $\mathrm{CTL}^{\star}$, we define the syntactic fragment $\mathrm{CTL}_{\circlearrowleft}$ of $\mathrm{CTL}^{\star}_{\circlearrowleft}$ in which the nesting of temporal operators in the scope of the same path quantifiers is not allowed. Formally, we have the following.

Definition 3 ($\mathrm{CTL}_{\circlearrowleft}$ syntax)**.** $\mathrm{CTL}_{\circlearrowleft}$ *formulas are inductively built from a set of atomic propositions* AP *, by using the following grammar, where* $p \in \mathrm{AP}$*:*

$$\phi := p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \mathtt{EX}\phi \mid \mathtt{AX}\phi \mid \mathtt{E}(\phi\mathtt{U}\phi) \mid \mathtt{A}(\phi\mathtt{U}\phi) \mid$$
$$\mathtt{E}^{\circlearrowleft}\mathtt{X}\phi \mid \mathtt{A}^{\circlearrowleft}\mathtt{X}\phi \mid \mathtt{E}^{\circlearrowleft}(\phi\mathtt{U}\phi) \mid \mathtt{A}^{\circlearrowleft}(\phi\mathtt{U}\phi)$$

Alternatively to the quantification over cycles, in this paper we also address the quantification over simple cycles. To do this, we introduce a variant of the logic, in which the cycle path quantification is replaced by a simple cycle path one. Formally, by knowing that, a (state or path) formula φ is in *normal form* if for all subformulas $\neg\psi$ in $\mathsf{sub}(\varphi)$, the formula ψ is an atomic proposition, we have the following definition.

Definition 4 ($\mathrm{CTL}^{\star}_{s\circlearrowleft}$ syntax)**.** *Given* AP *be a set of atomic propositions, we use the following grammar, where* $p \in \mathrm{AP}$*:*

$$\phi := p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \mathtt{E}\psi \mid \mathtt{A}\psi \mid \mathtt{E}_s^{\circlearrowleft}\psi \mid \mathtt{A}_s^{\circlearrowleft}\psi$$
$$\psi := \phi \mid \neg\psi \mid \psi \wedge \psi \mid \psi \vee \psi \mid \mathtt{X}\psi \mid \psi\mathtt{U}\psi$$

Finally, as in the case for $\mathrm{CTL}^{\star}_{\circlearrowleft}$, we also introduce the CTL-like fragment of $\mathrm{CTL}^{\star}_{s\circlearrowleft}$.

Definition 5 ($\mathrm{CTL}_{s\circlearrowleft}$ syntax)**.** $\mathrm{CTL}_{s\circlearrowleft}$ *formulas are inductively built from a set of atomic propositions* AP *, by using the following grammar, where* $p \in \mathrm{AP}$*:*

$$\phi := p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \mathtt{EX}\psi \mid \mathtt{AX}\phi \mid \mathtt{E}(\phi_1\mathtt{U}\phi_2) \mid \mathtt{A}(\phi_1\mathtt{U}\phi_2) \mid$$
$$\mathtt{E}_s^{\circlearrowleft}\mathtt{X}\phi \mid \mathtt{A}_s^{\circlearrowleft}\mathtt{X}\phi \mid \mathtt{E}_s^{\circlearrowleft}(\phi_1\mathtt{U}\phi_2) \mid \mathtt{A}_s^{\circlearrowleft}(\phi_1\mathtt{U}\phi_2)$$

Given a formula φ in normal form, we define its *simple cycle translation* as the formula obtained by replacing each symbol $\mathtt{E}^{\circlearrowleft}$ in the formula φ, by the symbol $\mathtt{E}_s^{\circlearrowleft}$. The simple cycle translation of φ is denoted by $(\varphi)_s$.

Semantics The semantics for $\mathrm{CTL}^{\star}_{\circlearrowleft}$ is defined *w.r.t.* Kripke structures. It extends the one for $\mathrm{CTL}^{\star}$, with the addition of two new definitions for two cycle path quantifiers.

Definition 6. *The semantics of* $\mathrm{CTL}^{\star}_{\circlearrowleft}$ *formulas is recursively defined as follows. For a Kripke structure* $\mathcal{K}$*, a world* w*, a path* π *and a natural number* $i \in \mathbb{N}$*, we have that:*

- *For all state formulas* ϕ*,* ϕ_1*, and* ϕ_2*:*
  - $\mathcal{K}, w \models p$ *if* $p \in \mathsf{L}(w)$*;*
  - $\mathcal{K}, w \models \neg\phi$ *if* $\mathcal{K}, w \not\models \phi$*;*
  - $\mathcal{K}, w \models \phi_1 \wedge \phi_2$ *if both* $\mathcal{K}, w \models \phi_1$ *and* $\mathcal{K}, w \models \phi_2$*;*
  - $\mathcal{K}, w \models \phi_1 \vee \phi_2$ *if either* $\mathcal{K}, w \models \phi_1$ *or* $\mathcal{K}, w \models \phi_2$*;*
  - $\mathcal{K}, w \models \mathtt{E}\psi$ *if there exists a path* π *in* $\mathrm{Pth}(w)$ *such that* $\mathcal{K}, \pi, 0 \models \psi$*;*
  - $\mathcal{K}, w \models \mathtt{A}\psi$ *if, for all paths* π *in* $\mathrm{Pth}(w)$*, it holds that* $\mathcal{K}, \pi, 0 \models \psi$
  - $\mathcal{K}, w \models \mathtt{E}^{\circlearrowleft}\psi$ *if there exists a path* π *in* $\mathrm{Cyc}(w)$ *and* $\mathcal{K}, \pi, 0 \models \psi$*;*
  - $\mathcal{K}, w \models \mathtt{A}^{\circlearrowleft}\psi$ *if, for all paths* π *in* $\mathrm{Cyc}(w)$*, it holds that* $\mathcal{K}, \pi, 0 \models \psi$*.*
- *For path formulas* ϕ*,* ψ*,* ψ_1*, and* ψ_2*:*
  - $\mathcal{K}, \pi, i \models \phi$ *if* $\mathcal{K}, (\pi)_i \models \phi$*;*

- $\mathcal{K}, \pi, i \models \neg\psi$ *if* $\mathcal{K}, \pi, i \not\models \psi$*;*
- $\mathcal{K}, \pi, i \models \psi_1 \wedge \psi_2$ *if both* $\mathcal{K}, \pi, i \models \psi_1$ *and* $\mathcal{K}, \pi, i \models \psi_2$*;*
- $\mathcal{K}, \pi, i \models \psi_1 \vee \psi_2$ *if either* $\mathcal{K}, \pi, i \models \psi_1$ *or* $\mathcal{K}, \pi, i \models \psi_2$*;*
- $\mathcal{K}, \pi, i \models \mathtt{X}\psi$ *if* $\mathcal{K}, \pi, i+1 \models \psi$*;*
- $\mathcal{K}, \pi, i \models \psi_1 \mathtt{U} \psi_2$ *if there exists* $k \in \mathbb{N}$ *such that* $\mathcal{K}, \pi, i+k \models \psi_2$ *and* $\mathcal{K}, \pi, i+j \models \psi_1$*, for all* $j \in [0, k[$*;*

We say that π *satisfies the path formula* ϕ *over* $\mathcal{K}$*, and write* $\mathcal{K}, \pi \models \phi$*, if* $\mathcal{K}, \pi, 0 \models \phi$*. Also, we say that* $\mathcal{K}$ *satisfies the state formula* φ*, and write* $\mathcal{K} \models \varphi$*, if* $\mathcal{K}, w_I \models \varphi$*.*

Definition 7. *The semantics of* $\mathrm{CTL}^{\star}_{s\circlearrowleft}$ *is recursively defined as follows. Given a Kripke structure* $\mathcal{K}$*, a world* w*, a path* π *and a natural number* $i \in \mathbb{N}$*, we have that:*

- $\mathcal{K}, w \models \mathtt{E}^{\circlearrowleft}_{s}\psi$ *if there is a* **simple** *cycle* π *with beginning state* w*, such that* $\mathcal{K}, \pi \models \psi$*;*
- $\mathcal{K}, w \models \mathtt{A}^{\circlearrowleft}_{s}\psi$ *if for all* **simple** *cycle* π *with beginning state* w*, it holds that* $\mathcal{K}, \pi \models \psi$*;*

All the remaining cases are defined as in Definition 6.

Regarding $\mathrm{CTL}_{\circlearrowleft}$ and $\mathrm{CTL}_{s\circlearrowleft}$, observe that they are a syntactic fragment of $\mathrm{CTL}^{\star}_{\circlearrowleft}$ and $\mathrm{CTL}^{\star}_{s\circlearrowleft}$, respectively. Therefore, their semantics is defined by following Definition 6 and Definition 7, respectively.

Examples In this section, we provide some properties that are expressible with $\mathrm{CTL}^{\star}_{\circlearrowleft}$.

Assume that there is a system composed by two processes, requesting to access a resource, and a scheduler, releasing such resource in a mutually exclusive way, *i.e.*, the resource is never used by the two processes at the same time. Every time the scheduler grants the resource to process i, such resource is exclusively used by process i until the system goes back to the decision point, that is, the state in which the scheduler released the resource. We denote by $\mathtt{dec}$ the atomic proposition labeling the states that are decision points (that is, the moment where the scheduler makes a decision) and by $\mathtt{res}_1$, $\mathtt{res}_2$ the atomic propositions representing the fact that the resource is

released to processes 1 and 2, respectively. The above described situation can be expressed with the $\mathrm{CTL}^{\star}_{\circlearrowleft}$ formula $\varphi_i = \mathtt{E}^{\circlearrowleft}((\mathtt{dec} \wedge \neg \mathtt{res}_i \wedge \mathtt{G}\neg \mathtt{res}_{3-i}) \rightarrow \mathtt{F}\ \mathtt{res}_i)$, for $i \in \{1, 2\}$. Note that in that formula, the use of the cycle operator is crucial as it allows us to loop at the decision point. Finally, note that, since the system is required to loop on a decision point from which it is possible to release the resource for either process 1 or process 2, this automatically implies the existence of an infinite path which is able to satisfy this lockout freedom condition, which is expressible in $\mathrm{CTL}^{\star}$ by means of the formula $\psi = \mathtt{E}(\mathtt{GFres}_1 \wedge \mathtt{GFres}_2)$. In other words, we have that $\varphi_1 \wedge \varphi_2 \rightarrow \psi$ is a valid $\mathrm{CTL}^{\star}_{\circlearrowleft}$ formula.

We now discuss another example involving prompt parity games, introduced in [26].

A Parity Game is a tuple of the form $\mathcal{P} = \langle \mathrm{V}, \mathrm{V}_0, \mathrm{V}_1, E, \mathsf{p}, v_0 \rangle$ where V is a nonempty finite set of states of the game, partitioned into V_0 and V_1, being the set belonging to Player 0 and Player 1, respectively, $E \subseteq \mathrm{V} \times \mathrm{V}$ is an edge relation, $\mathsf{p} : \mathrm{V} \rightarrow \mathbb{N}$ is a priority labeling function, assigning a natural number to each state, and $v_0 \in \mathrm{V}$ is a designated initial state. The game is played starting from v_0. At each state v of the game, if $v \in \mathrm{V}_i$, then Player i move to an E-successor of v. Such operation induces an infinite path π over V called play and then, by means of the function p, we also consider the infinite path $\mathsf{p}(\pi)$. Every occurrence of an odd priority on $\mathsf{p}(\pi)$ is called request. For any request, the successive occurrence of an even and greater priority is its response. We say that Player 0 wins the play π under the *parity condition* if every request occurring infinitely often is responded. Moreover, we say that Player 0 wins the play π under the *prompt parity condition* if there exists a natural number n such that each request occurring infinitely often is responded in less than n steps. For both the above cases, we say that Player 1 wins the game iff Player 0 does not win. A strategy for Player i is a function $\mathsf{f}_i : \mathrm{V}^* \cdot \mathrm{V}_i \rightarrow \mathrm{V}$ assigning an E-successor to each partial (finite) path of the game. Clearly, a pair of strategies f_0 and f_1 determines a unique path and therefore, the winner. A strategy f_i is *positional* if, for all partial paths ρ and ρ', with $\mathsf{lst}(\rho) = \mathsf{lst}(\rho') \in \mathrm{V}_i$, it holds that $\mathsf{f}_i(\rho) = \mathsf{f}_i(\rho')$.

Let $\mathcal{P}$ be a parity game and f_0 be a positional strategy for Player 0. By projecting the strategy on the arena, we obtain a KS $\mathcal{K}_{\mathcal{P},\mathsf{f}_0} = \langle \mathrm{AP}, \mathrm{W}, R, \mathsf{L}, w_0 \rangle$ defined as follows: $\mathrm{AP} = \mathsf{rng}(\mathsf{p}) = [0, n]$ [1], for some $n \in \mathbb{N}$, $\mathrm{W} = \mathrm{V}$,

[1] *W.l.o.g.*, we can assume that the range of a priority function is of the form $[0, n]$.

$R = \mathsf{f}_{\mathrm{o}} \cup E \cap (\mathrm{V}_{1} \times \mathrm{V})$, $\mathsf{L}(w) = \{\mathsf{p}(w)\}$, for all $w \in \mathrm{W}$, and $w_I = v_{\mathrm{o}}$. We can express that f_{o} is winning for Player 0 by means of the formula $\varphi^{par} = \mathtt{A}(\bigvee_{k \equiv_2 0}(\mathtt{GF}k \wedge \bigwedge_{l \geq k}^{l \equiv_2 1} \mathtt{FG}\neg l))$. Indeed, the formula asserts that, for all possible paths, there exists an even priority k occurring infinitely often such that each odd priority l greater than k occurs finitely many times. Hence, we have that f_{o} is winning over $\mathcal{P}$ iff $\mathcal{K}_{\mathcal{P},\mathsf{f}_{\mathrm{o}}}, v_{\mathrm{o}} \models \varphi^{par}$.

In addition to this, we can express the existence of a path violating the prompt condition by means of the formula $\varphi^{npmt} = \bigvee_{k \equiv_2 1} \mathtt{E}(\mathtt{GF}k \wedge \mathtt{G}(k \rightarrow (\bigwedge_{l \geq k}^{l \equiv_2 0} \neg l)\mathtt{U}(\mathtt{E}^{\circlearrowleft} \bigwedge_{l \geq k}^{l \equiv_2 0} \mathtt{G}\neg l)))$. Intuitively, φ^{npmt} holds if there exists an odd priority k occurring infinitely often along a path such that before any possible response there exists a cycle departing from a node of the path in which no response can be found. This loop can be used by Player 1 to avoid that Player 0 can reply to priority k within a fixed amount of time. At this point, the formula $\varphi^{par} \rightarrow \varphi^{npmt}$ is able to express the existence of a winning strategy for Player 1 under the prompt parity condition.

3. Model-Theoretic Properties

This section consists of two parts. First, we present invariance properties of $\mathrm{CTL}^{\star}_{\circlearrowleft}$. As trees do not contain any cycle and bisimulation do not preserve cycles, it does not come at a surprise that $\mathrm{CTL}^{\star}_{\circlearrowleft}$ is not invariant under bisimulation and does not have a tree-model property. Therefore, we introduce a new notion of bisimulation namely *cycle-bisimulation*, which takes cycles into account. We prove that $\mathrm{CTL}^{\star}_{\circlearrowleft}$ is invariant under cycle-bisimulation. Using that property, we show that $\mathrm{CTL}^{\star}_{\circlearrowleft}$ has a tree-like model property.

In the second part of the section, we investigate the expressive power of $\mathrm{CTL}^{\star}_{\circlearrowleft}$. We show that $\mathrm{CTL}^{\star}_{\circlearrowleft}$ strictly extends $\mathrm{CTL}^{\star}$ and is orthogonal to the μCalculus.

Invariance Properties We start by establishing that $\mathrm{CTL}^{\star}_{\circlearrowleft}$ is not invariant under bisimulation and does not have a tree-model or finite-model property.

Theorem 1 ($\mathrm{CTL}^{\star}_{\circlearrowleft}$ Negative Model Properties)**.** $\mathrm{CTL}^{\star}_{\circlearrowleft}$ *has neither the finite-model property, nor the tree-model property. It is also not invariant under bisimulation.*

Proof. Consider the formula $\varphi_1 = \mathtt{AG}\neg\mathtt{E}^{\circlearrowleft}\top$ stating that all paths starting from the initial state, do not contain any cycle. This formula is satisfiable.

However, since the transition relation is such that each state has a successor, φ_1 can only be true in an infinite model.

Consider now the formula $\varphi_2 = \mathtt{E}^{\circlearrowleft}\top$. It is true in a model iff its initial state is the first point of a cycle. So φ_2 is satisfiable but is never true at the root of a tree. Hence, $\mathrm{CTL}^{\star}_{\circlearrowleft}$ does not have the tree-model property and thus, is not invariant under bisimulation. □

Definition 8 (Cycle-Bisimulation). *Let* $\mathcal{K}_1 = \langle \mathrm{AP}_1, \mathrm{W}_1, R_1, \mathsf{L}_1, w_0^1 \rangle$ *and* $\mathcal{K}_2 = \langle \mathrm{AP}_2, \mathrm{W}_2, R_2, \mathsf{L}_2, w_0^2 \rangle$ *be two Kripke structures. Then, a relation* $B \subseteq \mathrm{W}_1 \times \mathrm{W}_2$ *is a* cycle-bisimulation relation *if the following hold:*

1. (w_I^1, w_I^2) *belongs to* B*;*
2. *for all* $w_1 \in \mathrm{W}_1$ *and* $w_2 \in \mathrm{W}_2$*, if* (w_1, w_2) *belongs to* B*, then:*
   (a) $\mathsf{L}_1(w_1) = \mathsf{L}_2(w_2)$*;*
   (b) *for all* $v_1 \in \mathrm{W}_1$ *such that* $(w_1, v_1) \in R_1$*, there is* $v_2 \in \mathrm{W}_2$ *such that* $(w_2, v_2) \in R_2$ *and* $(v_1, v_2) \in B$*;*
   (c) *for all* $v_2 \in \mathrm{W}_2$ *such that* $(w_2, v_2) \in R_2$*, there is* $v_1 \in \mathrm{W}_1$ *such that* $(w_1, v_1) \in R_1$ *and* $(v_1, v_2) \in B$*;*
   (d) *for all cycles* π_1 *with beginning state* w_1*, there is a cycle* π_2 *with beginning state* w_2 *such that for all* $i \in \mathbb{N}$*, the pair* $((\pi_1)_i, (\pi_2)_i)$ *belongs to* B*,*
   (e) *for all cycles* π_2 *with beginning state* w_2 *there is a cycle* π_1 *with beginning state* w_1 *such that for all* $i \in \mathbb{N}$*, the pair* $((\pi_1)_i, (\pi_2)_i)$ *belongs to* B*.*

We say that $\mathcal{K}_1$ *and* $\mathcal{K}_2$ *are* cycle-bisimilar w.r.t. *a relation* $B \subseteq \mathrm{W}_1 \times \mathrm{W}_2$ *if* B *is a cycle bisimulation. Moreover, two paths* π_1 *and* π_2 *are* bisimilar w.r.t. *a cycle-bisimulation* B *if for all* $i \in \mathbb{N}$*, the pair* $((\pi_1)_i, (\pi_2)_i)$ *belongs to* B*.*

The notion of cycle-bisimulation is quite intuitive. While the usual definition of a bisimulation allows us to "mimic" the transition relation from one model to the other, a cycle-bisimulation also ensures that we can "mimic" cycles from one model to the other.

As a remark, the cycle-bisimulation notion is interesting by itself, as it gives rise to a new notion of equivalence among structures, that might lead to model-reduction characterization of the logic. We plan to investigate this aspect in a future work.

Theorem 2 (Invariance under cycle-bisimulation)**.** $\mathrm{CTL}^{\star}_{\circlearrowleft}$ *is invariant under cycle-bisimulation.*

Proof. Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be two models and let B be a cycle-bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$. We have to prove that $\mathcal{K}_1 \models \varphi$ iff $\mathcal{K}_2 \models \varphi$, for all formulas φ in $\mathrm{CTL}^{\star}_{\circlearrowleft}$. We prove by induction on φ and ψ that:

- for all pairs $(w_1, w_2) \in B$, $\mathcal{K}_1, w_1 \models \varphi$ iff $\mathcal{K}_2, w_2 \models \varphi$;
- for all paths π_1 in $\mathcal{K}_1$ and all paths π_1 in $\mathcal{K}_2$ such that π_1 and π_2 are bisimilar *w.r.t.* B, $\mathcal{K}_1, \pi_1 \models \psi$, iff $\mathcal{K}_2, \pi_2 \models \psi$.

We only treat the cases of the induction that are not similar to the proof of the invariance of $\mathrm{CTL}^{\star}$ under bisimulation. We also restrict ourselves to the formulas of the form $\mathtt{E}^{\circlearrowleft}\psi$, as the case for the formulas of the form $\mathtt{A}^{\circlearrowleft}\psi$ is symmetric.

Consider a formula $\mathtt{E}^{\circlearrowleft}\psi$ in $\mathrm{CTL}^{\star}_{\circlearrowleft}$. We show that, for all $(w_1, w_2) \in B$, it holds that $\mathcal{K}_1, w_1 \models \mathtt{E}^{\circlearrowleft}\psi$ iff $\mathcal{K}_2, w_2 \models \mathtt{E}^{\circlearrowleft}\psi$. We prove only the direction from left to right. Suppose that for a pair (w_1, w_2) in B, we have $\mathcal{K}_1, w_1 \models \mathtt{E}^{\circlearrowleft}\psi$. That is, there is a cycle π_1 with starting state w_1 such that $\mathcal{K}_1, \pi_1 \models \psi$. By definition of a cycle-bisimulation, this means that there is a cycle π_2 with beginning state w_2, that is bisimilar to π_1 *w.r.t.* B. By induction hypothesis, since $\mathcal{K}_1, \pi_1 \models \psi$, this implies that $\mathcal{K}_2, \pi_2 \models \psi$. Together with the fact that π_2 is a cycle with beginning state w_2, this means that $\mathcal{K}_2, w_2 \models \mathtt{E}^{\circlearrowleft}\psi$, which finishes the proof. □

Using the invariance under cycle-bisimulation, we establish a tree-like model property for $\mathrm{CTL}^{\star}_{\circlearrowleft}$. Intuitively, the tree-model property for $\mathrm{CTL}^{\star}_{\circlearrowleft}$ fails as trees do not admit any cycle. Hence, the idea is to consider structures obtained by adding some restricted form of cycles over trees. We call those structures *trees with back edges* and they are defined as follows.

Definition 9. *A Kripke model* $\mathcal{K} = \langle \mathrm{AP}, \mathrm{W}, R, \mathsf{L}, w_{\mathrm{o}} \rangle$ *is a* tree with back edges *if there are a Kripke model* $\mathcal{T}_{\mathrm{o}} = (\mathrm{AP}, \mathrm{W}, R_{\mathrm{o}}, \mathsf{L}, w_{\mathrm{o}})$ *and a partial map* $\mathsf{f} : \mathrm{W} \rightharpoonup \mathrm{W}$ *such that*

(i) $(\mathrm{W}, R_{\mathrm{o}})$ *is a tree with root* w_{o} *over the alphabet* [2] AP*,*

[2]The relation R_{o} is the child relation of the tree.

(ii) R *is equal to* $R_o \cup \{(w, \mathsf{f}(w)) : w$ *belongs to the domain of* $\mathsf{f}\}$,

(iii) for all $w \in \mathrm{W}$, $\mathsf{f}(w)$ *is an ancestor of* w,

(iv) for all $w_1, w_2 \in \mathrm{W}$, *if* $\mathsf{f}(w_1)$ *is defined,* $(\mathsf{f}(w_2), w_1), (w_1, w_2) \in {R_o}^+$ [3], *then* $\mathsf{f}(w_1) = \mathsf{f}(w_2)$.

We say that $(\mathcal{T}_o, \mathsf{f})$ *is a* tree decomposition *of* $\mathcal{K}$, *where* $\mathcal{T}_o$ *is the* associated tree *and* f *is the* back-edge map. *If a pair* (w, v) *belongs to* R_o, *we say that* (w, v) *is associated with a* forward edge, *while if* $v = f(w)$, *the pair* (w, v) *is associated with a* back edge.

Note that if for every pair (w, v) in R we know whether (w, v) is associated with a forward or back edge, then this uniquely defines a tree decomposition.

Intuitively, a tree with back edges is a structure obtained from a tree by adding edges (called back edges) from some nodes to their ancestors. More precisely, we add a back edge from each node w in the domain of f to its image $\mathsf{f}(w)$. Such back edges need to satisfy two conditions. First, each node must admit at most one outgoing back edge. The second condition (condition (iv)) is a bit less intuitive. It requires that the partial map f preserves the ancestor relation, and, in addition, that the back edges cannot overlap, that is, in a tree back edges never cross each other.

We prove now the tree-like model property and show that each satisfiable formula of $\mathrm{CTL}^\star_{\circlearrowleft}$ is satisfiable in a tree with back edges. More specifically, given a Kripke model $\mathcal{K}$, we show how to define a tree with back edges $\mathcal{U}_\mathcal{K}$ such that $\mathcal{K}$ and $\mathcal{U}_\mathcal{K}$ are cycle-bisimilar. Together with Theorem 2, this implies that each satisfiable formula of $\mathrm{CTL}^\star_{\circlearrowleft}$, is satisfiable in a tree with back edges. Before defining $\mathcal{U}_\mathcal{K}$, we need to introduce two preliminaries notions: the projection map and the initial cycle state.

Let $\mathcal{K} = \langle \mathrm{AP}, \mathrm{W}, R, \mathsf{L}, w_o \rangle$ be a Kripke model and consider two constants $\mathtt{nw}$ and $\mathtt{cy}$. We define the *projection map* $\mathtt{pr} : (\mathrm{W} \times \{\mathtt{nw}, \mathtt{cy}\})^* \to \mathrm{W}$ as the unique surjective map such that $\mathtt{pr}(\epsilon) = w_o$ and for all $w^\bullet \neq \epsilon$, we have $\mathtt{pr}(w^\bullet) = w$, where $\mathsf{lst}(w^\bullet) = (w, \alpha)$ and $\alpha \in \{\mathtt{nw}, \mathtt{cy}\}$.

Given a state $w^\bullet \in (\mathrm{W} \times \{\mathtt{nw}, \mathtt{cy}\})^*$, we say that $w^\bullet$ admits a sequence $v^\bullet$ as an *initial cycle state* if there is a sequence $v_1 \dots v_k$ such that $w^\bullet$ is equal to $v^\bullet\ (v_1, \mathtt{nw})(v_2, \mathtt{cy}) \dots (v_k, \mathtt{cy})$. Given a sequence $w^\bullet \in (\mathrm{W} \times \{\mathtt{nw}, \mathtt{cy}\})^*$ such

[3] As usual, R_o^+ is the transitive closure of R_o and is the ancestor relation of the tree (W, R_o).

that $\mathsf{lst}(w^\bullet) = (w, \alpha)$, we say that $w^\bullet$ is labeled by w and α. Intuitively, the initial cycle state of a given state $w^\bullet$ is simply the parent of the closest ancestor of $w^\bullet$ that is labeled by $\mathtt{nw}$. Note that a state admits at most one initial cycle state. We are now ready to define $\mathcal{U}_\mathcal{K}$.

Definition 10. *Given a Kripke model* $\mathcal{K} = \langle \mathrm{AP}, \mathrm{W}, R, \mathsf{L}, w_\mathrm{o} \rangle$*, we define the* tree-like unwinding $\mathcal{U}_\mathcal{K} = \langle \mathrm{AP}, \mathrm{W}^\bullet, R^\bullet, \mathsf{L}^\bullet, w_\mathrm{o}^\bullet \rangle$ *of* $\mathcal{K}$ *in the following way:*

- $\mathrm{W}^\bullet = (\mathrm{W} \times \{\mathtt{nw}, \mathtt{cy}\})^*$*;*
- $w_\mathrm{o}^\bullet = \epsilon$*;*
- *for all* $w^\bullet \in \mathrm{W}^\bullet$*, we have* $\mathsf{L}^\bullet(w^\bullet) = \mathsf{L}(\mathtt{pr}(w^\bullet))$*;*
- *for all* $w^\bullet \in \mathrm{W}^\bullet$ *and for all* $(\mathtt{pr}(w^\bullet), v) \in R$*:*
  - *the pair* $(w^\bullet, w^\bullet(v, \mathtt{nw}))$ *belongs to* $R^\bullet$ *and is associated with a forward edge;*
  - *if* $w^\bullet$ *admits an initial cycle state* $u^\bullet$ *such that* $\mathtt{pr}(u^\bullet) \neq v$*, then the pair* $(w^\bullet, w^\bullet(v, \mathtt{cy}))$ *belongs to* $R^\bullet$ *and is associated with a forward edge;*
  - *if* $w^\bullet$ *admits an initial cycle state* $u^\bullet$ *such that* $\mathtt{pr}(u^\bullet) = v$*, then the pair* $(w^\bullet, u^\bullet)$ *belongs to* $R^\bullet$ *and is associated with a back edge.*

As mentioned earlier, knowing which edges are forward edges or back edges, uniquely determines a tree decomposition. We denote by $(\mathcal{T}_\mathrm{o}(\mathcal{K}), \mathsf{f}(\mathcal{K}))$ *the tree decomposition associated with the above definition.*

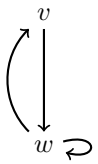

Figure 1: The Kripke Model $\mathcal{K}_\mathrm{o}$.

Note that ϵ is the only state of $\mathcal{U}_\mathcal{K}$ that does not admit any initial cycle state. It follows from the definition of $R^\bullet$ that all the successors of ϵ in $\mathcal{U}_\mathcal{K}$ are of the form $(w, \mathtt{nw})$ (where w is a successor in $\mathcal{K}$ of the initial state of $\mathcal{K}$). Intuitively, the tree with back edges $\mathcal{U}_\mathcal{K}$ is defined as follows. We consider the usual unwinding construction [4] of a Kripke model and we modify it in two steps.

[4] That is, the Kripke model with domain $\{\rho : \rho \text{ is a track in } \mathcal{K}\}$, initial state ϵ, transition relation $\{(\rho, \rho w) : \rho \text{ and } \rho \cdot w \text{ are tracks in } \mathcal{K}\}$ and a labeling function mapping each track ρ to the set $\mathsf{Llst}(\rho)$.

First, in the unwinding construction, given a track ρ with $\mathsf{lst}(\rho) = w$ and given a pair (w, v) in the transition relation R, we construct *one successor* of ρ of the form $\rho \cdot v$. Here, we make two "copies" of the successor $\rho \cdot v$, one labeled by `nw` and the other one labeled by `cy`.

The second modification is as follows: we delete certain edges and replace them with back edges (and finally, delete all the states that are not reachable from ϵ). An edge from track ρ_1 to ρ_2 is deleted iff ρ_2 is labeled by `cy` and ρ_2 and the initial cycle state of ρ_1 are labeled by the same state of $\mathcal{K}$.

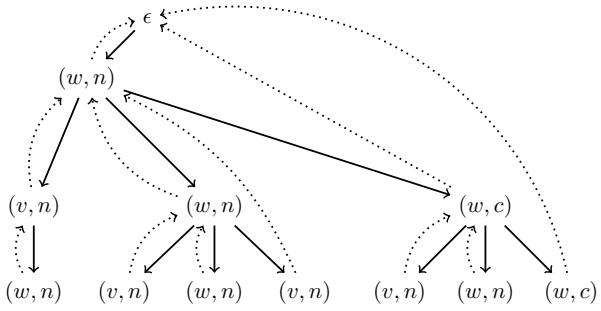

Figure 2: The tree with back edges $\mathcal{T}(\mathcal{K}_{\mathrm{o}})$.

It is easy to see that the tree-like unwinding of a Kripke structure fulfills Definition 9, and so it is a tree with back edges. Indeed, note that item *(i)*, *(ii)*, and *(iii)* of the definition are trivially satisfied. Regarding item *(iv)*, note that every state $w^\bullet$ is mapped backward only to its corresponding initial cycle state $v^\bullet$ and, for every state $z^\bullet$ in between $v^\bullet$ and $w^\bullet$, *i.e.* such that $(v^\bullet, z^\bullet) \in \mathrm{R}_{\mathrm{o}}$ and $(z^\bullet, w^\bullet) \in \mathrm{R}_{\mathrm{o}}$, it holds that $v^\bullet$ is its corresponding initial cycle state. Hence, the condition at item *(iv)* holds.

In order to illustrate the construction $\mathcal{U}_{\mathcal{K}}$, we provide an example in Figure 1 and Figure 2. To make notation easier in the figure, we abbreviate `nw` by `n` and `cy` by `c`. Also, instead of writing ρ for a state, we only write the pair of labels $\mathsf{lst}(\rho)$. The back edges are those that are not straight lines.

Theorem 3. $\mathrm{CTL}^\star_{\circlearrowleft}$ *has a tree-like model property. Every satisfiable formula of* $\mathrm{CTL}^\star_{\circlearrowleft}$ *is satisfiable in a tree with back edges.*

This follows immediately from the following proposition.

Proposition 1. *Let* $\mathcal{K} = \langle \mathrm{AP}, \mathrm{W}, R, \mathsf{L}, w_{\mathrm{o}} \rangle$ *be a Kripke model and* $\mathcal{U}_{\mathcal{K}} = \langle \mathrm{AP}^\bullet, \mathrm{W}^\bullet, R^\bullet, \mathsf{L}^\bullet, w^\bullet{}_{\mathrm{o}} \rangle$ *its tree-like unwinding. Then, the relation* $\{(w^\bullet, \mathtt{pr}(w^\bullet)) : w^\bullet \in \mathrm{W}^\bullet\}$ *is a cycle-bisimulation. Hence,* $\mathcal{K}$ *and* $\mathcal{U}_{\mathcal{K}}$ *satisfy exactly the same formulas in* $\mathrm{CTL}^\star_{\circlearrowleft}$.

Proof. Let B be the relation $\{(w^\bullet, \mathtt{pr}(w^\bullet)) : w^\bullet \in \mathrm{W}^\bullet\}$. The pair $(\epsilon, w_{\mathrm{o}})$ belongs to B. Next, consider a pair $(w^\bullet, w)$ in B. We have to check that it satisfies conditions 2a, 2b, 2c, 2d and 2e from Definition 8.

The fact that 2a is satisfied follows immediately from the definition of $\mathcal{U}_\mathcal{K}$. It also follows from the definition of $\mathtt{pr}$ and $\mathcal{U}_\mathcal{K}$ that if a pair $(w^\bullet, v^\bullet)$ belongs to $R^\bullet$, then $(\mathtt{pr}(w^\bullet), \mathtt{pr}(v^\bullet))$ belongs to R. In particular, condition 2b is satisfied.

For condition 2c, let $(\mathtt{pr}(w^\bullet), u)$ be a pair in R. We define $u^\bullet$ as the sequence $w^\bullet(u, \mathtt{nw})$. We have that $\mathtt{pr}(u^\bullet) = u$. Hence, the pair $(u^\bullet, u)$ belongs to B. It also follows from the definition of $R^\bullet$ that $(w^\bullet, u^\bullet)$ belongs to $R^\bullet$. Hence, condition 2c is satisfied.

For condition 2d, let $\pi^\bullet$ be a cycle with beginning state $w^\bullet$. By definition of $R^\bullet$ and B, the path $\mathtt{pr}(\pi^\bullet)$ is a cycle with initial node $\mathtt{pr}(w^\bullet) = w$, that is bisimilar *w.r.t.* B to $\pi^\bullet$. Hence, 2d holds.

Finally, for 2e, consider a cycle π with beginning state w. Assume that $\pi = w_0 w_1 \dots$. Since π is a cycle, there is an infinite sequence $(n_i)_{i \geq 0}$ such that $n_0 = 0$ and for all $i \in \mathbb{N}$, $n_i + 1 < n_{i+1}$ and $w_{n_i} = w, \mathrm{N}$, for all $n_i < j < n_{i+1}, w_j \neq w. \notin \{n_i : i \geq 0\}$.

Given a number $j \in \mathbb{N}$, we define $\mathtt{cl}(j)$ as the greatest number n_i such that $n_i \leq j$. We are now ready to define the path $\pi^\bullet$ as the path $(w^\bullet_j)_{j \geq 0}$, where for all $j \in \mathbb{N}$,

$$w^\bullet_j = \begin{cases} w^\bullet, & \text{if } j = n_i \text{ for some } i \in \mathbb{N}, \\ w^\bullet(w_j, \mathtt{nw}), & \text{if } j = \mathtt{cl}(j) + 1, \\ w^\bullet(w_{\mathtt{cl}(j)+1}, \mathtt{nw})(w_{\mathtt{cl}(j)+2}, \mathtt{cy}) \dots (w_j, \mathtt{cy}), & \text{otherwise.} \end{cases}$$

Note that $\mathtt{pr}(w^\bullet_j) = w_j$. Intuitively, the path $\pi^\bullet$ is defined as follows. We start at the state $w^\bullet$. The next state is the successor labeled by w_1 and $\mathtt{nw}$. Next, we will only choose successors (until we go back to $w^\bullet$) that are labeled by $\mathtt{cy}$. This ensures that for each chosen state, its initial cycle state is $w^\bullet$. Once we have gone down the tree for $n_1 - 1$ steps (and reached a node labeled by $w_{n_1 - 1}$ and $\mathtt{cy}$), we go back to the state $w^\bullet$. We keep iterating the procedure.

Now we prove that $\pi^\bullet$ is a cycle of $\mathcal{U}_\mathcal{K}$, that is bisimilar to π. Since $(\pi^\bullet)_{n_i} = w^\bullet$ for all $i \in \mathbb{N}$, we know that the path $\pi^\bullet$ goes infinitely often through the state $w^\bullet$. Moreover, since $\mathtt{pr}(w^\bullet_j) = w_j$, the path $\pi^\bullet$ is bisimilar to π.

Hence, it remains to prove that $\pi^\bullet$ is indeed a path of $\mathcal{U}_\mathcal{K}$. That is, for all $j > 0$, the pair $(w^\bullet_{j-1}, w^\bullet_j)$ belongs to the relation $R^\bullet$. The difficult case is when $j = n_{i_0}$ for some $i_0 \in \mathbb{N}$. So we have to show that $(w^\bullet_{n_{i_0} - 1}, w^\bullet_{n_{i_0}})$ belongs to $R^\bullet$. By definition, the state $w^\bullet_{n_{i_0}}$ is equal to $w^\bullet$. Hence, we

have to prove that $(w^\bullet_{n_{i_o}-1}, w^\bullet)$ belongs to $R^\bullet$. It is enough to show that there is back edge in $\mathcal{U}_\mathcal{K}$ from $w^\bullet_{n_{i_o}-1}$ to $w^\bullet$. By definition of $\mathcal{U}_\mathcal{K}$, this is equivalent to prove that *(i)* the pair $(\mathtt{pr}(w^\bullet_{n_{i_o}-1}), \mathtt{pr}(w^\bullet))$ belongs to R and *(ii)* $w^\bullet$ is the initial cycle state of $w^\bullet_{n_{i_o}-1}$. We start by *(i)*. We know that $\mathtt{pr}(w^\bullet_{n_{i_o}-1}) = w_{n_{i_o}-1}$ and $\mathtt{pr}(w^\bullet) = w$. So we have to prove that $(w_{n_{i_o}-1}, w)$ belongs to R. Since $w_{n_{i_o}} = w$ (by definition of the n_is), this is equivalent to show that $(w_{n_{i_o}-1}, w_{n_{i_o}})$ belongs to R. This follows from the fact that $(w_j)_{j\geq 0}$ is a path in $\mathcal{K}$.

We finish the proof by showing *(ii)*. It follows from the definition of $\pi^\bullet$ that $w^\bullet_{n_{i_o}-1}$ is equal to $w^\bullet(u_1, \mathtt{nw})(u_2, \mathtt{cy}) \dots (u_k, \mathtt{cy})$ for some sequence $u_1 \dots u_k$. Hence, $w^\bullet$ is the initial cycle state of $w^\bullet_{n_{i_o}-1}$. □

Before finishing the section on model properties, we state one more property concerning the tree-like unwinding of a model. It states that if a formula is true in a tree-like unwinding, then we may assume the "witness" cycles (for the subformulas of the form $\mathtt{E}^{\circlearrowleft}\psi$) to be simple cycles. This means that, from the satisfiability point of view, the operator $\mathtt{E}^{\circlearrowleft}$ can be replaces with its simple-cycle version $\mathtt{E}^{\circlearrowleft}_s$. The property will play an important role in the next section for obtaining a 2ExpTime upper-bound for the satisfiability problem of the existential fragment of $\mathrm{CTL}^\star_{\circlearrowleft}$.

Proposition 2. *Let φ be a formula in $\mathrm{CTL}^\star_{\circlearrowleft}$ in normal form and let $\mathcal{K}$ be a Kripke model. Then $\mathcal{K} \models \varphi$ iff $\mathcal{U}_\mathcal{K} \models (\varphi)_s$, where $\mathcal{U}_\mathcal{K}$ is the tree with back edges as in Definition 9.*

Proof. By induction on the structure of the formula, we prove that $\mathcal{K}, w \models \phi$ if and only if $\mathcal{U}_\mathcal{K}, w^\bullet \models (\varphi)_s$, where $w^\bullet$ is any state in $\mathcal{U}_\mathcal{K}$ corresponding to w. We show here only the case of $\mathtt{E}^{\circlearrowleft}\psi$, as all the other cases are trivial. Let us assume $\varphi = \mathtt{E}^{\circlearrowleft}\psi$ is such that $\mathcal{K}, w \models \mathtt{E}^{\circlearrowleft}\psi$. Then, by the definition of semantics, there exists a cycle path π in $\mathrm{Pth}_\mathcal{K}$ such that $\mathcal{K}, \pi \models \psi$. Now, consider the path $\pi^\bullet$ in $\mathcal{U}_\mathcal{K}$ inductively defined from π as follows. For $n = 0$, we define $(\pi^\bullet)_0 = w^\bullet$. For every $n \geq 0$, if $\pi_{n+1} = w$, then $(\pi^\bullet)_{n+1} = (w, \mathtt{nw})$, otherwise, define $(\pi^\bullet)_{n+1} = \pi_{\leq n} \cdot (\pi_{n+1}, \mathtt{cy})$. Now, observe that $\pi^\bullet$ is a simple cycle. Indeed, the initial state w occurs infinitely often along $\pi^\bullet$. Moreover, for every two consecutive occurrences of w, the states in between are strictly ordered according to the prefix relation, and so there are no repeating ones. Finally, it is immediate to see that $\pi^\bullet$ is cycle-bisimilar to π. Thus, by means of Theorem 1, it holds that $\mathcal{U}_\mathcal{K}, \pi^\bullet \models \psi$. □

Expressiveness We now investigate the expressive power of $\mathrm{CTL}^\star_{\circlearrowleft}$ *w.r.t.* the usual temporal logics. All the results are collected in the following theorem.

Theorem 4 (Expressiveness comparison)**.** $\mathrm{CTL}^\star_{\circlearrowleft}$ *is strictly more expressive than* $\mathrm{CTL}^\star$ *and is incomparable with the* μCALCULUS.

Proof. First we show that $\mathrm{CTL}^\star$ is strictly less expressive than $\mathrm{CTL}^\star_{\circlearrowleft}$. We observed in the proof of Theorem 1 that the formula $\varphi_1 = \mathtt{AG}\neg\mathtt{E}^{\circlearrowleft}\top$ is satisfiable but does not admit any finite model. Since $\mathrm{CTL}^\star$ has the finite-model property, this implies that φ_1 is a formula in $\mathrm{CTL}^\star_{\circlearrowleft}$, which is not equivalent to any formula in $\mathrm{CTL}^\star$.

Now, we prove that $\mathrm{CTL}^\star_{\circlearrowleft}$ is incomparable with the μCALCULUS. First, since the μCALCULUS has the finite model property, we also have that φ_1 is a formula in $\mathrm{CTL}^\star_{\circlearrowleft}$ that is not equivalent to any formula in $\mathrm{CTL}^\star$. Next, we show that there is a formula in the μCALCULUS that is not expressible in $\mathrm{CTL}^\star_{\circlearrowleft}$. Consider the formula $\psi = \nu x.p \wedge \Box\Box x$. The formula ψ is true in a model if for all paths π starting from the initial state, p is true in every even state $(\pi)_{2i}$ of the path π. In particular, ψ is true in an LTL model if p is true in every even state.

We show that ψ is not equivalent to any formula in $\mathrm{CTL}^\star_{\circlearrowleft}$. Suppose for contradiction that there is a formula χ in $\mathrm{CTL}^\star_{\circlearrowleft}$ that is equivalent to ψ. The idea is to transform χ into a formula $\mathtt{t}(\chi)$ in LTL such that χ and $\mathtt{t}(\chi)$ are equivalent over LTL models. We will do that in the next paragraph, but first we show how the existence of such a formula $\mathtt{t}(\chi)$ is sufficient to derive a contradiction. So let $\mathtt{t}(\chi)$ be a formula in LTL such that χ and $\mathtt{t}(\chi)$ are equivalent over LTL models. Recall that ψ is true in an LTL model if p is true in every even state. In particular, $\mathtt{t}(\chi)$ is a formula of LTL such that ψ is true in an LTL model if p is true in every even state. However, it is known that there is no LTL formula defining the class of LTL models in which p is true in every even state [30]. We obtained the desired contradiction.

Now, it remains to define the formula $\mathtt{t}(\chi)$ in LTL such that χ and $\mathtt{t}(\chi)$ are equivalent over LTL models. The formula $\mathtt{t}(\chi)$ is obtained by deleting all the occurrences of $\mathtt{A}$ and $\mathtt{E}$ and by replacing all the occurrences of the symbol $\circlearrowleft$ by $\bot$. The fact that χ and $\mathtt{t}(\chi)$ are equivalent over LTL model follows from the fact that $\circlearrowleft$ is never true in a state of an LTL model and the fact that for all LTL formulas χ_0, the formulas $\mathtt{A}\chi_0$ and $\mathtt{E}\chi_0$ are equivalent to χ_0 over LTL models.

□

4. Decision Problems

In this section, we deal with the solution of the model-checking and satisfiability problems for $\mathrm{CTL}^{\star}_{\circlearrowleft}$. Regarding the former, we show that we retain the same complexity as for $\mathrm{CTL}^{\star}$, that is PSpace. Concerning satisfiability, we also retain the same complexity of $\mathrm{CTL}^{\star}$ if we restrict to the existential-cycle fragment of the logic, that is 2ExpTime. Conversely, we show that it is 3ExpTime for the whole logic.

Model Checking For the solution of the model-checking problem of $\mathrm{CTL}^{\star}_{\circlearrowleft}$, we employ a standard bottom-up procedure on the nesting of the path quantifiers of the specification under exam, which extends the one originally proposed for $\mathrm{CTL}^{\star}$ [10]. With more details, starting from the innermost state formulas φ of the kind $\mathtt{E}\psi$, $\mathtt{A}\psi$, $\mathtt{E}^{\circlearrowleft}\psi$, and $\mathtt{A}^{\circlearrowleft}\psi$, we determine their truth value over a KS $\mathcal{K}$ at a world $w \in \mathrm{W}$ by checking the emptiness of a suitable nondeterministic Büchi word automaton $\mathcal{N}^{\varphi}_{\mathcal{K},w}$. In case of a positive result, we enrich the labeling of the world w with a fresh proposition φ representing the formula φ itself. Obviously, the path formula ψ is just seen as a classic LTL formula, where all its subformulas of the kind described above are interpreted as atomic propositions whose truth values on the worlds of $\mathcal{K}$ are already computed in some previous step of the algorithm. It is important to observe that the difference between the automata for $\mathtt{E}\psi$ or $\mathtt{A}\psi$ and those for $\mathtt{E}^{\circlearrowleft}\psi$ or $\mathtt{A}^{\circlearrowleft}\psi$ resides in the fact that, for the latter, we have to further verify that the initial state of the path is seen infinitely often. This can be done by means of the standard Büchi acceptance condition. Hence, we directly obtain that the model checking for $\mathrm{CTL}^{\star}_{\circlearrowleft}$ is not more complex than the same problem for $\mathrm{CTL}^{\star}$.

Theorem 5. *The model-checking problem for* $\mathrm{CTL}^{\star}_{\circlearrowleft}$ *is* PSpace-c w.r.t. *the formula complexity and* NLogSpace-c w.r.t. *the data complexity.*

Proof. We just describe the construction of the automaton $\mathcal{N}^{\varphi}_{\mathcal{K},w}$ for the formula $\mathtt{E}^{\circlearrowleft}\psi$, as the verification of the formula $\mathtt{A}^{\circlearrowleft}\psi$ immediately reduces to the one for $\neg\mathtt{E}^{\circlearrowleft}\neg\psi$. Let $\mathcal{K} = \langle \mathrm{AP}, \mathrm{W}, R, \mathtt{L}, w_{\mathrm{o}} \rangle$ be the KS under exam and $\mathcal{N}_{\psi} = \langle 2^{\mathrm{AP}}, \mathrm{Q}, \delta, \mathrm{Q}_I, \mathrm{F} \rangle$ the nondeterministic Büchi word automaton obtained from the LTL ψ by means of the classic Vardi-Wolper construction [29]. We build the automaton $\mathcal{N}^{\star}_{\psi} \triangleq \langle \mathrm{W}, \mathrm{Q}, \delta^{\star}, \mathrm{Q}_I, \mathrm{F} \rangle$ obtained from $\mathcal{N}_{\psi}$ by replacing the alphabet 2^{AP} with the set of worlds of $\mathcal{K}$ via the definition $\delta^{\star}(q, w) \triangleq \delta(q, \mathtt{L}(w))$. By construction, $\mathcal{N}^{\star}_{\psi}$ accepts an infinite word on

W iff its labeling satisfies the property ψ. Now, we need to to force $\mathcal{N}^{\star}_{\psi}$ to only accepts words that are effectively paths in the structure $\mathcal{K}$ passing infinitely often through the initial state w. We do this via the nondeterministic Büchi word automaton $\mathcal{N}_{\mathcal{K},w} \triangleq \langle \mathrm{W}, \mathrm{W}, \delta^{\bullet}, \{w\}, \{w\}\rangle$ with $\delta^{\bullet}(v,v) = R(v)$ and $\delta^{\bullet}(v,v') = \emptyset$, for $v \neq v'$, whose starting and accepting set $\{w\}$ ensures the recognizing of all and only those sequences of worlds that start and pass infinitely often through w. Finally, we construct the required automaton $\mathcal{N}^{\varphi}_{\mathcal{K},w}$ by making the product of $\mathcal{N}^{\star}_{\psi}$ and $\mathcal{N}_{\mathcal{K},w}$. It is easy to see that $\mathcal{K}, w \models \mathtt{E}^{\circlearrowleft}\psi$ iff $\mathcal{N}^{\varphi}_{\mathcal{K},w}$ accepts some word, which results to be a cycle path of $\mathcal{K}$.

As the complexity is concerned, observe that the size of $\mathcal{N}^{\varphi}_{\mathcal{K},w}$ is $\mathrm{O}(|\mathcal{K}| \cdot 2^{|\varphi|})$. Since the emptiness of this automaton can be computed in NLogSpace *w.r.t.* its size, the verification of $\mathcal{K}, w \models \mathtt{E}^{\circlearrowleft}\psi$ can be solved in PSpace *w.r.t.* $|\varphi|$ and in NLogSpace *w.r.t.* $|\mathcal{K}|$. □

For the special case of a $\mathrm{CTL}_{\circlearrowleft}$ formula φ, note that the size of the automaton $\mathcal{N}^{\varphi}_{\mathcal{K},w}$ is $\mathrm{O}(|\mathcal{K}| \cdot |\varphi|)$ [22]. From this and by following the same construction as in Theorem 5, we derive the complexity for solving the model-checking of $\mathrm{CTL}_{\circlearrowleft}$.

Theorem 6. *The model-checking problem for* $\mathrm{CTL}_{\circlearrowleft}$ *is* PTime-c w.r.t. *the formula complexity and* NLogSpace-c w.r.t. *the data complexity.*

We now address the cases of $\mathrm{CTL}^{\star}_{s\circlearrowleft}$ and $\mathrm{CTL}_{s\circlearrowleft}$. Differently from the case of a generic cycle, it is not enough to check whether the initial state occurs infinitely often with the automaton $\mathcal{N}_{\mathcal{K},w}$. This is because, in order to be a simple cycle, there have to be no repetitions of states in between two occurrences of the initial state in the path. This additional check makes the overall complexity to raise, as it is explain in the proof of the following theorem.

Theorem 7. *The two following hold:*

- *The model-checking problem for Simple-Cycle-*$\mathrm{CTL}^{\star}_{\circlearrowleft}$ *is* PSpace-c w.r.t. *the formula complexity and* PSpace-c w.r.t. *the data complexity.*
- *The model-checking problem for* $\mathrm{CTL}^{\star}_{\circlearrowleft}$ *is* PTime-c w.r.t. *the formula complexity and* PSpace-c w.r.t. *the data complexity.*

Proof. The proof of the two items are similar to the ones for Theorem 5 and Theorem 6, respectively. We only need to replace the automaton $\mathcal{N}_{\mathcal{K},w}$

with a more refined one, taking into account the acceptance of simple cycles, instead of cycles. To do this, consider the automaton $\mathcal{N}^{s}_{\mathcal{K},w} \triangleq \langle \mathrm{W}, \mathrm{W} \times 2^{\mathrm{W}}, \delta^{s\bullet}, \{(w,\emptyset)\}, \{w\} \times 2^{\mathrm{W}} \rangle$, with $\delta^{s\bullet}((v,\mathrm{V}),v') = \emptyset$ if $v \neq v'$ and

$$\delta^{s\bullet}((v,\mathrm{V}),v) = \begin{cases} R(v) \times \{\emptyset\}, & \text{if } v = w \\ (R(v) \setminus \mathrm{V}) \times (\mathrm{V} \cup \{v\}), & \text{otherwise} \end{cases},$$

The state space of the automaton is made by two components. The first corresponds to the one given for the automaton $\mathcal{N}_{\mathcal{K},w}$ defined in the proof of Theorem 5, and it is used to keep track of the path executed in the Kripke structure $\mathcal{K}$. The second, intuitively, keeps track of the states that have been visited since the last time the state w has been encountered, and it is reset every time w is seen in the execution. The reader notices that, in case the automaton is in a state $w' \neq w$, the transition relation allows to move from w' to $R(w') \setminus \mathrm{V}$, where V is the visited set. This prevents the accepting runs to visit the same state twice or more times, since the last visit of w, which corresponds to the definition of simple cycle paths. Hence, the automaton $\mathcal{N}^{s}_{\mathcal{K},w}$ recognizes the set $\mathrm{SCyc}(w)$. □

Satisfiability Differently from the model-checking problem, the two introduced looping quantifiers $\mathtt{E}^{\circlearrowleft}\psi$ and $\mathtt{A}^{\circlearrowleft}\psi$ and their simple-cycle counterpart $\mathtt{E}^{\circlearrowleft}_{s}\psi$ and $\mathtt{A}^{\circlearrowleft}_{s}\psi$ heavily affect the satisfiability of $\mathrm{CTL}^{\star}_{\circlearrowleft}$ and $\mathrm{CTL}^{\star}_{s\circlearrowleft}$. In particular, since this logic lacks of the standard tree-model property, we cannot use for the $\mathrm{CTL}^{\star}$ part of $\mathrm{CTL}^{\star}_{\circlearrowleft}$ and $\mathrm{CTL}^{\star}_{s\circlearrowleft}$ the automata approach as proposed in [22]. Instead, here we use symmetric two-way alternating tree automata [5], a simplified version of two-way graded alternating parity tree automata [5], which simply lifts standard (asymmetric) two-way alternating automata over ranked trees [28] to unranked trees, *i.e.*, trees with possibly unbounded width. These are automata that allow to traverse a tree in both forward and backward directions, helping us to search for tree representation of the tree-like unwinding of a structure, as described in the previous section. With more details, for every $\mathrm{CTL}^{\star}_{\circlearrowleft}$ state formula φ, we build an alternating parity two-way tree automaton $\mathcal{A}_{\varphi}$ such that a KS $\mathcal{K} = \langle \mathrm{AP}, \mathrm{W}, R, \mathsf{L}, w_{\mathrm{o}} \rangle$ is a model of φ iff $\mathcal{A}_{\varphi}$ accepts a tree $\mathcal{T}_{\mathcal{K}} = \langle \mathrm{AP} \cup \{\mathtt{nw}, \uparrow\}, \mathrm{W}^{\bullet}, R^{\star}, \mathsf{L}^{\star}, w^{\bullet}_{I} \rangle$ associated with the tree-like unwinding $\mathcal{U}_{\mathcal{K}} = \langle \mathrm{AP}, \mathrm{W}^{\bullet}, R^{\bullet}, \mathsf{L}^{\bullet}, w^{\bullet}_{I} \rangle$ of $\mathcal{K}$ satisfying the following properties: *(i)* $R^{\star} = \{(w^{\bullet}, v^{\bullet}) \in R^{\bullet} : |w^{\bullet}| < |v^{\bullet}|\}$, *(ii)* $\mathsf{L}^{\star}(w^{\bullet}) \cap \mathrm{AP} = \mathsf{L}^{\bullet}(w^{\bullet})$, *(iii)* $\mathtt{nw} \in \mathsf{L}^{\star}(w^{\bullet})$ iff $\mathsf{lst}(w^{\bullet}) = (w, \mathtt{nw})$, for some $w \in \mathrm{W}$, and *(iv)* $\uparrow \in \mathsf{L}^{\star}(w^{\bullet})$ iff there exists $(w^{\bullet}, v^{\bullet}) \in R^{\bullet}$ with $|v^{\bullet}| < |w^{\bullet}|$.

Intuitively, $\mathcal{T}_\mathcal{K}$ is built from $\mathcal{U}_\mathcal{K}$ by deleting all back edges (property *(i)*) and enriching the original labeling of every world $w^\bullet$ (property *(ii)*) with $\mathtt{nw}$, if the last letter of $w^\bullet$ contains the flag with the same name (property *(iii)*), and with $\uparrow$, if $w^\bullet$ is the origin of a back edge (property *(iv)*). It is not hard to see that, for every unwinding $\mathcal{U}_\mathcal{K}$ of a KS $\mathcal{K}$, there exists one and only one tree $\mathcal{T}_\mathcal{K}$ satisfying the previous four properties. Therefore, instead of looking for a model $\mathcal{K}$ of φ or its tree-like unwinding $\mathcal{U}_\mathcal{K}$, we just look for its tree representation $\mathcal{T}_\mathcal{K}$. This idea is at the basis for the automata-theoretic approach described in the proofs of the following theorems.

Theorem 8. *The satisfiability problem for* $\mathrm{CTL}^\star_{\circlearrowleft}$ *can be solved in 3*ExpTime *and is 2*ExpTime-h. *The same problem for* $\mathrm{CTL}_{\circlearrowleft}$ *is* ExpTime-c.

Proof. The 2ExpTime (resp, ExpTime) lower bound for $\mathrm{CTL}^\star_{\circlearrowleft}$ (*resp.*, $\mathrm{CTL}_{\circlearrowleft}$) immediately follows from the one of $\mathrm{CTL}^\star$ (resp, CTL). For the 3ExpTime (*resp.*, ExpTime) upper bound, given a $\mathrm{CTL}^\star_{\circlearrowleft}$ (*resp.*, $\mathrm{CTL}_{\circlearrowleft}$) state formula φ, we reduce the associated satisfiability question to the emptiness problem of a symmetric alternating parity two-way tree automaton $\mathcal{A}_\varphi$, whose size and index are, respectively, doubly and singly exponential (*resp.*, both polynomial) in $|\varphi|$. For a detailed definition of this type of automata and the related concepts of size and index, we refer to [5]. Since the emptiness of $\mathcal{A}_\varphi$ can be checked in time exponential *w.r.t.* both its states and index [5], we obtain the desired result [5].

As mentioned above, $\mathcal{A}_\varphi$ needs to recognize all and only the tree representations $\mathcal{T}_\mathcal{K}$ of the tree-like unwindings $\mathcal{U}_\mathcal{K}$ of KS models $\mathcal{K}$ of φ. As it is usually done for $\mathrm{CTL}^\star$, we slightly weaken this property by allowing $\mathcal{A}_\varphi$ to run on trees that also contain, as labeling of its worlds, the subformulas of φ of the form $\mathtt{E}\psi$, $\mathtt{A}\psi$, $\mathtt{E}^{\circlearrowleft}\psi$, and $\mathtt{A}^{\circlearrowleft}\psi$, which are interpreted as fresh atomic propositions. We denote by $\mathsf{sub}(\varphi)$ the set of subformulas of φ of the form $\mathtt{E}\psi$, $\mathtt{A}\psi$, $\mathtt{E}^{\circlearrowleft}\psi$, and $\mathtt{A}^{\circlearrowleft}\psi$. We also let $\mathsf{sub}^\neg(\varphi)$ be the the closure under negation of the set $\mathsf{sub}(\varphi)$, *i.e.*, for every $\mathtt{E}\psi$ (resp., $\mathtt{A}\psi$, $\mathtt{E}^{\circlearrowleft}\psi$, $\mathtt{A}^{\circlearrowleft}\psi$) in $\mathsf{sub}(\varphi)$, we have $\mathtt{A}\neg\psi$ (resp., $\mathtt{E}\neg\psi$, $\mathtt{A}^{\circlearrowleft}\neg\psi$, $\mathtt{E}^{\circlearrowleft}\neg\psi$) in $\mathsf{sub}^\neg(\varphi)$. So, instead of considering a model $\mathcal{K} = \langle \mathrm{AP}, \mathrm{W}, R, \mathsf{L}, w_{\mathrm{o}}\rangle$ of φ, we work on the enriched KS $\mathcal{K}^\star = \langle \mathrm{AP}^\star, \mathrm{W}, R, \mathsf{L}^\star, w_{\mathrm{o}}\rangle$ such that *(i)* $\mathrm{AP}^\star = \mathrm{AP} \cup \mathsf{sub}^\neg(\varphi)$, *(ii)* $\mathsf{L}^\star(w) \cap \mathrm{AP} = \mathsf{L}(w)$, and

[5] In particular, Theorem 6.7 in [5] can be used for the translation. Observe that, since we do not make use of any graded modalities (our box and diamond symbols stand for $[\![0]\!]$ and $\langle\!\langle 0\rangle\!\rangle$ in their syntax) the resulting automaton is simply a symmetric non-deterministic tree automaton.

(iii) $\eta \in \mathsf{L}^{\star}(w)$ iff $\mathcal{K}, w \models \eta$, for all $\eta \in \mathsf{sub}^{\neg}(\varphi)$, where the latter set denotes the closure under negation of the set $\mathsf{sub}(\varphi)$, *i.e.*, for every $\mathtt{E}\psi$ (resp., $\mathtt{A}\psi$, $\mathtt{E}^{\circlearrowleft}\psi$, $\mathtt{A}^{\circlearrowleft}\psi$) in $\mathsf{sub}(\varphi)$, we have $\mathtt{A}\neg\psi$ (resp., $\mathtt{E}\neg\psi$, $\mathtt{A}^{\circlearrowleft}\neg\psi$, $\mathtt{E}^{\circlearrowleft}\neg\psi$) in $\mathsf{sub}^{\neg}(\varphi)$.

The automaton $\mathcal{A}_{\varphi}$ is built as the conjunction of an automaton $\mathcal{A}_{\eta}$, for every subformula $\eta \in \mathsf{sub}^{\neg}(\varphi)$, and a deterministic safety (*i.e.*, without acceptance condition) automaton $\mathcal{D}_{\varphi}$ used to verify that φ is satisfied at the root of the input tree $\mathcal{T}_{\mathcal{K}}$, when φ is interpreted as a Boolean formula on $\mathrm{AP}^{\star}$. In addition, $\mathcal{A}_{\varphi}$ needs to check that, if a world is not labeled by a state formula $\eta \in \mathsf{sub}^{\neg}(\varphi)$, it is necessarily labeled by a formula $\overline{\eta} \in \mathsf{sub}^{\neg}(\varphi)$ equivalent to its negation, *i.e.*, $\overline{\eta} \equiv \neg\eta$. The automaton $\mathcal{A}_{\eta}$ is committed to check that a world labeled by $\eta \in \mathsf{sub}^{\neg}(\varphi)$ really satisfies this formula. Formally, we have $\mathcal{A}_{\varphi} \triangleq \mathcal{D}_{\varphi} \wedge \bigwedge_{\eta \in \mathsf{sub}^{\neg}(\varphi)} \mathcal{A}_{\eta}$. So, its size is the sum of the sizes of the components. The construction of $\mathcal{D}_{\varphi}$ is trivial. Moreover, the automata for $\mathtt{A}\psi$ and $\mathtt{A}^{\circlearrowleft}\psi$ can be directly derived via dualization from the automaton for $\mathtt{E}\neg\psi$ and $\mathtt{E}^{\circlearrowleft}\neg\psi$ by replacing $\vee$ and $\Diamond$ with $\wedge$ and $\Box$ in their definitions. Hence, we just focus on the constructions for the latter.

We start with the construction of $\mathcal{A}_{\mathtt{E}\psi}$ for $\mathtt{E}\psi$. Consider the nondeterministic Büchi word automaton $\mathcal{N}_{\psi} = \langle 2^{\mathrm{AP}^{\star}}, \mathrm{Q}, \delta, \mathrm{Q}_I, \mathrm{F}\rangle$ obtained by applying the Vardi-Wolper construction to ψ which is read as an LTL formula over $\mathrm{AP}^{\star}$ [29]. We define a two-way Büchi tree automaton $\mathcal{A}_{\mathtt{E}\psi} \triangleq \langle \Sigma^{\star}, \mathrm{Q}^{\star}, \delta^{\star}, q_I^{\star}, \mathrm{F}^{\star}\rangle$, where the alphabet $\Sigma^{\star} \triangleq 2^{\mathrm{AP}^{\star} \cup \{\mathtt{nw}, \uparrow\}}$ augments the set of extended atomic propositions $\mathrm{AP}^{\star}$ with the symbols $\mathtt{nw}$ and $\uparrow$, as required by the definition of the tree representations $\mathcal{T}_{\mathcal{K}}$. The set of states $\mathrm{Q}^{\star} \triangleq \{q_I^{\star}\} \cup \mathrm{Q} \times \{\downarrow, \uparrow\}$ contains the initial state $q_I^{\star}$ plus two copies of the states of $\mathcal{N}_{\psi}$, one for each direction of navigation over the tree $\mathcal{T}_{\mathcal{K}}$. For the Büchi acceptance condition we consider the set $\mathrm{F}^{\star} \triangleq \{q_I^{\star}\} \cup \mathrm{F} \times \{\downarrow\}$. The definition of the transition function $\delta^{\star}$ follows. For the sake of readability, we split it in three parts, depending on whether it focuses on $q_I^{\star}$, a state q flagged with $\downarrow$, or a state q flagged with $\uparrow$.

- The initial state $q_I^{\star}$ is used to start the evaluation of the formula $\mathtt{E}\psi$ on every world of the input tree labeled by $q_I^{\star}$. This is done by starting the simulation of $\mathcal{N}_{\psi}$. Formally, we have that $\delta^{\star}(q_I^{\star}, \sigma) \triangleq (\Box, q_I^{\star}) \wedge \bigvee_{q \in \mathrm{Q}_I} (\epsilon, (q, \downarrow))$, if $\mathtt{E}\psi \in \sigma$, and $\delta^{\star}(q_I^{\star}, \sigma) \triangleq (\Box, q_I^{\star})$, otherwise.
- Every copy of a state $q \in \mathrm{Q}$ flagged with $\downarrow$ is used to effectively verify the existence of an infinite path in $\mathcal{U}_{\mathcal{K}}$ satisfying ψ. This is done by guessing an extension of the finite path built up to now and sending,

to the corresponding direction, a successor p of q that complies with the transition function δ of $\mathcal{N}_\psi$, when the labeling σ of the world under exam is read. As the input tree $\mathcal{T}_\mathcal{K}$ is a representation of the tree with back edges $\mathcal{U}_\mathcal{K}$, we have also to take them into account when we guess the extension of the path from a world labeled with $\uparrow$. This is done by sending up along the tree the copy of the state $\mathtt{p}$ flagged with $\uparrow$, which is used to simulate a jump to the world destination of the back edge. Formally, we have $\delta^\star((q,\downarrow),\sigma) \triangleq \bigvee_{p\in\delta(q,\sigma\cap \mathrm{AP}^\star)} (\Diamond,(p,\downarrow)) \vee \uparrow(p)$, where $\uparrow(p)$ is set to $(\epsilon,(p,\uparrow))$ if $\uparrow \in \sigma$, and to $\mathtt{f}$, otherwise.

- Finally, for every copy of a state $q \in \mathrm{Q}$ flagged with $\uparrow$, we only have to modify the state and the direction of the automaton when we are approaching to the destination of the back edge that gave rise to the evaluation of $(q,\uparrow)$. Fortunately, due to the structure of the tree-like unwinding $\mathcal{U}_\mathcal{K}$ and, consequently, of its tree representation $\mathcal{T}_\mathcal{K}$, when we reach a world labeled by $\mathtt{nw}$, we are sure that the immediate ancestor of this world is the destination of the back edge. Thus, we can immediately change the flag of the state q to $\downarrow$ in order to resume the verification of the path formula ψ. Formally, $\delta^\star((q,\uparrow),\sigma) \triangleq (\uparrow,(q,\downarrow))$, if $\mathtt{nw} \in \sigma$, and $\delta^\star((q,\uparrow),\sigma) \triangleq (\uparrow,(q,\uparrow))$, otherwise.

Now, by construction, it is not hard to prove that $\mathcal{A}_{\mathtt{E}\psi}$ correctly verifies that every world of $\mathcal{T}_\mathcal{K}$ labeled by $\mathtt{E}\psi$ satisfies $\mathtt{E}\psi$ in $\mathcal{U}_\mathcal{K}$. Also, by the Vardi-Wolper procedure, it follows that $|\mathrm{Q}| = \mathrm{O}(2^{|\psi|})$. Consequently, the size of $\mathcal{A}_{\mathtt{E}\psi}$ is exponential in the length of $\mathtt{E}\psi$. Note that in case $\mathtt{E}\psi$ is a CTL path formula, *i.e.*, $\psi = \mathtt{X}\varphi$, $\psi = \varphi_1\mathtt{U}\varphi_2$, or $\psi = \varphi_1\mathtt{R}\varphi_2$ with φ, φ_1, and φ_2 state formulas, the set Q has constant size, so, the size of $\mathcal{A}_{\mathtt{E}\psi}$ is constant as well.

The construction of $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$ is quite more complex than the one previously described, as it also requires a projection operation that is the reason behind the exponential gap between the upper and lower bounds. Differently from the automata for classic path quantifiers, we cannot evaluate the correctness of the labeling $\mathtt{E}^{\circlearrowleft}\psi$ on all worlds of the tree in one shot. This is because of the possible interactions among the cycles starting in different worlds, which does not allow us to determine which is the origin of the path we are interested in. Consequently, we have to focus on one world labeled by $\mathtt{E}^{\circlearrowleft}\psi$ at a time and check the existence of a path passing infinitely often through that world, which also satisfies the property ψ. This unique world is identified by a fresh symbol #. Then, an universal projection operation over such a symbol

will take care of the fact that this check has to be done for every possible world labeled by $\mathtt{E}^{\circlearrowleft}\psi$. Formally, $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$ is built as follows: $\Pi^{\forall}_{\#}(\mathcal{N}_{\#} \vee \mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi})$. Intuitively, we make a universal projection over # of a disjunction between the automaton $\mathcal{N}_{\#}$, accepting all trees where the labeling # is incorrect (*i.e.*, there are more than one occurrences of # or this symbol is on a world that is not labeled by $\mathtt{E}^{\circlearrowleft}\psi$), and the automaton $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi}$, verifying the existence of a path satisfying ψ that starts and passes infinitely often through the world labeled by #. The construction of $\mathcal{N}_{\#}$ is trivial. For the computation of the projection, we use the equality $\Pi^{\forall}_{\#}\mathcal{A} = \neg\Pi^{\exists}_{\#}\neg\mathcal{A}$. Note however that there is no known projection operation that can act directly on a two-way automaton. Instead, we have first to translate it into a nondeterministic one-way automaton [5] and then apply the standard projection. Due to the nondeterminization procedure, $\Pi^{\forall}_{\#}\mathcal{A}$ has exponential size *w.r.t.* that of $\mathcal{A}$. So, $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$ is exponential in the size of $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi}$.

It remains to define the latter automaton. As above, let $\mathcal{N}_{\psi} = \langle 2^{\mathrm{AP}^{\star}}, \mathrm{Q}, \delta, \mathrm{Q}_I, \mathrm{F}\rangle$ be the nondeterministic Büchi word automaton obtained by applying the Vardi-Wolper construction to ψ. Then, we set $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi} \triangleq \langle \Sigma^{\star}, \mathrm{Q}^{\star}, \delta^{\star}, q^{\star}_I, \mathrm{F}^{\star}\rangle$ as a two-way Büchi tree automaton having alphabet $\Sigma^{\star} \triangleq 2^{\mathrm{AP}^{\star}\cup\{\mathtt{nw},\uparrow,\#\}}$. The set of states $\mathrm{Q}^{\star} \triangleq \{q^{\star}_I\} \cup \mathrm{Q} \times \{\mathtt{f},\mathtt{t}\} \times \{\#,\downarrow,\uparrow\}$ contains the initial state $q^{\star}_I$ plus six copies of the states of $\mathcal{N}_{\psi}$. Each of them is flagged with a Boolean value keeping track of the original acceptance condition derived from $\mathcal{N}_{\psi}$ and a symbol indicating the direction of navigation over the tree. Differently from the previous case, we have also # as a flag in order to indicate the passage though the state labeled by the flag itself. For the Büchi acceptance condition we consider the set $\mathrm{F}^{\star} \triangleq \{q^{\star}_I\} \cup \mathrm{Q} \times \{\mathtt{t}\} \times \{\#\}$. Intuitively, apart from the initial state, we assume as final those states that certify both the passage through the origin of the path indicated by # and the possibly previous occurrence of an accepting state. It remains to define the transition function $\delta^{\star}$. Here we use $\beta(q,\alpha)$ to denote the Boolean value $\mathtt{t}$, if $q \in \mathrm{F}$, and α, otherwise.

- The initial state $q^{\star}_I$ is used to start evaluating the formula $\mathtt{E}^{\circlearrowleft}\psi$ on the unique world of the input tree labeled by #. Formally, we have $\delta^{\star}(q^{\star}_I,\sigma) \triangleq \bigvee_{q\in \mathrm{Q}_I}(\epsilon,(q,\beta(q,\mathtt{f}),\downarrow))$, if $\# \in \sigma$, and $\delta^{\star}(q^{\star}_I,\sigma) \triangleq (\Box, q^{\star}_I)$, otherwise. Note that, since we are just starting with the simulation of $\mathcal{N}_{\psi}$, the flag $\beta(q,\mathtt{f})$ concerning the memory on the acceptance condition only depends on the state q, as the second argument is fixed to $\mathtt{f}$.

- Since a state $(q, \alpha, \#)$ is just used to verify the passage through the starting point of the path satisfying ψ, the automaton has to reset the memory on the acceptance condition and continue with the simulation of $\mathcal{N}_\psi$. Formally, $\delta^\star((q, \alpha, \#), \sigma) \triangleq (\epsilon, (q, \mathtt{f}, \downarrow))$.

- The automaton $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi}$ on the state $(q, \alpha, \downarrow)$ behaves almost as $\mathcal{A}_{\mathtt{E}\psi}$ on $(q, \downarrow)$, with only two differences. The first resides in the update of the memory on the acceptance condition via the function $\beta(p, \alpha)$, which takes into account both the previous memory α and the membership of p in F. The other difference is that, if σ contains the symbol $\#$, we have to record this fact in the state, by swapping the flag from $\downarrow$ to $\#$. Formally, we have $\delta^\star((q, \alpha, \downarrow), \sigma) \triangleq \bigvee_{p \in \delta(q, \sigma \cap \mathrm{AP}^\star)} (\Diamond, (p, \beta(p, \alpha), \gamma)) \vee \uparrow(p)$, where $\gamma = \#$, if $\# \in \sigma$, and $\gamma = \downarrow$, otherwise; moreover, $\uparrow(p)$ is set to $(\epsilon, (p, \beta(p, \alpha), \uparrow))$, if $\uparrow \in \sigma$, and to $\mathtt{f}$, otherwise.

- Finally, as for $\mathcal{A}_{\mathtt{E}\psi}$, a state of the form $(q, \alpha, \uparrow)$ identifies the destination of a back edge. Thus, we have $\delta^\star((q, \alpha, \uparrow), \sigma) \triangleq (\uparrow, (q, \alpha, \downarrow))$, if $\mathtt{nw} \in \sigma$, and $\delta^\star((q, \alpha, \uparrow), \sigma) \triangleq (\uparrow, (q, \alpha, \uparrow))$, otherwise.

It is easy to verify that the size of $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi}$ is exponential in the length of $\mathtt{E}^{\circlearrowleft}\psi$, which implies that $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$ is doubly exponential in the same length. Note that in case $\mathtt{E}^{\circlearrowleft}\psi$ is a CTL cycle-path formula, *i.e.*, $\psi = \mathtt{X}\varphi$, $\psi = \varphi_1 \mathtt{U} \varphi_2$, or $\psi = \varphi_1 \mathtt{R} \varphi_2$ with φ, φ_1, and φ_2 state formulas, the size of $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi}$ is constant. Consequently, so is the one of $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$. □

We say that a $\mathrm{CTL}^\star_{\circlearrowleft}$ formula is in *cycle-existential* form, if it is in normal form and does not contain any occurrence of the universal-cycle quantificer $\mathtt{A}^{\circlearrowleft}$. We call the subset of $\mathrm{CTL}^\star_{\circlearrowleft}$ formulas in cycle-existential form the *cycle-existential* fragment of $\mathrm{CTL}^\star_{\circlearrowleft}$. In case we want to restrict our attention to the satisfiability of the *cycle-existential* fragment of $\mathrm{CTL}^\star_{\circlearrowleft}$, we can improve the previous proof, obtaining a tight 2ExpTime procedure, by providing a single exponential construction for the automaton $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$. Indeed, thanks to the simple cycle property of the verification of the formula $\mathtt{E}^{\circlearrowleft}\psi$ on the tree-like unwinding $\mathcal{U}_\mathcal{K}$, we can just focus on cycle paths of $\mathcal{U}_\mathcal{K}$ going through the successors of their origin labeled by $\mathtt{nw}$. In this way, there are no interactions among the paths that start at different worlds labeled by $\mathtt{E}^{\circlearrowleft}\psi$, since two paths passing through the same world necessarily use different successors. Consequently, we can always uniquely identify the origin of a path on which we have to pass infinitely often.

Unfortunately, the same idea cannot be exploited for the verification of the universal looping quantifiers $\mathtt{A}^{\circlearrowleft}\psi$, as we have to check the property ψ on all cycle paths and not only on those that are simple. At the moment, it is left open whether a 2ExpTime satisfiability procedure for the whole $\mathrm{CTL}^{\star}_{\circlearrowleft}$ logic exists.

Theorem 9. *The satisfiability problem for* $\mathrm{CTL}^{\star}_{s\circlearrowleft}$ *and the existential-cycle fragment of* $\mathrm{CTL}^{\star}_{\circlearrowleft}$ *is 2*ExpTime-c.

Proof. As described above, in order to deal with the exponential blow up in the construction of the automaton $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$, we avoid the projection operation, by exploiting the simple cycle property. To implement this idea, when the automaton $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$ starts with the guessing of a path satisfying ψ, first it choses a successor of the initial world labeled by the flag $\mathtt{nw}$ and then continues the verification on descendants not labeled by this flag, unless it returns at the root of the path via a back edge, where the cycle starts again.

We can now formalize the structure of $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$. Let $\mathcal{N}_{\psi} = \langle 2^{\mathrm{AP}^{\star}}, \mathrm{Q}, \delta, \mathrm{Q}_I, \mathrm{F}\rangle$ be the Vardi-Wolper automaton for ψ. Then, we set $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi} \triangleq \langle \Sigma^{\star}, \mathrm{Q}^{\star}, \delta^{\star}, q_I^{\star}, \mathrm{F}^{\star}\rangle$ as a two-way automaton having alphabet $\Sigma^{\star} \triangleq 2^{\mathrm{AP}^{\star}\cup\{\mathtt{nw},\uparrow\}}$. The set of states $\mathrm{Q}^{\star} \triangleq \{q_I^{\star}\} \cup \mathrm{Q} \times \{\mathtt{f},\mathtt{t}\} \times \{\#, \flat, \downarrow, \uparrow\}$ contains the initial state $q_I^{\star}$ plus eight copies of the states of $\mathcal{N}_{\psi}$. As in the previous construction for $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi}$, each state is flagged with a Boolean value keeping track of the original acceptance condition derived from $\mathcal{N}_{\psi}$. Moreover, it is associated with two symbols indicating the direction of navigation over the tree plus two other symbols, # and $\flat$, used either to certify the passage through the root of the path or to reach one of its successors labeled by $\mathtt{nw}$. As for $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi}$, the Büchi acceptance condition is set to $\mathrm{F}^{\star} \triangleq \{q_I^{\star}\} \cup \mathrm{Q} \times \{\mathtt{t}\} \times \{\#\}$. At this point, it only remains to define the transition function $\delta^{\star}$.

- The initial state $q_I^{\star}$ starts the evaluation of the formula $\mathtt{E}^{\circlearrowleft}\psi$ on every world of the input tree labeled by it. Formally, we have $\delta^{\star}(q_I^{\star}, \sigma) \triangleq (\Box, q_I^{\star}) \wedge \bigvee_{q\in \mathrm{Q}_I}(\epsilon, (q, \beta(q,\mathtt{f}), \#))$, if $\mathtt{E}^{\circlearrowleft}\psi \in \sigma$, and $\delta^{\star}(q_I^{\star}, \sigma) \triangleq (\Box, q_I^{\star})$, otherwise. Observe that the state q is associated with the flag # to denote the fact that we are on the root of the path.

- At the root of the path, we have to send the automaton towards one of its successors labeled by $\mathtt{nw}$. Moreover, we have to reset the memory on the acceptance condition. Consequently, we have $\delta^{\star}((q,\alpha,\#),\sigma) \triangleq \bigvee_{p\in\delta(q,\sigma\cap\mathrm{AP}^{\star})}(\Diamond, (p, \beta(p,\mathtt{f}), \flat))$.

- In a state of the form $(q, \alpha, \flat)$, the automaton has to verify that the world under exam is labeled by $\mathtt{nw}$ and then continues by guessing the path and the associated verification of the property ψ. Hence, we have $\delta^\star((q, \alpha, \flat), \sigma) \triangleq \bigvee_{p \in \delta(q, \sigma \cap \mathrm{AP}^\star)} (\Diamond, (p, \beta(p, \alpha), \downarrow)) \vee \uparrow(p)$, if $\mathtt{nw} \in \sigma$, and $\delta^\star((q, \alpha, \flat), \sigma) \triangleq \mathtt{f}$, otherwise, where $\uparrow(p)$ is defined as in the construction of $\mathcal{A}^{\#}_{\mathtt{E}^{\circlearrowleft}\psi}$, *i.e.*, $\uparrow(p)$ is set to $(\epsilon, (p, \beta(p, \alpha), \uparrow))$, if $\uparrow \in \sigma$, and to $\mathtt{f}$, otherwise.

- The evolution of $\mathcal{A}_{\mathtt{E}^{\circlearrowleft}\psi}$ on a state $(q, \alpha, \downarrow)$ is the same of that on a state $(q, \alpha, \flat)$. The only difference is that they are applied when the world under exam is not labeled by $\mathtt{nw}$. Formally, $\delta^\star((q, \alpha, \downarrow), \sigma) \triangleq \bigvee_{p \in \delta(q, \sigma \cap \mathrm{AP}^\star)} (\Diamond, (p, \beta(p, \alpha), \downarrow)) \vee \uparrow(p)$, if $\mathtt{nw} \notin \sigma$, and $\delta^\star((q, \alpha, \downarrow), \sigma) \triangleq \mathtt{f}$, otherwise.

- As in the previous construction, a state flagged by $\uparrow$ is used to determine the destination of a back edge. In this case, however, we also record in the state the reaching of the root of the path by switching the flag to $\#$ instead of $\downarrow$. Formally, $\delta^\star((q, \alpha, \uparrow), \sigma) \triangleq (\uparrow, (q, \alpha, \#))$, if $\mathtt{nw} \in \sigma$, and $\delta^\star((q, \alpha, \uparrow), \sigma) \triangleq (\uparrow, (q, \alpha, \uparrow))$, otherwise.

By construction, it is immediate to see that the above automaton also works for the existential simple-cycle-path quantifier $\mathtt{E}^{\circlearrowleft}_s \psi$. Thus, by exploiting the dualization property of alternating automata, we can construct an automaton for the universal simple-cycle-path quantifier $\mathtt{A}^{\circlearrowleft}_s \psi$ as well. Consequently, the global automaton for an arbitrary $\mathrm{CTL}^\star_{s\circlearrowleft}$ formula is only exponential in the size of the specification. □

5. Discussion

Spurred by the observation that most of the solution techniques for solving the model-checking and satisfiability problems of temporal logics are cycle detection based, we introduced extensions of CTL* and CTL that explicitly take

	Model-Checking		Satisfiability
	Formula	Data	
CTL	PTime-c	NLogSpace-c	ExpTime-c
$\mathrm{CTL}_{\circlearrowleft}$	PTime-c	NLogSpace-c	ExpTime-c
$\mathrm{CTL}_{s\circlearrowleft}$	PTime-c	PSpace	ExpTime-c
CTL*	PSpace-c	NLogSpace-c	2ExpTime-c
$\mathrm{CTL}^\star_{\circlearrowleft}$	PSpace-c	NLogSpace-c	3ExpTime
$\mathrm{CTL}^\star_{s\circlearrowleft}$	PSpace-c	PSpace	2ExpTime-c

Table 1: Complexity Table

into account the existence of cycles in their models. Specifically, we introduced $\mathrm{CTL}^{\star}_{\circlearrowleft}$ and $\mathrm{CTL}_{\circlearrowleft}$ by adding cycle quantifiers to $\mathrm{CTL}^{\star}$ and CTL that restrict their range of quantification on the paths that are cyclic, *i.e.*, in which the initial state occurs infinitely often. We investigate on the expressive power of such logics, finding that they are strictly more expressive than their classic counter-parts $\mathrm{CTL}^{\star}$ and CTL, as well as orthogonal to the μ-calculus. We also introduced $\mathrm{CTL}^{\star}_{s\circlearrowleft}$ and $\mathrm{CTL}_{s\circlearrowleft}$ in which the cycle quantifiers are further restricted to predicate over simple cycles. For all these logics, we addressed both the model-checking and satisfiability problems. The results are summarized in Table 1.

The results found are very surprising. Given that the the model-checking and the satisfiability problems for $\mathrm{CTL}^{\star}$ and its fragments are based on cycle detection techniques, one might expect that adding an explicit control on the cycles in the logic would not make any difference in terms of both expressiveness and complexity. However, we have shown that the logics result to be expressively incomparable with μ-calculus. In addition to this, solving the satisfiability problem turns out to be a non-trivial extension of the one for $\mathrm{CTL}^{\star}$. Apparently, the reason why this happens is not directly related to the ability of stating the existence of a cycle, but instead of its negation, *i.e.* stating that there are non cyclic paths. As an indication for that, the lack of finite-model property (and consequently lack of bisimulation) follows from the ability of enforcing the satisfying subtrees to *never* start a cyclic branch. At the current state, this phenomenon has not been entirely singled out and we plan to do it on a future work.

Another possible direction for future work is to investigate the use of the introduced cycle construct in the realm of logics for multi-agent systems such as $\mathrm{ATL}^{\star}$ [2] and Strategy Logic [24, 25]. These logics have been proved to be useful to reasoning about strategic abilities in a number of complicated settings. In particular, the latter is able to express sophisticated solution concepts such as Nash Equilibria and Subgame Perfect Equilibria, as well as it has been used to express iterative extensive game forms such as the iterated prisoner dilemma. In all these contexts, talking explicitly about cycles could play a central role in solving the related game questions.

Finally, the notion of cycle dealt with in this work is extremely general: the sequences between two occurrences of the recurrent state can have different lengths and contain different states. It would be interesting to analyze a simpler notion of cycle, for instance a regular one, as well. In addition, one can think to explicitly verify whether a property holds between subse-

quent occurrences of the repeating state. These questions are left as future research.

Acknoledgments

We would like to thank the reviewer for their thoughtful and detailed comments. Fabio Mogavero, Aniello Murano, and Loredana Sorrentino thank the support of the GNCS 2018 project "Metodi Formali per la Verica e la Sintesi di Sistemi Discreti e Ibridi". Giuseppe Perelli thanks the support of the ERC Advanced Grant 291528 ("Race") at Oxford.

References

[1] Alur, R., Henzinger, T., 1998. Finitary Fairness. TOPLAS 20 (6), 1171–1194.

[2] Alur, R., Henzinger, T., Kupferman, O., 2002. Alternating-Time Temporal Logic. JACM 49 (5), 672–713.

[3] Alur, R., La Torre, S., 2004. Deterministic generators and games for ltl fragments. ACM Transactions on Computational Logic (TOCL) 5 (1), 1–25.

[4] Aminof, B., Murano, A., Rubin, S., Zuleger, F., 2016. Prompt Alternating-Time Epistemic Logics. In: KR'16. AAAI Press, pp. 258–267.

[5] Bonatti, P., Lutz, C., Murano, A., Vardi, M., 2008. The Complexity of Enriched muCalculi. LMCS 4 (3), 1–27.

[6] Bozzelli, L., Murano, A., Peron, A., 2010. Pushdown Module Checking. FMSD 36 (1), 65–95.

[7] Chatterjee, K., Henzinger, T., Horn, F., 2010. Finitary Winning in omega-Regular Games. TOCL 11 (1), 1:1–26.

[8] Clarke, E., Emerson, E., 1981. Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In: LP'81. LNCS 131. Springer, pp. 52–71.

[9] Clarke, E., Emerson, E., Sistla, A., 1986. Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications. TOPLAS 8 (2), 244–263.

[10] Clarke, E., Grumberg, O., Peled, D., 2002. Model Checking. MIT Press.

[11] Emerson, E., Halpern, J., 1985. Decision Procedures and Expressiveness in the Temporal Logic of Branching Time. JCSS 30 (1), 1–24.

[12] Emerson, E., Halpern, J., 1986. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. JACM 33 (1), 151–178.

[13] Emerson, E., Jutla, C., 1988. The Complexity of Tree Automata and Logics of Programs (Extended Abstract). In: FOCS'88. IEEE Computer Society, pp. 328–337.

[14] Ferrante, A., Murano, A., Parente, M., 2008. Enriched Mu-Calculi Module Checking. LMCS 4 (3), 1–21.

[15] Fontaine, G., Mogavero, F., Murano, A., Perelli, G., Sorrentino, L., 2016. Cycle detection in computation tree logic. In: Proceedings of the Seventh International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2016, Catania, Italy, 14-16 September 2016. pp. 164–177.

[16] Grädel, E., Thomas, W., Wilke, T., 2002. Automata, Logics, and Infinite Games: A Guide to Current Research. LNCS 2500. Springer.

[17] Kripke, S., 1963. Semantical Considerations on Modal Logic. APF 16, 83–94.

[18] Kupferman, O., Morgenstern, G., Murano, A., 2006. Typeness for omega-Regular Automata. IJFCS 17 (4), 869–884.

[19] Kupferman, O., Piterman, N., Vardi, M., 2002. Pushdown Specifications. In: LPAR'02. LNCS 2514. Springer, pp. 262–277.

[20] Kupferman, O., Piterman, N., Vardi, M., 2009. From Liveness to Promptness. FMSD 34 (2), 83–103.

[21] Kupferman, O., Pnueli, A., Vardi, M., 2012. Once and For All. JCSS 78 (3), 981–996.

[22] Kupferman, O., Vardi, M., Wolper, P., 2000. An Automata Theoretic Approach to Branching-Time Model Checking. JACM 47 (2), 312–360.

[23] Kupferman, O., Vardi, M., Wolper, P., 2001. Module Checking. IC 164 (2), 322–344.

[24] Mogavero, F., Murano, A., Perelli, G., Vardi, M., 2014. Reasoning About Strategies: On the Model-Checking Problem. TOCL 15 (4), 34:1–42.

[25] Mogavero, F., Murano, A., Perelli, G., Vardi, M., 2017. Reasoning About Strategies: On the Satisfiability Problem. LMCS 13 (1).

[26] Mogavero, F., Murano, A., Sorrentino, L., 2015. On promptness in parity games. Fundamenta Informaticae 139 (3), 277–305.

[27] Pnueli, A., 1977. The Temporal Logic of Programs. In: FOCS'77. IEEE Computer Society, pp. 46–57.

[28] Vardi, M., 1998. Reasoning about The Past with Two-Way Automata. In: ICALP'98. LNCS 1443. Springer, pp. 628–641.

[29] Vardi, M., Wolper, P., 1986. An Automata-Theoretic Approach to Automatic Program Verification. In: LICS'86. IEEE Computer Society, pp. 332–344.

[30] Wolper, P., 1983. Temporal Logic Can Be More Expressive. IC 56 (1-2), 72–99.

[31] Zielonka, W., 1998. Infinite Games on Finitely Coloured Graphs with Applications to Automata on Infinite Trees. TCS 200 (1-2), 135–183.