Decidability of a Syntactic Fragment of the Hybrid Computation Tree Logic with the ↓ Operator

Mario R. F. Benevides* L. Menasché Schechter†

{mario,luis}@cos.ufrj.br

Abstract

Hybrid logics with the ↓ operator are generally undecidable. In this paper, we present a fragment of the hybrid CTL with the ↓ operator (HCTL(@,↓)), called the reducible fragment, that has the same complexity for the satisfiability problem as standard CTL (EXPTIME-Complete). In order to prove this, we show that the satisfiability problem for a formula in this fragment can be polynomially reduced to the satisfiability problem for a formula in the hybrid CTL without the ↓ operator (HCTL(@)).

1 Introduction

The branching-time temporal logic known as Computation Tree Logic, or CTL, was developed as a tool to reason about concurrent or nondeterministic systems. It is a very used logic for automated verification (model-checking) of computer software and hardware specifications, because of its polynomial-time model-checking complexity [6].

CTL was first introduced by Emerson and Clarke in [8]. Emerson and Halpern, in [9], presented a complete axiomatization for this logic, established a finite model property for it and provided an exponential time decision procedure for its satisfiability problem. What allowed these three results to be proven was the development of a completeness proof based on finitary models.

Hybrid logics are extensions of modal logics that allow explicit references to individual states of a model. Their goal is to extend the expressive power of ordinary modal logics, without losing their good properties, such as decidability.

However, the task of increasing the expressive power of a language without losing decidability is not so easy. For instance, one hybrid operator that is very expressive and very appealing is the ↓ operator. But, as [4] and [2] show, hybrid logics with this operator are generally undecidable. This has motivated research on decidable fragments of such logics.

*Computer Science Department and Systems and Computer Engineering Program, Federal University of Rio de Janeiro, Brazil. The author was partially supported by a grant from CNPq.

†Systems and Computer Engineering Program, Federal University of Rio de Janeiro, Brazil. The author was supported by a D.Sc. scholarship from CNPq.

In order to bring back decidability in the presence of the $\downarrow$ operator, the logic has to be somehow restricted. The restrictions fall into two broad categories: semantic restrictions and syntactic restrictions.

Semantic restrictions impose conditions on the models in which the formulas of the logic are evaluated. [11] shows that the basic hybrid logic with the $\downarrow$ operator is decidable if the models are restricted to linear transitive structures and [15] shows an analogous result with the restriction to transitive tree models. Another semantic restriction involves limiting the out-degree of every state in the model to a fixed upper-bound. [17] shows that, with any such finite upper-bound, the basic hybrid logic with the $\downarrow$ operator is decidable.

Syntactic restrictions impose conditions on the construction of formulas, allowing only certain types of interaction between the operators of the language (mainly between $\downarrow$ and the other operators). [1] shows that, for a series of hybrid logics, the fragment in which the $\downarrow$ operator cannot appear inside the scope of *any* other operator (including another $\downarrow$) is decidable. Later, this fragment was extended to a larger one in the (seemingly independent) works [18, 11, 17]. Their fragment, called *innocent fragment* in [18], *existential fragment* in [11] and FHL$\setminus \Box \downarrow$ in [17], consists of the set of formulas that, when put in negation normal form, do not have a $\downarrow$ appearing inside the scope of any universal operator of the language. This fragment is larger than the previous one because the $\downarrow$ operator can now appear inside the scope of some operators (including another $\downarrow$). [17] goes further and present an even larger fragment, called FHL$\setminus \Box \downarrow \Box$, consisting of the set of formulas that, when put in negation normal form, do not have any $\downarrow$ that simultaneously has an universal operator in its scope and is inside the scope of an universal operator.

The investigation of hybrid temporal logics with the $\downarrow$ operator goes back to a series of papers by Goranko [12, 13, 14]. [12] is also the paper that introduces the $\downarrow$ operator for hybrid logics. However, only [14], that introduces a branching-time temporal logic that is strong enough to express all CTL* formulas and [19], that explicitly mix the CTL operators and the hybrid operators, explore hybrid extensions of the Computation Tree Logic. Nevertheless, both of them make a restriction to tree models.

Our goal in this paper is to explore the hybrid extension of CTL with the $\downarrow$ operator with no restrictions on the shape of the models. In order to keep decidability, we analyze a syntactic fragment similar to the one presented in [18], [11] and [17]. We call it the *reducible fragment*.

The rest of this paper is organized as follows. In section 2, we present the syntax and semantics of CTL. In section 3, we introduce some of the main resources that are available in hybrid logics and define the syntax and semantics of the hybrid-CTL with (HCTL(@,$\downarrow$)) and without the $\downarrow$ operator (HCTL(@)). We also present the syntactic fragments of HCTL(@,$\downarrow$) that are the object of our study: the reducible, co-reducible, strongly reducible and unreducible fragments. In section 4, we prove that the satisfiability problem for formulas in the reducible fragment is decidable and its complexity is EXPTIME-Complete, the same complexity of standard CTL. Finally, in section 5, we state our concluding remarks.

2 Computation Tree Logic

In this section, we present the syntax and semantics of the temporal logic CTL.

Definition 2.1. *The CTL language is a language consisting of a set Φ of countably many proposition symbols (the elements of Φ are denoted by $p, q, \ldots$), the boolean connectives $\neg$, $\wedge$ and $\vee$ and the temporal operators* **EX**, **AX**, **EU**, **AU**, **ER** *and* **AR**. *The formulas are defined as follows:*

$$\phi ::= p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \mathbf{EX}\phi \mid \mathbf{AX}\phi \mid \mathbf{E}(\phi_1 \mathbf{U} \phi_2) \mid \mathbf{A}(\phi_1 \mathbf{U} \phi_2) \mid$$

$$\mathbf{E}(\phi_1 \mathbf{R} \phi_2) \mid \mathbf{A}(\phi_1 \mathbf{R} \phi_2).$$

We freely use the standard boolean abbreviations $\top \equiv p \vee \neg p$, $\bot \equiv p \wedge \neg p$, $\phi_1 \rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2$ and $\phi_1 \leftrightarrow \phi_2 \equiv (\phi_1 \rightarrow \phi_2) \wedge (\phi_2 \rightarrow \phi_1)$ and also the following abbreviations: $\mathbf{EF}\phi \equiv \mathbf{E}(\top \mathbf{U} \phi)$, $\mathbf{AF}\phi \equiv \mathbf{A}(\top \mathbf{U} \phi)$, $\mathbf{AG}\phi \equiv \neg\mathbf{EF}\neg\phi$ and $\mathbf{EG}\phi \equiv \neg\mathbf{AF}\neg\phi$. It should be noted that we could write $\mathbf{EU}(\phi_1, \phi_2)$, $\mathbf{AU}(\phi_1, \phi_2)$, $\mathbf{ER}(\phi_1, \phi_2)$ and $\mathbf{AR}(\phi_1, \phi_2)$, but the infix notation is more common.

We now define the structures in which we evaluate formulas in CTL (and in modal logics in general): *frames* and *models*.

Definition 2.2. *A* frame *for CTL is a pair $\mathcal{F} = (V, R)$, where V is a set (finite or not) of states and R is a binary* serial *relation over V, i.e., $R \subseteq V \times V$ and, for all $v \in V$, there is a $w \in V$ such that vRw.*

Definition 2.3. *A* model *for CTL is a pair $\mathcal{M} = (\mathcal{F}, \mathbf{V})$, where $\mathcal{F}$ is a frame and $\mathbf{V}$ is a valuation function mapping proposition symbols into subsets of V, i.e., $\mathbf{V} : \Phi \mapsto 2^V$.*

Definition 2.4. *A* ω-path *in a model is an infinite sequence $P = (v_1, v_2, \ldots)$ of states, such that $v_i R v_{i+1}$, for $i \geq 1$. If v is the first element of the sequence P, we say that P starts at v. The finite sequence constructed with the first k elements of P is called the k-*prefix *of P and denoted by P_k.*

The semantical notion of satisfaction for CTL is defined as follows:

Definition 2.5. *Let $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ be a model. The notion of* satisfaction *of a formula ϕ in a model $\mathcal{M}$ at a state v, notation $\mathcal{M}, v \Vdash \phi$, can be inductively defined as follows:*

1. *$\mathcal{M}, v \Vdash p$ iff $v \in \mathbf{V}(p)$;*
2. *$\mathcal{M}, v \Vdash \neg\phi$ iff $\mathcal{M}, v \nVdash \phi$;*
3. *$\mathcal{M}, v \Vdash \phi_1 \wedge \phi_2$ iff $\mathcal{M}, v \Vdash \phi_1$ and $\mathcal{M}, v \Vdash \phi_2$;*
4. *$\mathcal{M}, v \Vdash \phi_1 \vee \phi_2$ iff $\mathcal{M}, v \Vdash \phi_1$ or $\mathcal{M}, v \Vdash \phi_2$;*
5. *$\mathcal{M}, v \Vdash \mathbf{EX}\phi$ iff there is a $w \in V$ such that vRw and $\mathcal{M}, w \Vdash \phi$;*
6. *$\mathcal{M}, v \Vdash \mathbf{AX}\phi$ iff for all $w \in V$ such that vRw, $\mathcal{M}, w \Vdash \phi$;*
7. *$\mathcal{M}, v \Vdash \mathbf{E}(\phi_1 \mathbf{U} \phi_2)$ iff there is a ω-path P starting at v such that, for at least one $k \geq 1$, the elements of $P_k = (v_1, \ldots, v_k)$ satisfy $\mathcal{M}, v_k \Vdash \phi_2$ and $\mathcal{M}, v_i \Vdash \phi_1$, for $1 \leq i < k$;*

8. $\mathcal{M}, v \Vdash \mathbf{A}(\phi_1 \mathbf{U} \phi_2)$ *iff for all* ω*-paths* P *starting at* v*, there is at least one* $k \geq 1$*, such that the elements of* $P_k = (v_1, \ldots, v_k)$ *satisfy* $\mathcal{M}, v_k \Vdash \phi_2$ *and* $\mathcal{M}, v_i \Vdash \phi_1$*, for* $1 \leq i < k$*.*

9. $\mathcal{M}, v \Vdash \mathbf{E}(\phi_1 \mathbf{R} \phi_2)$ *iff there is a* ω*-path* P *starting at* v *such that either* $\mathcal{M}, v_i \Vdash \phi_2$ *for all* $v_i \in P$ *or, for at least one* $k \geq 1$*, the elements of* $P_k = (v_1, \ldots, v_k)$ *satisfy* $\mathcal{M}, v_k \Vdash \phi_1$ *and* $\mathcal{M}, v_i \Vdash \phi_2$*, for* $1 \leq i \leq k$*;*

10. $\mathcal{M}, v \Vdash \mathbf{A}(\phi_1 \mathbf{R} \phi_2)$ *iff for all* ω*-paths* P *starting at* v*, either* $\mathcal{M}, v_i \Vdash \phi_2$ *for all* $v_i \in P$ *or, for at least one* $k \geq 1$*, the elements of* $P_k = (v_1, \ldots, v_k)$ *satisfy* $\mathcal{M}, v_k \Vdash \phi_1$ *and* $\mathcal{M}, v_i \Vdash \phi_2$*, for* $1 \leq i \leq k$*;*

Definition 2.6. *Two formulas* ϕ_1 *and* ϕ_2 *are* semantically equivalent *(or just* equivalent*), denoted by* $\phi_1 \equiv \phi_2$*, if* $\mathcal{M}, v \Vdash \phi_1$ *if and only if* $\mathcal{M}, v \Vdash \phi_2$*, for any model* $\mathcal{M}$ *and any* $v \in \mathcal{M}$[1]*.*

From definition 2.5, it is not difficult to see that the operators of the language are related through the following semantic equivalences: $\neg\neg\phi \equiv \phi$, $\phi_1 \vee \phi_2 \equiv \neg(\neg\phi_1 \wedge \neg\phi_2)$, $\mathbf{AX}\phi \equiv \neg\mathbf{EX}\neg\phi$, $\mathbf{E}(\phi_1 \mathbf{R} \phi_2) \equiv \neg\mathbf{A}(\neg\phi_1 \mathbf{U} \neg\phi_2)$ and $\mathbf{A}(\phi_1 \mathbf{R} \phi_2) \equiv \neg\mathbf{E}(\neg\phi_1 \mathbf{U} \neg\phi_2)$.

If $\mathcal{M}, v \Vdash \phi$, we say that ϕ is satisfied in the model $\mathcal{M}$ at the state v. We also say that a formula ϕ is *satisfiable* iff there is a model $\mathcal{M}$ and a state $v \in \mathcal{M}$ such that $\mathcal{M}, v \Vdash \phi$. The problem of deciding whether a given formula of the logic is satisfiable or not is called the *satisfiability problem*.

Theorem 2.7 ([7])**.** *The satisfiability problem for CTL formulas is decidable, having EXPTIME-Complete complexity.*

A formula ϕ is *valid in a frame* $\mathcal{F}$ ($\mathcal{F} \Vdash \phi$) iff for every model $\mathcal{M}$ of the frame $\mathcal{F}$ and every state $v \in \mathcal{M}$, $\mathcal{M}, v \Vdash \phi$. ϕ is *valid* ($\Vdash \phi$) iff it is valid in all frames.

3 Hybrid Extensions of CTL

A good way to improve the expressive power of a modal logic is to consider hybrid extensions of it. The fundamental resource that allows a logic to be called "hybrid" is a set of *nominals*. Nominals are a new kind of atomic symbol and they behave similarly to proposition symbols. The key difference between a nominal and a proposition symbol is related to their valuation in a model. While the set $\mathbf{V}(p)$ for a proposition symbol p can be any element of 2^V, the set $\mathbf{V}(i)$ for a nominal i has to be a singleton set. This way, each nominal is true at exactly one state of the model, and thus, can be used to refer to this unique state. This is why these logics are called "hybrid": they are still modal logics, but they have the capacity to refer to specific states of the model, like in first-order logic. Hybrid logics may also have state-variables, the $\downarrow$ operator, which binds a state-variable to the current state, and the satisfaction operators $@_i$ and $@_x$, for each nominal i and each state-variable x, which evaluate formulas in specific states of the model. For a general introduction to hybrid logics, [3] and [4] can be consulted.

[1] $v \in \mathcal{M}$ means that $v \in V$, where V is the set of states of $\mathcal{M}$.

In this section, we present two hybrid extensions of CTL. The first does not have state-variables and the $\downarrow$ operator, while the second has. We show that this very small difference in the language results in a huge difference in complexity: the satisfiability problem for the first logic is no more difficult than the one for standard CTL, while for the second logic it is undecidable.

3.1 HCTL(@)

Definition 3.1. *The HCTL(@) language is a hybrid language consisting of a set Φ of countably many proposition symbols (the elements of Φ are denoted by $p, q, \ldots$), a set Ω of countably many nominals (the elements of Ω are denoted by $i, j, \ldots$), such that $\Phi \cap \Omega = \emptyset$, the boolean connectives $\neg$, $\wedge$ and $\vee$, the temporal operators of CTL and the operators $@_i$, for each nominal i (called* satisfaction operators*). The formulas are defined as follows:*

$$\phi ::= p \mid i \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \mathcal{X}\phi \mid \mathcal{U}(\phi_1, \phi_2) \mid @_i\phi,$$

where $\mathcal{X} \in \{\mathbf{EX}, \mathbf{AX}\}$ and $\mathcal{U} \in \{\mathbf{EU}, \mathbf{AU}, \mathbf{ER}, \mathbf{AR}\}$.

The definition of a frame for HCTL(@) is the same as definition 2.2, but the definition of a model for HCTL(@) is slightly different from definition 2.3.

Definition 3.2. *A* model *for HCTL(@) is a pair $\mathcal{M} = (\mathcal{F}, \mathbf{V})$, where $\mathcal{F}$ is a frame and $\mathbf{V}$ is a valuation function mapping proposition symbols into subsets of V, i.e., $\mathbf{V} : \Phi \mapsto 2^V$ and mapping nominals into singleton subsets of V, i.e, if i is a nominal then $\mathbf{V}(i) = \{v\}$ for some $v \in V$. We call this unique state that belongs to $\mathbf{V}(i)$ the* denotation *of i under $\mathbf{V}$. We can also say that i denotes the single state belonging to $\mathbf{V}(i)$ or that i* names *this single state.*

The semantical notion of satisfaction for HCTL(@) is defined as follows:

Definition 3.3. *The notion of satisfaction for formulas in HCTL(@) is defined adding two extra clauses to the satisfaction definition of CTL (definition 2.5):*

1. *$\mathcal{M}, v \Vdash i$ iff $v \in \mathbf{V}(i)$;*
2. *$\mathcal{M}, v \Vdash @_i\phi$ iff $\mathcal{M}, d \Vdash \phi$, where d is the denotation of i under $\mathbf{V}$;*

For each nominal i, the formula $@_i\phi$ means that if $\mathbf{V}(i) = \{v\}$ then ϕ is satisfied at v.

CTL can be polynomially translated into the Modal Mu-Calculus, a modal logic with a least-fixpoint operator [5]. [16] proposes a hybrid extension of the Mu-Calculus in which not only standard CTL formulas can be expressed, but so can formulas that use nominals and satisfaction operators. We can then polynomially translate HCTL(@) into Sattler and Vardi's Hybrid Mu-Calculus. This provides a decidability result for HCTL(@), based on the following result:

Lemma 3.4 ([16])**.** *The satisfiability problem for Hybrid Mu-Calculus formulas is decidable, having EXPTIME-Complete complexity.*

Theorem 3.5. *The satisfiability problem for HCTL(@) formulas is decidable, having EXPTIME-Complete complexity.*

Proof. EXPTIME-hardness follows from theorem 2.7 and the EXPTIME upper-bound follows from lemma 3.4. □

3.2 HCTL(@,↓)

Definition 3.6. *The HCTL(@,↓) language is a hybrid language consisting of a set Φ of countably many proposition symbols (the elements of Φ are denoted by $p, q, \ldots$), a set Ω of countably many nominals (the elements of Ω are denoted by $i, j, \ldots$), a set $\mathcal{S}$ of countably many state-variables (the elements of $\mathcal{S}$ are denoted by $x, y, \ldots$), such that Φ, Ω and $\mathcal{S}$ are pairwise disjoint, the boolean connectives $\neg$, $\wedge$ and $\vee$, the temporal operators of CTL, the $\downarrow$ operator and the operators $@_i$, for each nominal i, and $@_x$, for each state-variable x. The formulas are defined as follows:*

$$\phi ::= p \mid i \mid x \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \mathcal{X}\phi \mid \mathcal{U}(\phi_1, \phi_2) \mid @_i\phi \mid @_x\phi \mid \downarrow x.\phi,$$

where $\mathcal{X} \in \{\mathbf{EX}, \mathbf{AX}\}$ and $\mathcal{U} \in \{\mathbf{EU}, \mathbf{AU}, \mathbf{ER}, \mathbf{AR}\}$.

The definition of a frame and a model for HCTL(@,↓) are the same as definitions 2.2 and 3.2, respectively.

In order to deal with the state-variables, we need to introduce the notion of *assignments*.

Definition 3.7. *An* assignment *is a function g that maps state-variables to states of the model, i.e., $g : \mathcal{S} \mapsto V$. We use the notation $g' = g[v_1/x_1, \ldots, v_n/x_n]$ to denote an assignment such that $g'(x) = g(x)$ if $x \notin \{x_1, \ldots, x_n\}$ and $g'(x_i) = v_i$, otherwise.*

The semantical notion of satisfaction for HCTL(@,↓) is defined as follows:

Definition 3.8. *The notion of satisfaction of a formula ϕ in a model $\mathcal{M}$ at a state v with an assignment g, notation $\mathcal{M}, g, v \Vdash \phi$, is inductively defined adding the assignment g to all the clauses in definitions 2.5 and 3.3 and the following three extra clauses:*

1. *$\mathcal{M}, g, v \Vdash x$ iff $g(x) = v$;*
2. *$\mathcal{M}, g, v \Vdash @_x\phi$ iff $\mathcal{M}, d \Vdash \phi$, where $d = g(x)$;*
3. *$\mathcal{M}, g, v \Vdash \downarrow x.\phi$ iff $\mathcal{M}, g[v/x], v \Vdash \phi$.*

The formula $\downarrow x.\phi$ means that, using x as a name for the present state (state-variables can be thought as "on-the-fly nominals"), ϕ is satisfied. The $\downarrow$ operator is the only operator that binds a variable. Free and bounded variables are defined in the usual way. The only case worth mentioning is that in the formula $@_x\psi$, x is free. A *sentence* is a formula with no free variables.

Definition 3.9. *A formula of HCTL(@,↓) is in* negation normal form *(NNF) if the negation symbol ($\neg$) appears only in front of proposition symbols, nominals and state-variables.*

Lemma 3.10. *Every formula of HCTL(@,↓) is semantically equivalent to a formula in NNF.*

Theorem 3.11 ([2])**.** *The satisfiability problem for HCTL(@,↓) is undecidable.*

This result shows that the inclusion of state-variables and the $\downarrow$ operator turns a logic that had the same complexity of the standard CTL into an undecidable logic. In order to bring back decidability, we need to consider fragments of HCTL(@,$\downarrow$). As discussed in section 1, we can do this using semantic or syntactic restrictions.

In this work, as we do not want to impose any restrictions on the shape of the models, we use a syntactic restriction to obtain a decidable fragment of HCTL(@,$\downarrow$). This syntactic fragment is an extension that includes all CTL operators of the fragment presented as the *innocent fragment* in [18], *existential fragment* in [11] and FHL $\setminus \Box \downarrow$ in [17]. We call our fragment the *reducible fragment*, because, in our opinion, this denomination highlights the key property of this fragment, which is used to prove its decidability.

Definition 3.12. *Let $\mathcal{O}$ be any operator of HCTL(@,$\downarrow$), excluding $\neg$ and $\downarrow$. We say that $\mathcal{O}$ is an* existential operator *if the satisfaction of the formula $\mathcal{O}\phi$ depends on ϕ being satisfied at a single state or if it does not even depend on the satisfaction of ϕ (e.g.: $\mathbf{E}(\phi_1 \mathbf{R} \phi_2)$ may still be satisfied even if ϕ_1 is never satisfied in the model) and that $\mathcal{O}$ is an* universal operator *if the satisfaction of the formula $\mathcal{O}\phi$ may depend on ϕ being satisfied at multiple states. If $\mathcal{O}$ is a binary operator $\mathcal{O}(\phi_1, \phi_2)$ (like $\mathbf{AU}$ and $\wedge$), we have to check whether it is existential or universal with respect to each operand. Thus, the boolean operators, the satisfaction operators and $\mathbf{EX}$ are existential and $\mathbf{AX}$ is universal. $\mathbf{AU}$ and $\mathbf{AR}$ are universal with respect to both operands, $\mathbf{EU}$ is existential with respect to the second operand and universal with respect to the first and $\mathbf{ER}$ is existential with respect to the first and universal with respect to the second.*

Definition 3.13. *A formula ϕ of HCTL(@,$\downarrow$) belongs to the* reducible fragment *(RF) iff, when put in NNF, there is no $\downarrow$ occurring inside the scope of an universal operator. It is important to see that RF is not closed under negation.*

Definition 3.14. *We can also define three other fragments related to the reducible fragment:*

1. *The* co-reducible fragment *(co-RF): $\phi \in$ co-RF iff $\neg\phi \in RF$.*
2. *The* strongly reducible fragment *(SRF): $\phi \in SRF$ iff $\phi \in RF$ and $\neg\phi \in RF$. An equivalent definition is $SRF = RF \cap$ co-RF.*
3. *The* unreducible fragment *(UF): $\phi \in UF$ iff $\phi \notin RF$ and $\neg\phi \notin RF$. An equivalent definition is $UF = HCTL(@, \downarrow) \setminus (RF \cup$ co-RF$)$.*

Example 3.15. *It is easy to see that HCTL(@) is a subset of SRF.*

Example 3.16. *$\phi = \mathbf{E}(p\mathbf{U}(\downarrow x.\mathbf{EX}x))$ belongs to RF, since $\mathbf{EU}$ is existential with respect to the second operand. On the other hand, $\phi = \mathbf{E}((\downarrow x.\mathbf{EX}x)\mathbf{U}p)$ belongs to UF, since $\mathbf{EU}$ is universal with respect to the first operand, $\neg\phi = \mathbf{A}((\downarrow x.\mathbf{AX}\neg x)\mathbf{R}\neg p)$ and $\mathbf{AR}$ is universal with respect to both operands.*

4 Decidability of the Reducible Fragment

In this section, we show that the satisfiability problem for sentences in RF is decidable. Moreover, it has the same complexity as the satisfiability problem

for standard CTL. Our restriction to sentences is not a serious limitation, since free state-variables can always be substituted by fresh nominals. The first thing we need to do is to define the semantics of a new hybrid binder: $\exists$.

Definition 4.1. *We add to the language formulas of the form $\exists x.\alpha$, where x is a state-variable. The satisfaction of this kind of formula is defined as follows:*

$$\mathcal{M}, g, v \Vdash \exists x.\phi \text{ iff there is a } w \in V \text{ such that } \mathcal{M}, g[w/x], v \Vdash \phi.$$

This new binder is useful because of the semantic equivalence $\downarrow x.\phi \equiv \exists x.(x \wedge \phi)$ and the following lemma:

Lemma 4.2. *The following are semantic equivalences, if x does not have free occurrences in the formulas and is not bound by any other operator besides the highlighted $\exists$:*

1. $\phi_1 \wedge \exists x.\phi_2 \equiv \exists x.(\phi_1 \wedge \phi_2)$;
2. $\phi_1 \vee \exists x.\phi_2 \equiv \exists x.(\phi_1 \vee \phi_2)$;
3. $\mathbf{EX}\exists x.\phi \equiv \exists x.\mathbf{EX}\phi$;
4. $\mathbf{E}(\phi_1 \mathbf{U} \exists x.\phi_2) \equiv \exists x.\mathbf{E}(\phi_1 \mathbf{U} \phi_2)$;
5. $\mathbf{E}(\exists x.\phi_1 \mathbf{R} \phi_2) \equiv \exists x.\mathbf{E}(\phi_1 \mathbf{R} \phi_2)$;
6. $@_i \exists x.\phi \equiv \exists x.@_i \phi$;
7. $@_y \exists x.\phi \equiv \exists x.@_y \phi$.

This lemma shows that the $\exists$ operator commutes with every existential operator of HCTL(@,$\downarrow$). With it, we can now prove the following theorem, that shows why the reducible fragment has this name.

Theorem 4.3 (Reduction Theorem)**.** *Every sentence ϕ of RF can be reduced to a formula $\widetilde{\phi}$ of HCTL(@) such that ϕ is satisfiable if and only if $\widetilde{\phi}$ is satisfiable. We use the notation $\phi \rightsquigarrow \widetilde{\phi}$ to denote that ϕ can be reduced to $\widetilde{\phi}$.*

Proof. First, we rename the state-variables in ϕ so that no state-variable has free and bound occurrences in ϕ and so that no state-variable is bound by more than one $\downarrow$ operator. Then, we put ϕ in NNF, obtaining ϕ'. As $\phi \in$ RF, there is no $\downarrow$ inside the scope of any universal operator in ϕ'. Now, we substitute every subformula of the form $\downarrow x.\psi$ in ϕ' by $\exists x.(x \wedge \psi)$, obtaining ϕ''. As there was no $\downarrow$ inside the scope of any universal operator in ϕ', there is no $\exists$ inside the scope of any universal operator in ϕ''. We can then use lemma 4.2 and rewrite ϕ'' as a formula of the form $\phi''' = \exists x_1.\exists x_2. \ldots \exists x_n.\alpha$. Until now, we have $\phi \equiv \phi' \equiv \phi'' \equiv \phi'''$. As the last step, we replace each variable x_k bound by a $\exists$ operator by a distinct nominal (denoted by i_{x_k}) not occurring in ϕ''', obtaining $\widetilde{\phi}$.

Now we prove that ϕ is satisfiable if and only if $\widetilde{\phi}$ is satisfiable.

($\Rightarrow$) Let $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ be a HCTL(@,$\downarrow$) model, g an assignment and v a state in $\mathcal{M}$ such that $\mathcal{M}, g, v \Vdash \phi$. As $\phi \equiv \phi'''$, $\mathcal{M}, g, v \Vdash \phi'''$. Since $\phi''' = \exists x_1.\exists x_2. \ldots \exists x_n.\alpha$, there is a tuple $(m_1, \ldots, m_n)$ of states such that $\mathcal{M}, g[m_1/x_1, \ldots, m_n/x_n], v \Vdash \alpha$. It follows that $\mathcal{M}', g, v \Vdash \widetilde{\phi}$, where $\mathcal{M}' =$

$(\mathcal{F}, \mathbf{V}')$ and $\mathbf{V}'$ differs from $\mathbf{V}$ only on the evaluation of the new nominals i_{x_k}, for which $\mathbf{V}'(i_{x_k}) = \{m_k\}$.

($\Leftarrow$) Let $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ be a HCTL(@,↓) model, g an assignment and v a state in $\mathcal{M}$ such that $\mathcal{M}, g, v \Vdash \widetilde{\phi}$. Hence, there is a tuple $(m_1, \ldots, m_n)$ of states, with $\mathbf{V}(i_{x_k}) = \{m_k\}$, such that $\mathcal{M}, g[m_1/x_1, \ldots, m_n/x_n], v \Vdash \alpha$. It follows that $\mathcal{M}, g, v \Vdash \exists x_1.\exists x_2. \ldots \exists x_n.\alpha$, that is $\mathcal{M}, g, v \Vdash \phi'''$. As $\phi''' \equiv \phi$, $\mathcal{M}, g, v \Vdash \phi$. □

Theorem 4.4. *The satisfiability problem for sentences in the reducible fragment is decidable, having EXPTIME-Complete complexity.*

Proof. EXPTIME-hardness follows from theorem 2.7. As for the upper-bound, it is not difficult to see that theorem 4.3 provides a method to polynomially transform a sentence ϕ of RF into a equi-satisfiable formula $\widetilde{\phi}$ of HCTL(@). From theorem 3.5, the satisfiability of $\widetilde{\phi}$ can be checked in EXPTIME, providing the desired upper-bound. □

5 Concluding Remarks

The logic HCTL(@,↓) is undecidable. In this paper, we present a fragment of this logic, called *reducible fragment* (RF), for which the satisfiability problem is decidable and has the same complexity as for standard CTL (EXPTIME-Complete). This fragment is built using a syntactic restriction on the formulas, which allows the satisfiability problem of RF to be reduced to the satisfiability problem of HCTL(@).

As our next steps, we want to use the Reduction Theorem as a tool to show the finite model property for the reducible fragment and to prove a weak completeness result for an axiomatic system for the co-reducible fragment.

It would also be interesting to investigate the complexity of the model-checking problem for HCTL(@,↓). Without any restriction, its complexity is PSPACE-hard [10]. As one of the reasons why CTL is a very attractive logic is exactly its polynomial-time model-checking complexity [6], it is interesting to research convenient fragments of HCTL(@,↓) in which the model-checking problem can be somehow simplified, similarly to what was done in this paper with respect to the satisfiability problem.

References

[1] C. Areces. *Logic Engineering: The Case of Description and Hybrid Logics.* PhD thesis, ILLC, University of Amsterdam, 2000.

[2] C. Areces, P. Blackburn, and M. Marx. A road-map on complexity for hybrid logics. In *Proceedings of the 8th Conference of the EACSL*, volume 1683 of *LNCS*, pages 307–321. Springer, 1999.

[3] C. Areces and B. ten Cate. Hybrid logics. In *Handbook of Modal Logics*, pages 821–868. Elsevier, 2006.

[4] P. Blackburn. Representation, reasoning, and relational structures: a hybrid logic manifesto. *Logic Journal of the IGPL*, 8(3):339–365, 2000.

[5] J. Bradfield and C. Stirling. Modal mu-calculi. In *Handbook of Modal Logic*, pages 721–756. Elsevier, 2006.

[6] E. M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 2000.

[7] E. A. Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics*, pages 995–1072. North-Holland Pub. Co./MIT Press, 1990.

[8] E. A. Emerson and E. M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Science of Computer Programming*, 2(3):241–266, 1982.

[9] E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computer and Systems Science*, 30(1):1–24, 1985.

[10] M. Franceschet and M. de Rijke. Model checking hybrid logics (with an application to semistructured data). *Journal of Applied Logic*, 4(3):279–304, 2006.

[11] M. Franceschet, M. de Rijke, and B. H. Schlingloff. Hybrid logic on linear structures: expressivity and complexity. In *Proceedings of the 10th Int. Symp. on Temporal Representation and Reasoning and 4th Int. Conf. on Temporal Logic*, pages 166–173. IEEE Computer Society Press, 2003.

[12] V. Goranko. Temporal logic with reference pointers. In *Proceedings of the 1st International Conference in Temporal Logic*, volume 827 of *LNAI*, pages 133–148. Springer, 1994.

[13] V. Goranko. Hierarchies of modal and temporal logics with reference pointers. *Journal of Logic, Language and Information*, 5(1):1–24, 1996.

[14] V. Goranko. Computation tree logics and temporal logics with reference pointers. *Journal of Applied Non-classical Logics*, 10(3–4):221–242, 2000.

[15] M. Mundhenk, T. Schneider, T. Schwentick, and V. Weber. Complexity of hybrid logics over transitive frames. In *Proceedings of the 4th M4M Workshop*, volume 194 of *Informatik-Berichte*, pages 62–78. Humboldt-Universität zu Berlin, 2005.

[16] U. Sattler and M. Y. Vardi. The hybrid μ-calculus. In *Proceedings of the 1st International Joint Conference in Automated Reasoning*, volume 2083 of *LNAI*, pages 76–91. Springer, 2001.

[17] B. ten Cate and M. Franceschet. On the complexity of hybrid logics with binders. In *Proceedings of the 19th International Workshop of Computer Science Logic*, volume 3634 of *LNCS*, pages 339–354. Springer, 2005.

[18] J. van Eijck. Constraint tableaux for hybrid logics. Available online in http://homepages.cwi.nl/∼jve/hylotab/, 2002.

[19] V. Weber. Hybrid branching-time logics. In *Proceedings of the International Workshop on Hybrid Logic 2007*, pages 51–60, 2007. Available online in http://folli.loria.fr/cds/2007/content/id27/id27.pdf.