Hybrid Logics on Linear Structures: Expressivity and Complexity

Massimo Franceschet
Department of Sciences, University of Chieti-Pescara, Italy
ILLC, University of Amsterdam, The Netherlands
E-mail: francesc@science.uva.nl

Maarten de Rijke
ILLC, University of Amsterdam, The Netherlands
E-mail: mdr@science.uva.nl

Bernd-Holger Schlingloff
Fraunhofer FIRST and Humboldt University, Berlin
E-mail: hs@informatik.hu-berlin.de

Abstract

We investigate expressivity and complexity of hybrid logics on linear structures. Hybrid logics are an enrichment of modal logics with certain first-order features which are algorithmically well behaved. Therefore, they are well suited for the specification of certain properties of computational systems. We show that hybrid logics are more expressive than usual modal and temporal logics on linear structures, and exhibit a hierarchy of hybrid languages. We determine the complexities of the satisfiability problem for these languages and define an existential fragment of hybrid logic for which satisfiability is still NP-complete. Finally, we examine the linear time model checking problem for hybrid logics and its complexity.

1 Introduction

Modal and temporal logics are algorithmically well-behaved and mathematically natural fragments of classical logics [6]. However, from the point of view of *reasoning about graphs*, something crucial is missing in the usual propositional modal and temporal logics: they lack mechanisms for *naming* states or sets of states, and for dynamically creating new names.

There is a good reason for the lack of such naming mechanisms in traditional modal and temporal logics: they are only able to express properties that satisfy the *tree model property*, i.e., properties that are satisfiable iff they are satisfiable in a tree-like model [6]. The ability to name states in a model violates the tree model property. Now, one can view the tree model property as a good feature for a logic, since it usually implies nice computational behavior, and as a bad feature, since it indicates a lack in expressivity. Are there natural extensions of modal and temporal logics with the naming facilities required by various modeling tasks (and thus violating the tree model property) that are still well-behaved from a computational point of view?

Hybrid logics provide a positive answer to the previous question. They allow reference to states in a modal framework, and, hence, mix features from first-order logic with features from modal logic, whence the name *hybrid* logic [7]. On top of ordinary propositional variables, hybrid languages have a type of atomic formulas called *nominals*. Syntactically, nominals behave like ordinary propositional variables, but they are names, true at exactly one state in any model. Hybrid languages may contain the *at operator* $@_i$ which gives 'random' access to the state named by i: $@_i\phi$ holds iff ϕ holds at the state named by i. They may also include the *name binder* $\downarrow x.$ which assigns the variable name x to the state of evaluation. Referencing states by $@$ combines naturally with naming them by $\downarrow$: The binder $\downarrow$ 'stores' the current state, and $@$ 'retrieves' the information stored. The *existential binder* $\exists x.$ binds the variable name x to some state in the model.

In this paper we investigate the expressivity and complexity of hybrid logics over *linear frames* (that is, irreflexive, transitive and trichotomous frames), and we compare our findings with the known results on general frames (that is, frames with no restriction). We show that, on linear

frames, basic hybrid logic (i.e., the logic extending propositional logic with nominals, @ operator, future **F** and past **P** temporal operators) is no more complex than propositional logic: its satisfiability problem is indeed NP-complete. The same complexity bounds hold over $(\mathbb{N}, <)$, the linear frame of the natural numbers with the usual ordering relation. The same logic over general structures is known to be EXPTIME-complete [1]. Whenever the name binder $\downarrow$ is added, the resulting hybrid logic is as expressive as first-order logic on linear structures, a result that fails in the case of general structures. As a consequence, its satisfiability problem is nonelementarily decidable. The same result holds over $(\mathbb{N}, <)$. Moreover, the same logic is undecidable on general structures [3]. If we omit the **P** and @ operators, the resulting logic (hence, the hybrid logic with nominals, $\downarrow$ and **F** only) has an NP-complete satisfiability problem over linear structures, whence over general structures it is still undecidable [3].

We furthermore isolate a large fragment of the nonelementary hybrid logic with hybrid operators @ and $\downarrow$, and temporal operators **F** and **P** (and their duals **G** and **H**). It is characterized by the fact that the name binder $\downarrow$ may not occur in the scope of universal temporal operators **G** and **H**. We show that the satisfiability problem for this fragment is NP-complete; hence it is basically not harder than propositional logic. As a corollary, we show NP-membership of a temporal logic equipped with a limited version of Until, Since and Next-time temporal operators, whereas temporal logic with either Until or Next-time is PSPACE-hard.

We finally investigate the linear time model checking problem for hybrid logics and give some examples of properties in which the use of nominals is crucial.

2 Hybrid logics

In this section we introduce hybrid logics and give some examples of hybrid formulas.

Definition 2.1 *Let* $\mathsf{PROP} = \{p_1, p_2, \ldots\}$ *be a countable set of* propositional variables, $\mathsf{NOM} = \{i_1, i_2, \ldots\}$ *a countable set of* nominals, *and* $\mathsf{WVAR} = \{x_1, x_2, \ldots\}$ *a countable set of* state variables. *We assume that* PROP, NOM *and* WVAR *are pairwise disjoint. We call* $\mathsf{WSYM} = \mathsf{NOM} \cup \mathsf{WVAR}$ *the set of* state symbols, $\mathsf{ALET} = \mathsf{PROP} \cup \mathsf{NOM}$ *the set of* atomic letters, *and* $\mathsf{ATOM} = \mathsf{PROP} \cup \mathsf{NOM} \cup \mathsf{WVAR}$ *the set of* atoms. *The* well-formed formulas *of the hybrid language* $\mathrm{HL}(@, \downarrow, \exists, \mathbf{F}, \mathbf{P}, \mathbf{U}, \mathbf{S})$ *(over the signature* $\langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{WVAR} \rangle$*) are given by the rule*

$$\begin{aligned} \varphi := \quad & \bot \mid a \mid (\varphi \to \varphi') \\ & \mid \mathbf{F}\varphi \mid \mathbf{P}\varphi \mid (\varphi \mathbf{U} \varphi') \mid (\varphi \mathbf{S} \varphi') \\ & \mid @_s\varphi \mid (\downarrow x_j.\varphi) \mid (\exists x_j.\varphi) \end{aligned}$$

where $a \in \mathsf{ATOM}$, $x_j \in \mathsf{WVAR}$ *and* $s \in \mathsf{WSYM}$.

In formulas, we will omit parenthesis whenever appropriate. Furthermore, we assume all boolean operators are defined as usual. **U** and **S** are the Until and Since operators, respectively. As usual, $\mathbf{G}\varphi$ is short for $\neg\mathbf{F}\neg\varphi$ and $\mathbf{H}\varphi$ for $\neg\mathbf{P}\neg\varphi$. Moreover, we define $\mathbf{E}\varphi$ as $(\mathbf{P}\varphi \vee \varphi \vee \mathbf{F}\varphi)$ and $\mathbf{A}\varphi$ as $\neg\mathbf{E}\neg\varphi$. On linear frames, **E** and **A** are the existential and universal modality, respectively.

The notions of *free* and *bound* state variable (with respect to the binding operator $\downarrow$) are obvious generalizations from first-order logic. Other syntactic notions (such as *substitution*, and of a state symbol t being *substitutable for* x *in* φ) are defined like the corresponding notions in first-order logic. We write $\varphi[t/s]$ for the formula obtained by replacing all free instances of the state symbol t by the state symbol s. A *sentence* is a formula without free state variables. A formula is *pure* if it contains no propositional variables, and *nominal-free* if it contains no nominals.

Definition 2.2 *A* hybrid model $\mathcal{M}$ *for the full hybrid language is a triple* $\mathcal{M} = \langle M, R, V \rangle$ *with* M *is a non-empty set,* R *a binary relation on* M*, and* $V : \mathsf{ALET} \to Pow(M)$ *such that for all nominals* $i \in \mathsf{NOM}$, $V(i)$ *is a singleton. (We use calligraphic letters* $\mathcal{M}$ *for models, italic roman* M *for their domains.) We call the elements of* M states, R *the* accessibility relation *and* V *the* valuation.

An assignment g *for* $\mathcal{M}$ *is a mapping* $g : \mathsf{WVAR} \to M$. *Given an assignment* g*, we define* g^x_m *(*an x-variant of g*) by* $g^x_m(x) = m$ *and* $g^x_m(y) = g(y)$ *for* $x \neq y$.

Let $\mathcal{M} = \langle M, R, V \rangle$ *be a model,* $m \in M$*, and* g *an assignment. For any atom* a*, let* $[V, g](a) = \{g(a)\}$ *if* a *is a state variable, and* $V(a)$ *otherwise. The* satisfaction relation *is defined as follows (we omit the clauses for the Booleans and for the past temporal operators):*

$$\begin{aligned} &\mathcal{M}, g, m \Vdash a && \textit{iff} \quad m \in [V, g](a) \quad (a \in \mathsf{ATOM}) \\ &\mathcal{M}, g, m \Vdash \mathbf{F}\varphi && \textit{iff} \quad \exists m' \, (Rmm' \wedge \mathcal{M}, g, m' \Vdash \varphi) \\ &\mathcal{M}, g, m \Vdash \psi \mathbf{U} \varphi && \textit{iff} \quad \exists m' \, (Rmm' \wedge \mathcal{M}, g, m' \Vdash \varphi \\ &&& \quad \wedge \forall m'' \, (Rmm'' \wedge Rm''m' \to \mathcal{M}, g, m'' \Vdash \psi)) \\ &\mathcal{M}, g, m \Vdash @_s\varphi && \textit{iff} \quad \mathcal{M}, g, m' \Vdash \varphi, \textit{ where} \\ &&& \quad [V, g](s) = \{m'\} \ (s \in \mathsf{WSYM}) \\ &\mathcal{M}, g, m \Vdash \downarrow x.\varphi && \textit{iff} \quad \mathcal{M}, g^x_m, m \Vdash \varphi \\ &\mathcal{M}, g, m \Vdash \exists x.\varphi && \textit{iff} \quad \exists m' (\mathcal{M}, g^x_{m'}, m \Vdash \varphi) \end{aligned}$$

A formula φ *is* satisfiable *if there is a model* $\mathcal{M}$*, an assignment* g *on* $\mathcal{M}$*, and a state* $m \in M$ *such that* $\mathcal{M}, g, m \Vdash \varphi$. *A formula* φ *is* valid *if* $\neg\varphi$ *is not satisfiable.*

The *at* operator $@_s$ shifts evaluation to the state named by s. The *name binder* $\downarrow x$. ("call it x, and ...") binds the state variable x to the *current* state, and the *existential binder* $\exists x$. ("some state is called x, and ...") binds the state variable x to *some* state in the model. Both $\downarrow$ and $\exists$ do not shift evaluation away from the current state.

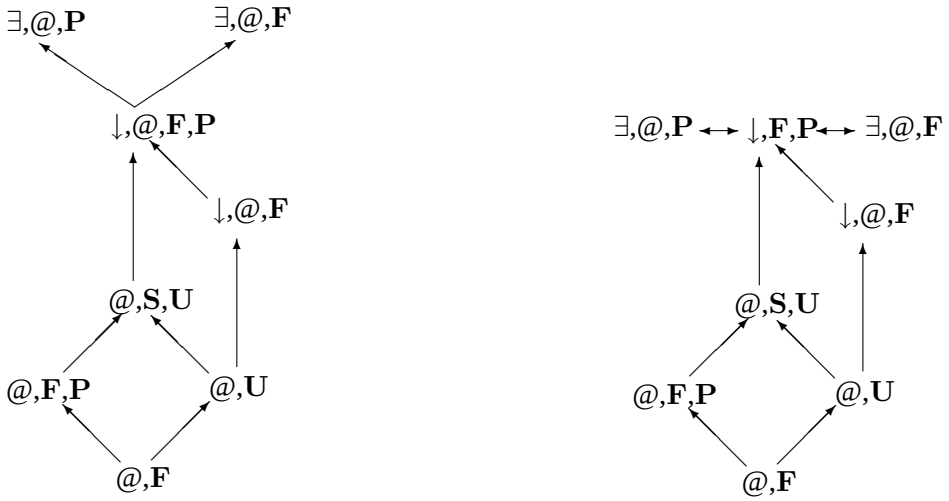

Figure 1. The hybrid hierarchy on general frames (left side) and on linear frames (right side)

A model $\mathcal{M} = \langle M, R, V\rangle$ is called *linear* if the relation R is irreflexive (i.e., $\forall x(\neg Rxx)$), transitive (i.e., $\forall xyz(Rxy \wedge Ryz \rightarrow Rxz)$) and trichotomous (i.e., $\forall xy(Rxy \vee x = y \vee Ryx)$).

The language of hybrid logic has a great expressive power, especially compared to its frugal syntax and perspicuous semantics. We give some examples of its expressive power. On linear models, $\mathbf{A}{\downarrow}x.\mathbf{F}\neg\mathbf{P}\mathbf{P}x$ defines discreteness, which cannot be defined in the temporal logic of future $\mathbf{F}$ and past $\mathbf{P}$.

Kamp's temporal operator Until can be defined in terms of $\{@, \downarrow, \mathbf{F}\}$ as follows:

$$\alpha\mathbf{U}\beta = {\downarrow}x.\mathbf{F}{\downarrow}y.@_x(\mathbf{F}(y \wedge \beta) \wedge \mathbf{G}(\mathbf{F}y \rightarrow \alpha)).$$

An alternative definition using past $\mathbf{P}$ instead of $@$ is the following:

$$\alpha\mathbf{U}\beta = {\downarrow}x.\mathbf{F}(\beta \wedge \mathbf{H}(\mathbf{P}x \rightarrow \alpha)).$$

Similar definitions can be given for the temporal operator Since.

The dual of the Until is called Release and denoted by $\mathbf{R}$: $\alpha\mathbf{R}\beta = \neg(\neg\alpha\mathbf{U}\neg\beta)$. It expresses the fact that β must hold up to and including the first future state where α holds, or indefinitely if there is no such state. The dual of Since is called Trigger and denoted by $\mathbf{T}$: $\alpha\mathbf{T}\beta = \neg(\neg\alpha\mathbf{S}\neg\beta)$. Both the Release and Trigger operators can be concisely expressed in terms of $\{\downarrow, \mathbf{F}, \mathbf{P}\}$. For instance,

$$\alpha\mathbf{R}\beta = {\downarrow}x.\mathbf{G}(\beta \vee \mathbf{P}(\mathbf{P}x \wedge \alpha)).$$

Finally, Stavi's extended Until and Since operators can also be expressed in hybrid logic. For instance, Stavi's $\mathbf{U}'$ is captured as follows:

$$\begin{aligned}\alpha\mathbf{U}'\beta \;=\; & {\downarrow}x.\mathbf{F}\mathbf{H}(\mathbf{P}x \rightarrow \alpha) \wedge \\ & \neg\alpha\mathbf{R}(\alpha \wedge \alpha\mathbf{U}\top) \wedge \\ & {\downarrow}x.\mathbf{F}(\neg\alpha \wedge \beta \wedge \\ & \quad \mathbf{H}(\mathbf{P}x \wedge \mathbf{P}(\mathbf{P}x \wedge \neg\alpha) \rightarrow \beta)).\end{aligned}$$

This operator cannot be expressed in linear temporal logic with $\mathbf{U}$ and $\mathbf{S}$ (see [11]).

3 Expressivity and complexity on linear frames

In this section we investigate the expressive power of hybrid logics on linear structures. Moreover, we study the computational complexity of the satisfiability and model checking problems for hybrid logics on linear frames.

3.1 Expressivity

Figure 1 summarizes how the expressive powers of the various hybrid languages are related, both on linear (right side) and general structures (left side). To increase readability, we omit the HL prefix and write, e.g., ∃,@,**F** instead of $\mathrm{HL}(\exists, @, \mathbf{F})$. Arrows point from less expressive to more expressive languages. We only prove the following expressivity result:

Theorem 3.1 *The languages* $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$, $\mathrm{HL}(\exists, @, \mathbf{F})$, *and* $\mathrm{HL}(\exists, @, \mathbf{P})$ *are all as expressive as monadic first-order logic over linear structures.*

Proof. We proceed as follows. We first show that $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$, $\mathrm{HL}(\exists, @, \mathbf{F})$, and $\mathrm{HL}(\exists, @, \mathbf{P})$ have all the same expressive power. Then, we show that $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$ is as expressive as first-order logic.

Recall that on linear frames, $\mathbf{E}$ acts as the *existential modality*, that is, $\mathbf{E}\varphi$ is true at m iff there exists some world m' in the model such that φ is true at m'.

We prove the following embedding loop: $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P}) \Rightarrow \mathrm{HL}(\exists, @, \mathbf{F}) \Rightarrow \mathrm{HL}(\exists, @, \mathbf{P}) \Rightarrow \mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$. We show $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P}) \Rightarrow \mathrm{HL}(\exists, @, \mathbf{F})$. We encode the past temporal operator $\mathbf{P}$ as follows: $\mathbf{P}\varphi = \exists x(x \wedge \exists y.@_y(\mathbf{F}x \wedge \varphi))$. We encode the hybrid binder $\downarrow$ as follows: $\downarrow x.\varphi = \exists x.(x \wedge \varphi)$. We show $\mathrm{HL}(\exists, @, \mathbf{F}) \Rightarrow \mathrm{HL}(\exists, @, \mathbf{P})$. We encode the future temporal operator $\mathbf{F}$ as follows: $\mathbf{F}\varphi = \exists x(x \wedge \exists y.@_y(\mathbf{P}x \wedge \varphi))$. We finally show $\mathrm{HL}(\exists, @, \mathbf{P}) \Rightarrow \mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$. We encode the hybrid binder $\exists$ as follows: $\exists x.\varphi = \downarrow y.\mathbf{E}\downarrow x.@_y\varphi$, and we remove the @ operator as follows: $@_i\varphi = \mathbf{E}(i \wedge \varphi)$.

We now show that $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$ is as expressive as monadic first-order logic. This completes the proof of the theorem. To be more precise, the first-order language under consideration contains *equality*, a binary predicate R, a unary predicate P_j for each $p_j \in \mathsf{PROP}$, and whose *constants* are the elements of NOM. We first show that $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$ is a fragment of this first-order logic. To see this, recall that the *standard translation* is an embedding of modal logic into monadic first-order logic [6]; it can easily be extended to hybrid logics. Clearly, a hybrid model can be regarded as a first-order model for this language and vice versa. The translation ST from the hybrid language $\mathrm{HL}(\exists, \downarrow, @, \mathbf{F}, \mathbf{P})$ into first-order correspondence logic is defined by mutual recursion between two functions ST_x and ST_y (we only give the clauses for ST_x; the ones for ST_y are completely analogous; the Boolean cases are left out):

$$
\begin{array}{lcl}
ST_x(p_j) & = & P_j(x),\ p_j \in \mathsf{PROP} \\
ST_x(i_j) & = & (x = i_j),\ i_j \in \mathsf{NOM} \\
ST_x(x_j) & = & (x = x_j),\ x_j \in \mathsf{WVAR} \\
ST_x(\mathbf{F}\varphi) & = & \exists y\,(Rxy \wedge ST_y(\varphi)) \\
ST_x(\mathbf{P}\varphi) & = & \exists y\,(Ryx \wedge ST_y(\varphi)) \\
ST_x(@_t\varphi) & = & (ST_x(\varphi))[x/t] \\
ST_x(\downarrow x_j.\varphi) & = & \exists x_j.(x = x_j \wedge ST_x(\varphi)) \\
ST_x(\exists x_j.\varphi) & = & \exists x_j.ST_x(\varphi)
\end{array}
$$

Finally we encode our first-order logic into $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$ as follows. Recall that $\mathbf{E}\varphi$ is defined as $(\mathbf{P}\varphi \vee \varphi \vee \mathbf{F}\varphi)$.

$$
\begin{array}{lcl}
\tau(x = y) & = & \mathbf{E}(x \wedge y) \\
\tau(Rxy) & = & \mathbf{E}(x \wedge \mathbf{F}y) \\
\tau(P(x)) & = & \mathbf{E}(x \wedge p) \\
\tau(\alpha \to \beta) & = & \tau(\alpha) \to \tau(\beta) \\
\tau(\bot) & = & \bot \\
\tau(\exists x.\alpha) & = & \downarrow y.\mathbf{E}\downarrow x.\mathbf{E}(y \wedge \tau(\alpha))
\end{array}
$$

⊣

The above result is more general: it holds on any class of frames such that the existential modality $\mathbf{E}$ can be defined in $\mathrm{HL}(\downarrow, \mathbf{F}, \mathbf{P})$. An interesting example is the class of transitive trees. On transitive trees, $\mathbf{E}\varphi = (\varphi \vee \mathbf{P}\varphi \vee \mathbf{PF}\varphi)$. However, the above result does not hold on the class of any frames: on general frames, $\mathrm{HL}(\downarrow, @, \mathbf{F}, \mathbf{P})$ is as expressive as the *bounded fragment* of first-order logic, a strict subfragment of first-order logic [3].

3.2 Complexity of the satisfiability problem

In [2], the authors notice that on linear frames, we can get rid of nominals and @ as soon as we have at disposal (strict) past and future temporal operators. Indeed, we can simulate nominals by singleton propositions, that is propositions true at exactly one state: i is a singleton proposition iff $\mathbf{E}(i \wedge \mathbf{H}\neg i \wedge \mathbf{G}\neg i)$ holds. Moreover, $@_i\varphi$ can be expressed as $\mathbf{E}(i \wedge \varphi)$ or $\mathbf{A}(i \to \varphi)$. It follows that there is a translation of hybrid formulas in $\mathrm{HL}(@, \mathbf{F}, \mathbf{P})$ into temporal formulas in $\mathrm{TL}(\mathbf{F}, \mathbf{P})$ preserving equi-satisfiability. Moreover, they show that there is a *polynomial translation* with the same features. Hence, the satisfiability problem for $\mathrm{HL}(@, \mathbf{F}, \mathbf{P})$ can be reduced to the same problem for $\mathrm{TL}(\mathbf{F}, \mathbf{P})$, which is known to be decidable in nondeterministic polynomial time [14]. NP-hardness for $\mathrm{HL}(@, \mathbf{F}, \mathbf{P})$ holds since it extends Propositional Calculus.

Theorem 3.2 *The satisfiability problem for* $\mathrm{HL}(@, \mathbf{F}, \mathbf{P})$ *on linear frames is NP-complete.*

The same complexity bounds for $\mathrm{HL}(@, \mathbf{F}, \mathbf{P})$ hold over $(\mathbb{N}, <)$, the linear frame of the natural numbers with the usual ordering relation. The proof is the same and $\mathrm{TL}(\mathbf{F}, \mathbf{P})$ over natural numbers is NP-complete, a recent result proved in [13]. However, $\mathrm{HL}(@, \mathbf{F}, \mathbf{P})$ on general structures has higher complexity: its satisfiability problem is known to be EXPTIME-complete [1]. In the proof, only one nominal is used.

If we replace future $\mathbf{F}$ and past $\mathbf{P}$ temporal operators by Until $\mathbf{U}$ and Since $\mathbf{S}$, respectively, the satisfiability problem on natural numbers is harder.

Theorem 3.3 *The satisfiability problem for* $\mathrm{HL}(@, \mathbf{U}, \mathbf{S})$ *on natural numbers is PSPACE-complete.*

Proof. PSPACE-hardness holds since $\mathrm{TL}(\mathbf{U})$ on natural numbers is already PSPACE-hard [16]. Moreover, nominals and @ operator may be removed from $\mathrm{HL}(@, \mathbf{U}, \mathbf{S})$, as shown above, by taking advantage of past $\mathbf{P}$ and future $\mathbf{F}$ operators, which can be defined in terms of Until $\mathbf{U}$ and Since $\mathbf{S}$, respectively. It follows that the linear time satisfiability problem for $\mathrm{HL}(@, \mathbf{U}, \mathbf{S})$ can be embedded into the same problem for $\mathrm{TL}(\mathbf{U}, \mathbf{S})$, which is decidable in PSPACE [16]. ⊣

We do not know whether the same result works on linear frames. In [15] the author proves that, on linear structures,

the temporal logic of Until is PSPACE-complete, and he conjectures the same result for the temporal logic with Until and Since. Hence, on linear frames, $\mathrm{HL}(@, \mathbf{U}, \mathbf{S})$ is at least PSPACE-hard. The situation on general structures is the following: both $\mathrm{HL}(@, \mathbf{U})$ and $\mathrm{HL}(@, \mathbf{U}, \mathbf{S})$ are complete for EXPTIME, and the proof uses only one nominal [1]

We now consider the addition of the hybrid binder $\downarrow$. We already showed that, on linear frames, $\mathrm{HL}(\downarrow, @, \mathbf{F}, \mathbf{P})$ is as expressive as first-order logic in the correspondence language with free monadic predicates. Universal (and existential) monadic second order logic over linear frames is decidable [12]. Furthermore, full monadic second-order logic over natural numbers is nonelementarily decidable [8]. The following theorem follows.

Theorem 3.4 *The satisfiability problem for* $\mathrm{HL}(\downarrow, @, \mathbf{F}, \mathbf{P})$ *on linear frames and on natural numbers is nonelementarily decidable.*

The situation on the class of any frame is even worse: already $\mathrm{HL}(\downarrow, \mathbf{F})$ is undecidable, even without nominals and propositions [3]. The latter neatly contrasts with the situation on linear frames: $\mathrm{HL}(\downarrow, \mathbf{F})$ is decidable in NP.

Theorem 3.5 *The satisfiability problem for* $\mathrm{HL}(\downarrow, \mathbf{F})$ *on linear structures is NP-complete.*

Proof. We can get rid of $\downarrow$ from $\mathrm{HL}(\downarrow, \mathbf{F})$ as follows: given a formula $\downarrow x.\varphi$, any instance of x appearing in φ in the scope of an $\mathbf{F}$ operator evaluates to false, since a linear structure has no loop and there are no past or @ operators that can jump back to it. It hence may be replaced by $\bot$, that is, by $\neg\top$, without changing the meaning of the formula. Moreover, any instance of x appearing in φ not in the scope of an $\mathbf{F}$ operator evaluates to true, since it refers to the current point of evaluation. It may be replaced by $\top$ without changing the meaning of the formula. Finally the $\downarrow$ binder may be removed. The resulting is an equivalent formula in $\mathrm{HL}(\mathbf{F})$, which is decidable in NP by Theorem 3.2. NP-hardness holds since $\mathrm{HL}(\downarrow, \mathbf{F})$ extends Propositional Calculus. $\dashv$

It turns out that, on linear structures, $\downarrow$ is a kind of *bully* operator: it shows its strength only in presence of @ and $\mathbf{P}$. The reason is clear: since linear structures are acyclic, the only way to access a variable stored by $\downarrow$ is by using either the @ or the $\mathbf{P}$ operator. Hence, the power of $\downarrow$ is tamed without them.

Our aim in the following is to isolate existential fragments of $\mathrm{HL}(\exists, @, \mathbf{F}, \mathbf{P})$ and $\mathrm{HL}(\downarrow, @, \mathbf{F}, \mathbf{P})$ with nice computational behaviour. The *existential hybrid logic* $\mathrm{EHL}(\exists, @, \mathbf{F}, \mathbf{G}, \mathbf{P}, \mathbf{H})$ is obtained from $\mathrm{HL}(\exists, @, \mathbf{F}, \mathbf{G}, \mathbf{P}, \mathbf{H})$ by (1) allowing formulas in negation normal form only (which means that negation in ap-

φ_1. Since φ_1 has the form $\exists x_1 \ldots \exists x_n.\alpha$, there is a tuple $m_1, \ldots, m_n \in M^n$ such that $\mathcal{M}, g', m \Vdash \alpha$, where $g' = g[x_1/m_1, \ldots, x_n/m_n]$. It follows that $\mathcal{M}', g, m \Vdash \varphi_2$, where $\mathcal{M}' = \langle M, R, V' \rangle$, and V' differs from V only on the evaluation of the new nominals i_{x_j}, for which $V'(i_{x_j}) = \{m_j\}$.

For the right to left direction. Let $\mathcal{M} = \langle M, R, V \rangle$ be a hybrid model, g be an assignment, and m be a world in M such that $\mathcal{M}, g, m \Vdash \varphi_2$. Hence there is a tuple $m_1, \ldots, m_n \in M^n$, with $V(i_{x_j}) = \{m_j\}$, such that $\mathcal{M}, g', m \Vdash \alpha$, where $g' = g[x_1/m_1, \ldots, x_n/m_n]$. It follows that $\mathcal{M}, g, m \Vdash \exists x_1 \ldots \exists x_n.\alpha$, that is, $\mathcal{M}, g, m \Vdash \varphi_1$. ⊣

A nice corollary of Theorem 3.6 is the following. We know that temporal logic with future and past operators admits an NP-complete satisfiability problem, and we remain in NP if we add nominals and the @ operator. However, as soon as we add either Until or Since temporal operators, we jump up into PSPACE. Nevertheless, if we manage these operators with care, we don't leave NP.

Corollary 3.7 *Let* `LiteLTL` *be the fragment of* $\mathrm{HL}(@, \mathbf{F}, \mathbf{G}, \mathbf{P}, \mathbf{H}, \mathbf{U}, \mathbf{S})$ *such that*

1. $\neg$ *is applied over atomic symbols only;*
2. $\mathbf{U}$ *and* $\mathbf{S}$ *are not allowed in the scope of* $\mathbf{G}$ *and* $\mathbf{H}$*;*
3. *formulas* $\alpha\mathbf{U}\beta$ *and* $\alpha\mathbf{S}\beta$ *are such that* $\mathbf{U}$ *and* $\mathbf{S}$ *are not allowed in* α*.*

Then, the satisfiability problem for `LiteLTL` *on linear frames is NP-complete.*

Proof. Recall the $\mathbf{U}$ and $\mathbf{S}$ can be defined in terms of $\{\downarrow, @, \mathbf{F}, \mathbf{P}\}$ as follows:

$$\begin{aligned} \alpha\mathbf{U}\beta &= \downarrow x.\mathbf{F}\downarrow y.@_x(\mathbf{F}(y \wedge \beta) \wedge \mathbf{G}(\mathbf{F}y \rightarrow \alpha)) \\ \alpha\mathbf{S}\beta &= \downarrow x.\mathbf{P}\downarrow y.@_x(\mathbf{P}(y \wedge \beta) \wedge \mathbf{H}(\mathbf{P}y \rightarrow \alpha)) \end{aligned}$$

Hence in $\alpha\mathbf{U}\beta$, only α is in the scope of an universal temporal operator. It follows that `LiteLTL` is a fragment of $\mathrm{EHL}(\downarrow, @, \mathbf{F}, \mathbf{G}, \mathbf{P}, \mathbf{H})$, whose satisfiability problem is in NP by Theorem 3.6. ⊣

In the above fragment you can, for instance, write properties like "p will hold (held) exactly n times in the future (past)", which are beyond the expressive power of $\mathrm{HL}(@, \mathbf{F}, \mathbf{P})$. For instance, the `LiteLTL`-formula that follows claims that p will hold exactly 2 times in the future:

$$\neg p\mathbf{U}(p \wedge \neg p\mathbf{U}(p \wedge \mathbf{G}\neg p))$$

A limited form of Next-time and Previous-time operator is also allowed in `LiteLTL` (these operators make sense only on discrete linear frames, like natural numbers). We define the Next-time operator $\mathbf{X}\varphi$ as $\bot\mathbf{U}\varphi$ and the Previous-time operator $\mathbf{Y}\varphi$ as $\bot\mathbf{S}\varphi$. We are allowed to nest $\mathbf{X}$, $\mathbf{Y}$ and all the other temporal operators in the scope of $\mathbf{X}$ and $\mathbf{Y}$. Moreover, $\mathbf{X}$ and $\mathbf{Y}$ are allowed in the scope of existential temporal operators $\mathbf{F}$ and $\mathbf{P}$ and in the scope of temporal operators $\mathbf{U}$ and $\mathbf{S}$ as soon as they appear only in the existential part of them (that is, in β, whenever the formula is $\alpha\mathbf{U}\beta$ or $\alpha\mathbf{S}\beta$). All the rest is prohibited. For example, we can write the property "p will hold until q will hold continuously for 3 times" as

$$p\mathbf{U}(q \wedge \mathbf{X}q \wedge \mathbf{X}\mathbf{X}q).$$

Release and Trigger operators, as well as Stavi's Until and Since, are also allowed in `LiteLTL`, as soon as we manage them with care. These operators can be nested only in the scope of existential unary temporal operators and in the scope of the existential part of Kamp's Since and Until. Moreover, any binary temporal operator is not allowed in the scope of them.

It is worth remarking that, on linear frames, the temporal logic with Until is PSPACE-hard [15]. The same result holds on natural numbers [16]. Moreover, on natural numbers, the temporal logic with future and Next-time is PSPACE-hard too [16]. A closer analysis of the latter two results shows that both the fragments of $\mathrm{TL}(\mathbf{F}, \mathbf{X})$ and $\mathrm{TL}(\mathbf{U})$ in which the temporal operators are not nested are in NP [10]. However, already $\mathrm{TL}(\mathbf{U})$-formulas with temporal height 2 are enough to encode QBF, and hence the resulting fragment is PSPACE-hard. Similarly, a bounded temporal nesting in $\mathrm{TL}(\mathbf{F}, \mathbf{X})$ is enough to encode QBF [10]. It follows that the linear temporal logic `LiteLTL` lies in NP but "very close" to PSPACE.

The reader should wonder about the practical usefulness of decreasing a complexity bound from PSPACE to NP. Computationally, NP is still intractable. However, problems in NP may be polynomially reduced to SAT, the popular NP-complete problem, for which many heuristic solvers have been implemented. Problems in PSPACE, on the contrary, cannot be reduced to SAT, unless NP = PSPACE. Techniques for embedding the model checking (and satisfiability) problem for linear time logics into SAT are described and implemented in [5, 9]. These techniques have been recently extended to cope with past temporal operators [4].

3.3 The linear time model checking problem

Model checking is a generic term for a class of algorithms which determine whether a given formula holds in a given model or class of models. Often a Kripke structure denotes some computational system, and paths through the system denote computations. Hence, in linear time model

checking formulas are evaluated not on the Kripke structure itself, but on the set of paths through it.

The hybrid Kripke structure $\mathcal{M} = \langle M, R, V\rangle$ is *total* if every state in M has at least one R-successor. In this section we will consider only total and finite Kripke structures. A *path* from s_0 in $\mathcal{M}$ is an infinite state sequence $\pi = s_0, s_1, \ldots$ such that Rs_is_{i+1} for every $i \geq 0$. We denote by π_i the i-th state s_i of π. Any path in $\mathcal{M}$ can be naturally associated to a linear structure $\mathcal{M}_\pi = \langle \mathbb{N}, <, V'\rangle$ such that $\mathbb{N}$ is the set of natural numbers, $<$ is the usual ordering relation on the natural numbers and, for every $i \geq 0$ and $p \in \mathsf{ALET}$, $i \in V'(p)$ iff $\pi_i \in V(p)$. Notice that $\mathcal{M}_\pi$ is not necessarily a hybrid structure, since, because of the unfolding process, the same nominal may label different states of $\mathcal{M}_\pi$. The meaning of formulas of the form $@_i\alpha$, where i is a nominal, is hence ambiguous. There are several possibilities to deal with this situation. Our choice here is to consider nominals as additional propositions which must be true at exactly one state of the branching structure $\mathcal{M}$ but may be true at several states of the linear structure $\mathcal{M}_\pi$ obtained by unfolding the path π of $\mathcal{M}$. The meaning of the formula $@_i\alpha$ is then "α holds at *some* state labelled with i". That is, $\mathcal{M}_\pi, m \Vdash @_i\alpha$ iff $\mathcal{M}_\pi, m \Vdash \mathbf{E}(i \wedge \alpha)$.

We are now ready to define the linear time model checking problem for hybrid logics. We distinguish between the existential and the universal version of the model checking problem. The *existential linear time model checking problem* for hybrid logics is to determine whether a given hybrid formula is true in *some* path of the model: $\mathcal{M}, m \models_\exists \varphi$ iff $\mathcal{M}_\pi, 0 \Vdash \varphi$ for some path π starting at m in $\mathcal{M}$. Moreover, $\mathcal{M} \models_\exists \varphi$ iff $\mathcal{M}, m \models_\exists \varphi$ for some $m \in \mathcal{M}$. The *universal linear time model checking problem* for hybrid logics is to determine whether a given hybrid formula is true in *every* path of the model: $\mathcal{M}, m \models_\forall \varphi$ iff $\mathcal{M}_\pi, 0 \Vdash \varphi$ for every path π starting at m in $\mathcal{M}$. Moreover, $\mathcal{M} \models_\forall \varphi$ iff $\mathcal{M}, m \models_\forall \varphi$ for every $m \in \mathcal{M}$. Notice that the universal model checking problem is the dual of the existential one: $\mathcal{M}, m \models_\forall \varphi$ iff it is not the case that $\mathcal{M}, m \models_\exists \neg\varphi$.

The existential (respectively, universal) linear time model checking problem for $\mathrm{TL}(\mathbf{F}, \mathbf{P})$ has been recently proved to be NP-complete (respectively, coNP-complete) [13]. The following follows:

Theorem 3.8 *The existential (respectively, universal) linear time model checking problem for* $\mathrm{HL}(@, \mathbf{F}, \mathbf{P})$ *is NP-complete (respectively, coNP-complete).*

Moreover, it is well-known that both the existential and the universal linear time model checking problems for $\mathrm{TL}(\mathbf{F}, \mathbf{P}, \mathbf{U}, \mathbf{S})$ are PSPACE-complete [16]. Hence, we have the following:

Theorem 3.9 *Both the existential and universal linear time model checking problem for* $\mathrm{HL}(@, \mathbf{F}, \mathbf{P}, \mathbf{U}, \mathbf{S})$ *is PSPACE-complete.*

We conclude this section by giving some model checking examples involving nominals. Let the nominal *Start* designate the *unique* initial state in a system modelled by $\mathcal{M}$. Then the check

$$\mathcal{M}, Start \models_\forall \mathbf{F}Start$$

is true iff "each computation of the system starting at the initial state will eventually return to the initial state". This implies that the initial state will be visited infinitely often in every computation of the system. Since *Start* is a nominal, it is true at exactly one state in $\mathcal{M}$ (the initial state), but it may be true at several states in the computation paths in $\mathcal{M}$. If *Start* were a proposition, instead of a nominal, then the above check would be true iff "each computation of the system starting at a state labelled with *Start* will eventually reach a (possibly different) state labelled with *Start*". The check

$$\mathcal{M}, Start \models_\forall \mathbf{G}\neg Start$$

is true iff "each computation of the system starting at the initial state will never return to the initial state". Finally the check

$$\mathcal{M}, Start \models_\exists (\neg Start)\mathbf{U}(Start \wedge \mathbf{G}\neg Start)$$

is true iff "there is a computation of the system starting at the initial state that will return to the initial state exactly once". These examples show that it can can be of advantage to use nominals in specifications.

4 Conclusion

In this paper, we have analyzed the expressivity and complexity of several variants of hybrid logic on linear structures. There are a number of open questions for further work. Firstly, there is the question of second order extensions of these languages, e.g., by fixpoint operators or propositional quantifiers. Secondly, it would be interesting to find a generic format for hybrid specifications, similar as it is TLA (Lamport's temporal logic of actions) for linear temporal logic. Thirdly, we want to apply hybrid logic in the specification of an industrial application (an electronic funds transfer / point of sale banking system). A challenge is to find a way to combine the specification of various spatial and temporal properties such that the resulting formulas are still tractable. An interesting project in this context is to derive an intuitive high-level specification language which can be mapped into the hybrid framework and allows to formulate correctness properties without detailled knowledge of the underlying logic.

References

[1] C. Areces, P. Blackburn, and M. Marx. A road-map on complexity for hybrid logics. In J. Flum and M. Rodriguez-Artalejo, editors, *Computer Science Logic*, volume 1683 of *LNCS*, pages 307–321. Springer, 1999.

[2] C. Areces, P. Blackburn, and M. Marx. The computational complexity of hybrid temporal logics. *Logic Journal of the IGPL*, 8(5):653–679, 2000.

[3] C. Areces, P. Blackburn, and M. Marx. Hybrid logics: Characterization, interpolation, and complexity. *Journal of Symbolic Logic*, 66(3):977–1010, 2001.

[4] M. Benedetti and A. Cimatti. Bounded model checking for past LTL. In *Proceedings of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2003.

[5] A. Biere, A. Cimatti, E. M. Clarke, M. Fujita, and Y. Zhu. Symbolic model checking using SAT procedures instead of BDDs. In *Proceedings of Design Automation Conference*, volume 1579 of *Lectures Notes in Computer Science*, pages 193–207, 1999.

[6] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 2001.

[7] P. Blackburn and J. Seligman. What are hybrid languages? In M. Kracht, M. de Rijke, H. Wansing, and M. Zakharyaschev, editors, *Advances in Modal Logic*, *Volume 1*, pages 41–62. CSLI Publications, 1998.

[8] J. R. Büchi. On a decision method in restricted second-order arithmetic. In *Proceedings of the International Congress on Logic, Methodology, and Philosophy of Science*, pages 1–11. Stanford University Press, 1962.

[9] A. Cimatti, M. Pistore, M. Roveri, and R. Sebastiani. Improving the enconding of LTL model checking into SAT. In *Proceedings of the Workshop on Verification Model Checking and Abstract Interpretation (VMCAI 2002)*, volume 2294 of *Lectures Notes in Computer Science*, 2002.

[10] S. Demri and P. Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *Information and Computation*, 174(1):84–103, 2002.

[11] D. Gabbay, I. Hodkinson, and M. Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects*. Oxford logic guides, Vol. 28. Oxford University Press, 1994.

[12] Y. Gurevich. Elementary properties of ordered abelian groups. *Algebra and Logic*, 3(1):5–39, 1964. Russian.

[13] N. Markey. Past is for free: on the complexity of verifying linear temporal properties with past. In *Proceedings of the International Workshop on Expressiveness in Concurrency (EXPRESS'2002)*, volume 68.2 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science, 2002.

[14] H. Ono and A. Nakamura. On the size of refutation Kripke models for some linear modal and tense logics. *Studia Logica*, 39:325–333, 1980.

[15] M. Reynolds. The complexity of the temporal logic with until over general linear time. *Journal of Computer and System Science*, to appear.

[16] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, 1985.