Symbolic Model Checking of Tense Logics on Rational Kripke Models

Wilmari Bekker[1,2] and Valentin Goranko[2,⋆]

[1] Department of Mathematics, University of Johannesburg, PO Box 524, Auckland Park, 2006, South Africa
bekkerw@gmail.com
[2] School of Mathematics, University of the Witwatersrand, Private Bag 3, Wits 2050, South Africa
goranko@maths.wits.ac.za

Abstract. We introduce the class of *rational Kripke models* and study symbolic model checking of the basic tense logic $\mathbf{K}_t$ and some extensions of it on that class. Rational Kripke models are based on (generally infinite) *rational graphs*, with vertices labeled by the words in some regular language and transitions recognized by asynchronous two-head finite automata, also known as *rational transducers*. Every atomic proposition in a rational Kripke model is evaluated in a rational set of states. We show that every formula of $\mathbf{K}_t$ has an effectively computable rational extension in every rational Kripke model, and therefore local model checking and global model checking of $\mathbf{K}_t$ in rational Kripke models are decidable. These results are lifted to a number of extensions of $\mathbf{K}_t$. We study and partly determine the complexity of the model checking procedures.

1 Introduction

Verification of models with infinite state spaces using algorithmic symbolic model checking techniques has been an increasingly active area of research over recent years. One very successful approach to infinite state verification is based on the representation of sets of states and transitions by means of automata. It is the basis of various automata-based techniques for model checking, e.g., of linear and branching-time temporal logics on finite transition systems [23,17], regular model checking [7], pushdown systems [8,24,11], automatic structures [14,6] etc.

⋆ This research has been supported by the National Research Foundation of South Africa through a research grant and a student bursary. Part of the work of the second author was done during his visit to the Isaac Newton Institute, Cambridge, as a participant to the 'Logic and Algorithms' programme in 2006. We wish to thank Arnaud Carayol, Balder ten Cate, Carlos Areces, Christophe Morvan, and Stéphane Demri, for various useful comments and suggestions. We are also grateful to the anonymous referee for his/her careful reading of the submitted version and many remarks and corrections which have helped us improve the content and presentation of the paper. (*Received by the editors.* 10 February 2008. *Revised.* 18 September 2008; 15 October 2008. *Accepted.* 17 October 2008.)

M. Archibald et al. (Eds.): ILC 2007, LNAI 5489, pp. 2–20, 2009.

In most of the studied cases of infinite-state symbolic model checking (except for automatic structures), the logical languages are sufficiently expressive for various reachability properties, but the classes of models are relatively restricted.

In this paper we study a large and natural class of *rational Kripke models*, on which global model checking of the basic tense[1] logic $\mathbf{K}_t$ (with forward and backward one-step modalities) and of some extensions thereof, are decidable. The language of $\mathbf{K}_t$ is sufficient for expressing *local properties*, i.e., those referring to a bounded width neighborhood of predecessors or successors of the current state. In particular, pre-conditions and post-conditions are local, but not reachability properties. Kesten et al. [15] have formulated the following minimal requirements for an *assertional language $\mathcal{L}$ to be adequate for symbolic model checking*:

1. The property to be verified and the initial conditions (i.e., the set of initial states) should be expressible in $\mathcal{L}$.
2. $\mathcal{L}$ should be effectively closed under the boolean operations, and should possess an algorithm for deciding equivalence of two assertions.
3. There should exist an algorithm for constructing the predicate transformer pred, where $\mathrm{pred}(\phi)$ is an assertion characterizing the set of states that have a successor state satisfying ϕ.

Assuming that the property to be verified is expressible in $\mathbf{K}_t$, the first condition above is satisfied in our case. Regarding the set of initial states, it is usually assumed a singleton, but certainly an effective set, and it can be represented by a special modal constant S. The second condition is clearly satisfied, assuming the equivalence is with respect to the model on which the verification is being done. As for the third condition, $\mathrm{pred}(\phi) = \langle R \rangle \phi$. Thus, the basic modal logic $\mathbf{K}$ is *the minimal natural logical language satisfying these requirements*, and hence it suffices for specification of *pre-conditions* over regular sets of states. The tense extension $\mathbf{K}_t$ enables specification of post-conditions, as well, thus being the basic adequate logic for specifying *local properties* of transition systems and warranting the potential utility of the work done in the present paper. In particular, potential areas of applications of model checking of the basic tense logic to verification of infinite state systems are *bounded model checking* [2], applied to infinite state systems, and (when extended with reachability) *regular model checking* [7] – a framework for algorithmic verification of generally infinite state systems which essentially involves computing reachability sets in regular Kripke models.

The paper is organized as follows: in Section 2 we introduce $\mathbf{K}_t$ and rational transducers. Section 3 introduces and discusses rational Kripke models, and in Section 4 we introduce synchronized products of transducers and automata. We use them in Section 5 to show decidability of global and local symbolic model checking of $\mathbf{K}_t$ in rational Kripke models and in Section 6 we discuss its complexity. The model checking results are strengthened in Section 7 to hybrid and other extensions of $\mathbf{H}_t(U)$, for which some model checking tasks remain decidable.

[1] We use the term 'tense' rather than 'temporal' to emphasize that the accessibility relation is not assumed transitive, as in a usual flow of time.

2 Preliminaries

2.1 The Basic Tense Logic $\mathbf{K}_t$

We consider transition systems with one transition relation R. The *basic tense logic* $\mathbf{K}_t$ for such transition systems extends the classical propositional logic with two unary modalities: one associated with R and the other with its inverse R^{-1}, respectively denoted by $[R]$ and $[R^{-1}]$. The generalization of what follows to the case of languages and models for transition systems with many relations is straightforward. Note that the relation R is not assumed transitive, and therefore the language of $\mathbf{K}_t$ cannot express R-reachability properties.

2.2 Rational Transducers and Rational Relations

Rational transducers, studied by Eilenberg [9], Elgot and Mezei [10], Nivat, Berstel [1], etc., are *asynchronous automata* on pairs of words. Intuitively, these are finite automata with two autonomous heads that read the input pair of words asynchronously, i.e. each of them can read arbitrarily farther ahead of the other. The transitions are determined by a finite set of pairs of (possibly empty) words; alternatively, a transition can be labeled either by a pair of letters (when both heads make a move on their respective words) or by $\langle a, \epsilon\rangle$ or $\langle \epsilon, a\rangle$, where a is a letter, and ϵ is the empty word (when one of the heads reads on, while the other is waiting). The formal definition follows.

Definition 1. *A **(rational) transducer** is a tuple $\mathcal{T} = \langle Q, \Sigma, \Gamma, q_i, F, \rho\rangle$ where Σ and Γ are the input and output alphabets respectively, Q a set of states, $q_i \in Q$ a unique starting state, $F \subseteq Q$ a set of accepting states and $\rho \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$ is the transition relation, consisting of* finitely many *tuples, each containing the current state, the pair of letters (or ε) triggering the transition, and the new state. Alternatively, one can take $\rho \subseteq Q \times \Sigma^* \times \Gamma^* \times Q$.*

The language recognized by the transducer $\mathcal{T}$ is the set of all pairs of words for which it has a reading that ends in an accepting state. Thus, the transducer $\mathcal{T}$ recognizes a binary relation $R \subseteq \Sigma^* \times \Gamma^*$.

This is the 'static' definition of rational transducers; they can also be defined 'dynamically', as reading an input word, and transforming it into an output word, according to the transition relation which is now regarded as a mapping from words to sets of words (because it can be non-deterministic).

Example 1. For $\mathcal{T} = \langle Q, \Sigma, \Gamma, q_i, F, \rho\rangle$ let: $Q = \{q_1, q_2\}$; $\Sigma = \{0, 1\} = \Gamma$; $q_i = q_1$; $F = \{q_2\}$; $\rho = \{(q_1, 0, 0, q_1), (q_1, 1, 1, q_1), (q_1, \epsilon, 0, q_2), (q_1, \epsilon, 1, q_2)\}$

Notice that in the representation of $\mathcal{T}$ there is only one edge between two states but that an edge may have more than one label.

A relation $R \subseteq \Sigma^* \times \Gamma^*$ is **rational** if it is recognizable by a rational transducer. Equivalently (see [1]), given finite alphabets Σ, Γ, a (binary) **rational relation** over (Σ, Γ) is a rational subset of $\Sigma^* \times \Gamma^*$, i.e., a subset generated by a rational

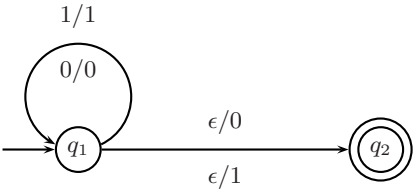

Fig. 1. The transducer $\mathcal{T}$ which recognizes pairs of words of the forms $(u, u0)$ or $(u, u1)$ where $u \in \Sigma^*$

expression (built up using union, concatenation, and iteration) over a finite subset of $\Sigma^* \times \Gamma^*$. Hereafter, we will assume that the input and output alphabets Σ and Γ coincide.

Besides the references above, rational relations have also been studied by Johnson [13], Frougny and Sakarovich [12], and more recently by Morvan [20]. It is important to note that the class of rational relations is closed under *unions*, *compositions*, and *inverses* [1]. On the other hand, the class of rational relations is not closed under intersections, complements, and transitive closure (*ibid*).

3 Rational Kripke Models

3.1 Rational Graphs

Definition 2. *A graph $\mathcal{G} = (S, E)$ is **rational**, if the set of vertices S is a regular language in some finite alphabet Σ and the set of edges E is a rational relation on Σ.*

Example 2. **The infinite grid.** Let $\Sigma = \{0, 1\}$, then the infinite grid with vertices in Σ^* is given by Figure 2 and the edge relation of this graph is recognized by the transducer given in Figure 2.

Example 3. The **complete binary tree** Λ.
Figure 3 contains the complete binary tree with vertices in $\{0, 1\}^*$ and labeled by $\Gamma = \{a, b\}$, as well as the transducer recognizing it, in which the accepting states are labeled respectively by a and b. The pairs of words for which the transducer ends in the accepting state q_4 belong to the left successor relation in the tree (labeled by a), and those for which the transducer ends in the accepting state q_5 belong to the right successor relation in the tree (labeled by b).

An important and extensively studied subclass of rational graphs is the class of *automatic graphs* [14,6]. These are rational graphs whose transition relations are recognized by *synchronized* transducers.

As shown by Blumensath [5], the configuration graph of every Turing machine is an automatic graph. Consequently, important queries, such as reachability, are generally undecidable on automatic graphs, and hence on rational graphs. Furthermore, Morvan showed in [20] that the configuration graphs of Petri nets [21] are rational (in fact, automatic) graphs, too.

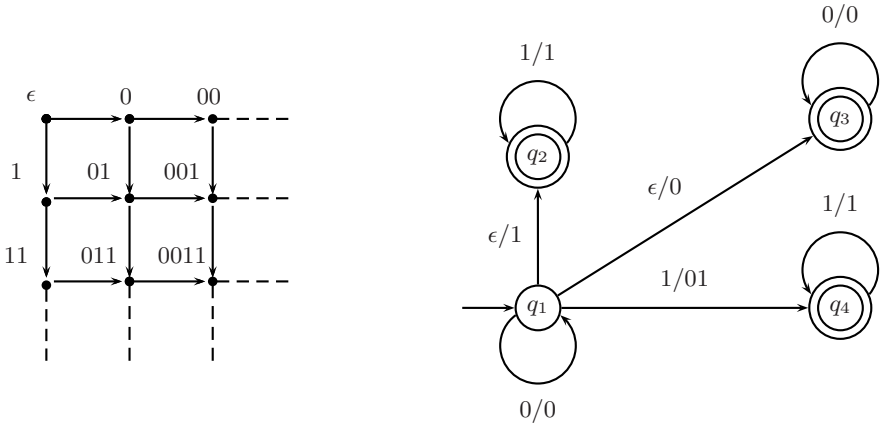

Fig. 2. The infinite grid with set of vertices $S = 0^*1^*$ and a transducer that recognizes the infinite grid

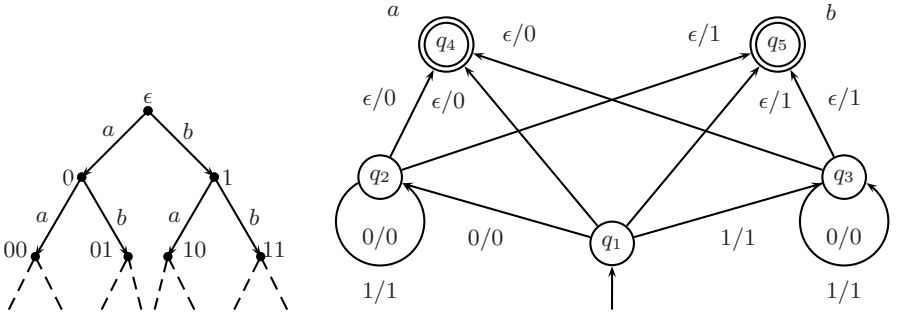

Fig. 3. The complete binary tree Λ and a labeled transducer recognizing it

Moreover, Johnson [13] proved that even very simple first-order definable properties of a rational relation, e.g., reflexivity, transitivity, symmetry, turn out to be undecidable (with an input the transducer recognizing the relation), by reduction from the Post Correspondence Problem (PCP). Independently, Morvan [20] has shown that the query $\exists x Rxx$ on rational frames is undecidable, as well. The reduction of PCP here is straightforward: given a PCP $\{(u_1, v_1), \ldots, (u_n, v_n)\}$, consider a transducer with only one state, which is both initial and accepting, and it allows the transitions $(u_1, v_1), \ldots, (u_n, v_n)$. Then, the PCP has a solution precisely if some pair (w, w) is accepted by the transducer. Inclusion and equality of rational relations are undecidable, too, [1].

Furthermore, in [22] W. Thomas has constructed a single rational graph with undecidable first-order theory, by encoding the halting problem of a universal Turing machine.

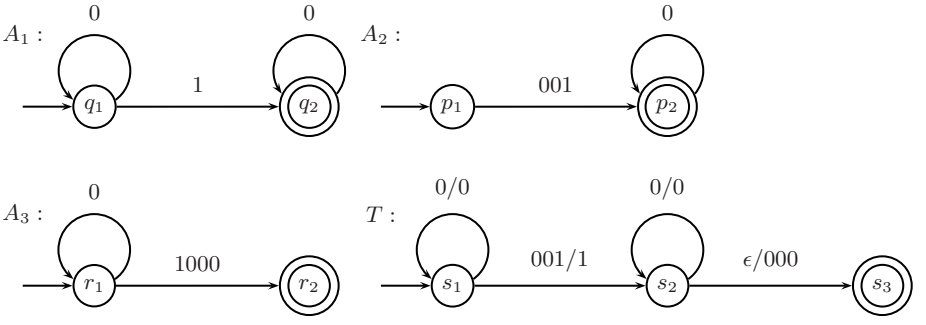

Fig. 4. A finite presentation $\mathcal{M}$: A_1, A_2 and A_3 recognize $S, V(p)$ and $V(q)$ respectively, and T recognizes R

3.2 Rational Kripke Models

Rational graphs can be viewed as Kripke frames, hereafter called *rational Kripke frames.*

Definition 3. *A Kripke model $\mathcal{M} = (\mathcal{F}, V) = (S, R, V)$ is a **rational Kripke model** (RKM) if the frame $\mathcal{F}$ is a rational Kripke frame, and the valuation V assigns a regular language to each propositional variable, i.e., $V(p) \in \mathsf{REG}(\Sigma^*)$ for every $p \in \Phi$. A valuation satisfying this condition is called a rational valuation.*

Example 4. In this example we will present a RKM based on the configuration graph of a Petri net. To make it self-contained, we give the basic relevant definitions here; for more details, cf., e.g., [21]. A Petri net is a tuple (P, T, F, M) where P and T are disjoint finite sets and their elements are called *places* and *transitions* respectively. $F : (P \times T) \cup (T \times P) \to \mathbb{N}$ is called a flow function and is such that if $F(x, y) > 0$ then there is an arc from x to y and $F(x, y)$ is the multiplicity of that arc. Each of the places contain a number of *tokens* and a vector of integers $M \in \mathbb{N}^{|P|}$ is called a *configuration* (or, *marking*) of the Petri net if the i^{th} component of M is equal to the number of tokens at the i^{th} place in the Petri net. The *configuration graph* of $\mathcal{N}$ has as vertices all possible configurations of $\mathcal{N}$ and the edges represent the possible transitions between configurations.

Now, let $\mathcal{N} = (P, T, F, M)$ be a Petri net, where $P = \{p_1, p_2\}$, $T = \{t\}$, $F(p_1, t) = 2$, $F(t, p_2) = 3$ and $M = (4, 5)$. Let $\mathcal{M} = (S, R, V)$ where $S = 0^*10^*$, R the transition relation of the configuration graph of $\mathcal{N}$ and V the valuation defined by $V(p) = 0010^*$ and $V(q) = 0^*1000$. Then $\mathcal{M}$ is a RKM and can be presented by the various machines in Figure 4.

4 Synchronized Products of Transducers and Automata

In this section ϵ will denote the empty word, but will also be treated as a special symbol in an extended alphabet.

Definition 4. *Let u be a word in some alphabet Γ and $\gamma \in \Gamma$. The* ***γ-reduction*** *of u, denoted $u|_\gamma$, is the word obtained from u after deleting all occurrences of γ. Likewise, if Y is a language in the alphabet Γ, then the* ***γ-reduction*** *of Y, denoted $Y|_\gamma$, is the language consisting of all γ-reductions of words in Y.*

Lemma 1. *If Y is a regular language over an alphabet Γ then $Y|_\gamma$ is a regular language over the alphabet $\Gamma - \{\gamma\}$.*

Proof. (Sketch) An automaton $\mathcal{A}|_\gamma$ recognizing $Y|_\gamma$, called here the **γ-reduction** of $\mathcal{A}$ can be constructed from an automaton $\mathcal{A}$ recognizing Y as follows:

1. Remove all γ-transitions.
2. Add (q, γ', q'') as a transition in $\mathcal{A}|_\gamma$ whenever (q, γ, q') and (q', γ', q'') are transitions in $\mathcal{A}$ and $\gamma \neq \gamma'$.
3. Finally, define the accepting states of $\mathcal{A}|_\gamma$ as all accepting states of $\mathcal{A}$ plus those states q such that $(q \overset{\gamma^*}{\rightarrow} q')$ in $\mathcal{A}$ and q' is an accepting state in $\mathcal{A}$. ■

Definition 5. *A* ***run*** *of a finite automaton $\mathcal{A} = \langle Q, \Sigma, q^0, F, \delta \rangle$ is a sequence of states and transitions of $\mathcal{A}$: $q_0 \overset{x_1}{\rightarrow} q_1 \overset{x_2}{\rightarrow} q_2 \cdots \overset{x_n}{\rightarrow} q_n$, such that $q_0 = q^0$, $q_j \in Q, x_j \in \Sigma$, and $q_j \in \delta(q_{j-1}, x_j)$ for every $j = 1, 2, \ldots, n$.*

A run is ***accepting*** *if it ends in an accepting state.*

Run and accepting runs of transducers are defined likewise.

Definition 6. *A* ***stuttering run*** *of a finite automaton $\mathcal{A} = \langle Q, \Sigma, q^0, F, \delta \rangle$ is a sequence $q_0 \overset{x_1}{\rightarrow} q_1 \overset{x_2}{\rightarrow} q_2 \cdots \overset{x_n}{\rightarrow} q_n$, such that $q_0 = q^0$, $q_j \in Q$, and either $x_j \in \Sigma$ and $q_j \in \delta(q_{j-1}, x_j)$, or $x_j = \epsilon$ and $q_j = q_{j-1}$ for every $j = 1, 2, \ldots, n$.*

Thus, a stuttering run of an automaton can be obtained by inserting ϵ-transitions from a state to itself into a run of that automaton. If the latter run is accepting, we declare the stuttering run to be an ***accepting stuttering run****.*

A ***stuttering word*** *in an alphabet Σ is any word in $\Sigma \cup \{\epsilon\}$.*

The ***stuttering language*** *of the automaton $\mathcal{A}$ is the set $L^\epsilon(\mathcal{A})$ of all stuttering words whose ϵ-reductions are recognized by $\mathcal{A}$; equivalently, all stuttering words for which there is an accepting stuttering run of the automaton.*

Definition 7. *Let $\mathcal{T} = \langle Q_\mathcal{T}, \Sigma, q^0_\mathcal{T}, F_\mathcal{T}, \rho_\mathcal{T} \rangle$ be a transducer, and let $\mathcal{A}$ be a (non-deterministic) finite automaton given by $\mathcal{A} = \langle Q_\mathcal{A}, \Sigma, q^0_\mathcal{A}, F_\mathcal{A}, \delta_\mathcal{A} \rangle$.*

The ***synchronized product*** *of $\mathcal{T}$ with $\mathcal{A}$ is the finite automaton:*

$$\mathcal{T} \prec \mathcal{A} = \langle Q_\mathcal{T} \times Q_\mathcal{A}, \Sigma, (q^0_\mathcal{T}, q^0_\mathcal{A}), F_\mathcal{T} \times F_\mathcal{A}, \delta_{\mathcal{T} \prec \mathcal{A}} \rangle$$

where $\delta_{\mathcal{T} \prec \mathcal{A}} : (Q_\mathcal{T} \times Q_\mathcal{A}) \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}(Q_\mathcal{T} \times Q_\mathcal{A})$ is such that, for any $p^1_\mathcal{T}, p^2_\mathcal{T} \in Q_\mathcal{T}$ and $p^1_\mathcal{A}, p^2_\mathcal{A} \in Q_\mathcal{T}$ then $(p^2_\mathcal{T}, p^2_\mathcal{A}) \in \delta_{\mathcal{T} \prec \mathcal{A}}((p^1_\mathcal{T}, p^1_\mathcal{A}), x)$ if and only if

1. *either there exists a $y \in \Sigma$ such that $\delta_\mathcal{A}(p^1_\mathcal{A}, y) = p^2_\mathcal{A}$ and $(p^1_\mathcal{T}, x, y, p^2_\mathcal{T}) \in \rho_\mathcal{T}$,*
2. *or $(p^1_\mathcal{T}, x, \epsilon, p^2_\mathcal{T}) \in \rho_\mathcal{T}$ and $p^1_\mathcal{A} = p^2_\mathcal{A}$.*

Note that every run $R_{\mathcal{T}\prec\mathcal{A}} = (p^0_{\mathcal{T}}, p^0_{\mathcal{A}}) \xrightarrow{u_1} (p^1_{\mathcal{T}}, p^1_{\mathcal{A}}) \xrightarrow{u_2} \cdots \xrightarrow{u_n} (p^n_{\mathcal{T}}, p^n_{\mathcal{T}})$ of the automaton $\mathcal{T} \prec \mathcal{A}$ can be obtained from a pair:

a run $R_{\mathcal{T}} = p^0_{\mathcal{T}} \xrightarrow{(u_1/w_1)} p^1_{\mathcal{T}} \xrightarrow{(u_2/w_2)} p^2_{\mathcal{T}} \cdots \xrightarrow{(u_n/w_n)} p^n_{\mathcal{T}}$ in $\mathcal{T}$,
and a stuttering run $R^s_{\mathcal{A}} = p^0_{\mathcal{A}} \xrightarrow{w_1} p^1_{\mathcal{A}} \xrightarrow{w_2} p^2_{\mathcal{A}} \cdots \xrightarrow{w_n} p^n_{\mathcal{A}}$ in $\mathcal{A}$,
by pairing the respective states $p^j_{\mathcal{T}}$ and $p^j_{\mathcal{A}}$ and removing the output symbol w_j for every $j = 1, 2, \ldots, n$.

Let the reduction of $R^s_{\mathcal{A}}$ be the run $R_{\mathcal{A}} = q^0_{\mathcal{A}} \xrightarrow{v_1} q^1_{\mathcal{A}} \xrightarrow{v_2} q^2_{\mathcal{A}} \cdots \xrightarrow{v_m} q^m_{\mathcal{A}}$, with $m \leq n$. Then we say that the run $R_{\mathcal{T}\prec\mathcal{A}}$ is a **synchronization of the runs** $R_{\mathcal{T}}$ and $R_{\mathcal{A}}$.

Note, that the synchronization of accepting runs of $\mathcal{T}$ and $\mathcal{A}$ is an accepting run of $R_{\mathcal{T}\prec\mathcal{A}}$. The following lemma is now immediate:

Lemma 2. *Let $\mathcal{T} = \langle Q_{\mathcal{T}}, \Sigma, q^0_{\mathcal{T}}, F_{\mathcal{T}}, \rho_{\mathcal{T}} \rangle$ be a transducer recognizing the relation $R(\mathcal{T})$ and let $\mathcal{A} = \langle Q_{\mathcal{A}}, \Sigma, q^0_{\mathcal{A}}, F_{\mathcal{A}}, \delta_{\mathcal{A}} \rangle$ be a finite automaton recognizing the language $L(\mathcal{A})$. Then the language recognized by the synchronized product of $\mathcal{T}$ and $\mathcal{A}$ is*

$$L(\mathcal{T} \prec \mathcal{A}) = \{u \mid \exists w \in L^{\epsilon}(\mathcal{A})(uR(\mathcal{T})w).\}$$

5 Model Checking of K_t in Rational Kripke Models

In this section we will establish decidability of the basic model checking problems for formulae of K_t in rational Kripke models.

Lemma 3. *Let Σ be a finite non-empty alphabet, $X \subseteq \Sigma^*$ a regular subset, and let $R \subseteq \Sigma^* \times \Sigma^*$ be a rational relation. Then the sets*

$$\langle R \rangle X = \{u \in \Sigma^* | \exists v \in X(uRv)\}$$

and

$$\left\langle R^{-1} \right\rangle X = \{u \in \Sigma^* | \exists v \in X(vRu)\}$$

are regular subsets of Σ^*.

Proof. This claim essentially follows from results of Nivat (see [1]). However, using Lemmas 1 and 2, we give a constructive proof, which explicitly produces automata that recognize the resulting regular languages. Let $\mathcal{A}$ be a finite automaton recognizing X and $\mathcal{T}$ be a transducer recognizing R. Then, the ϵ-reduction of the synchronized product of $\mathcal{T}$ with $\mathcal{A}$ is an automaton recognizing $\langle R \rangle X$; for $\left\langle R^{-1} \right\rangle X$ we take instead of $\mathcal{T}$ the transducer for R^{-1} obtained from $\mathcal{T}$ by swapping the input and output symbols in the transition relation[2]. ■

Example 5. Consider the automaton $\mathcal{A}$ and transducer $\mathcal{T}$ in Figure 5. The language recognized by $\mathcal{A}$ is $X = 1^*(1 + 0^+)$ and the relation R recognized by $\mathcal{T}$ is $R = \left\{(1^n 0, 10^n 1)^m \left(1^k, 10^k\right) \mid n, m, k \in \mathbb{N}\right\} \cup$

[2] Note that, in general, the resulting automata need not be minimal, because they may have redundant states and transitions.

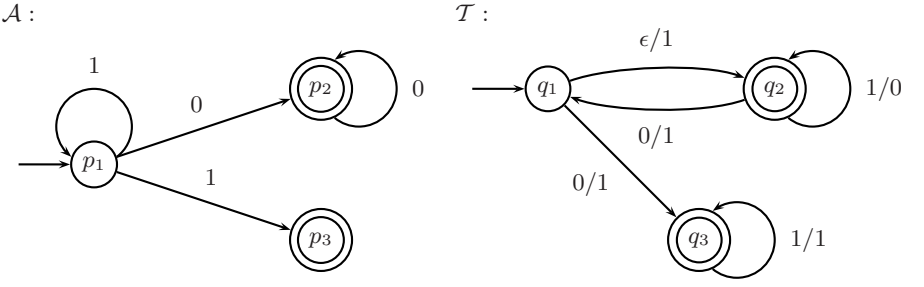

Fig. 5. The automaton $\mathcal{A}$ and the transducer $\mathcal{T}$

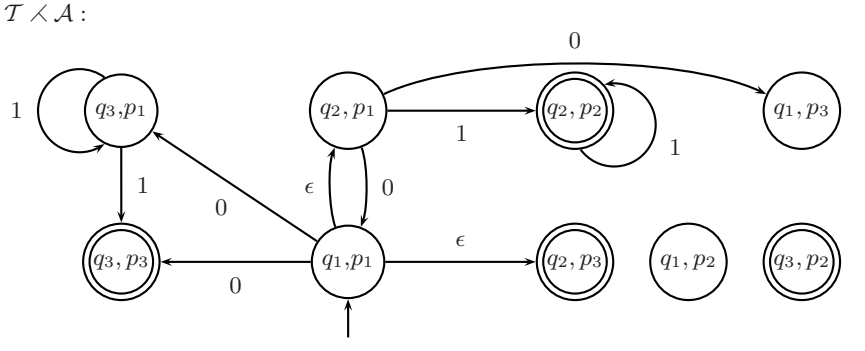

Fig. 6. The synchronized product $\mathcal{T} \asymp \mathcal{A}$ recognizing $\langle R \rangle X$

$\left\{(1^n 0, 10^n 1)^m \left(01^k, 11^k\right) \mid n, m, k \in \mathbb{N}\right\}$, where $X_1 X_2$ denotes the componentwise concatenation of the relations X_1 and X_2, i.e., $X_1 X_2 = \{(u_1 u_2, v_1 v_2) \mid (u_1, v_1) \in X_1, (u_2, v_2) \in X_2\}$. For instance, if we take $n = 1$, $m = 2$ and $k = 3$ we obtain that $(10, 101)^2(1^3, 10^3) = (1010111, 1011011000) \in R$ (coming from the first set of the union) and $(10, 101)^2(01^3, 11^3) = (10100111, 1011011111) \in R$ (coming from the second set of that union).

Then, the synchronized product $\mathcal{T} \asymp \mathcal{A}$ is the finite automaton given in Figure 6 recognizing $\langle R \rangle X = 0^* + 0^* 1^+$. Note that it can be simplified by removing redundant states and edges.

Theorem 1. *For every formula $\varphi \in \mathbf{K}_t$ and rational Kripke model $\mathcal{M} = (\Sigma^*, R, V)$, the set $[\![\varphi]\!]_{\mathcal{M}}$ is a rational language, effectively computable from φ and the rational presentation of $\mathcal{M}$.*

Proof. We prove the claim by induction on φ.

1. If φ is an atomic proposition, the claim follows from the definition of a rational model.
2. The boolean cases follow from the effective closure of regular languages under boolean operations.

3. If $\varphi = \langle R \rangle \psi$ then $[\![\varphi]\!]_{\mathcal{M}} = \langle R \rangle [\![\psi]\!]_{\mathcal{M}}$, which is regular by the inductive hypothesis and Lemma 3. Likewise for the case $\varphi = \langle R^{-1} \rangle \psi$. ■

We now consider the following algorithmic model checking problems, where the Kripke model is supposed to be given by some effective presentation:

1. *Local model checking:* given a Kripke model $\mathcal{M}$, a state s in $\mathcal{M}$, and a formula φ of $\mathbf{K}_t$, determine whether $\mathcal{M}, s \models \varphi$.
2. *Global model checking:* given a Kripke model $\mathcal{M}$ and a formula φ of $\mathbf{K}_t$, determine (effectively) the set $[\![\varphi]\!]_{\mathcal{M}}$ of all states in $\mathcal{M}$ where φ is true.
3. *Checking satisfiability in a model:* given a Kripke model $\mathcal{M}$ and a formula φ of $\mathbf{K}_t$, determine whether $[\![\varphi]\!]_{\mathcal{M}} \neq \varnothing$.

Corollary 1. *Local model checking, global model checking, and checking satisfiability in a model, of formulae in* $\mathbf{K}_t$ *in rational Kripke models are decidable.*

Proof. Decidability of the global model checking follows immediately from Theorem 1. Then, decidability of the local model checking and of checking satisfiability in a rational model follow respectively from the decidability of membership in a regular language, and of non-emptiness of a regular language (cf., e.g., [18]). ■

6 Complexity

We will now attempt to analyze the complexity of global model checking a formula in $\mathbf{K}_t$ on a rational Kripke model. Depending on which of these is fixed, we distinguish two complexity measures (cf., e.g., [16]): **formula (expression) complexity** (when the model is fixed and the formula is feeded as input) and **structure complexity** (when the formula is fixed and the model is feeded as input).

6.1 Normal Forms and Ranks of Formulae

We will first need to define some standard technical notions.

A formula $\varphi \in \mathbf{K}_t$ is in **negation normal form** if every occurrence of the negation immediately precedes a propositional variable. Clearly every formula $\varphi \in \mathbf{K}_t$ is equivalent to a formula $\psi \in \mathbf{K}_t$ in negation normal form, of size linear in the size φ. For the remainder of this section, we will assume that a formula φ we wish to model check is in a negation normal form.

The modal rank of a formula counts the greatest number of nested modalities in the formula, while the alternating box (resp., diamond) rank of a formula counts the greatest number of nested alternations of modalities with an outmost box (resp., diamond) in that formula. Formally:

Definition 8. *The* **modal rank** *for a formula* $\varphi \in \mathbf{K}_t$*, denoted by* $\mathrm{mr}(\varphi)$ *is defined inductively as follows:*

1. *if* p *is an atomic proposition, then* $\mathrm{mr}(p) = 0$ *and* $\mathrm{mr}(\neg p) = 0$*;*

2. $\mathrm{mr}(\phi_1 \vee \psi_2) = \mathrm{mr}(\phi_1 \wedge \psi_2) = \max\{\mathrm{mr}(\psi_1), \mathrm{mr}(\psi_2)\}$;
3. $\mathrm{mr}(\triangle\, \psi) = \mathrm{mr}(\psi) + 1$ *where* $\triangle \in \{[R], \langle R\rangle, [R^{-1}], \langle R^{-1}\rangle\}$.

Definition 9. *The* **alternating box rank** *and* **alternating diamond rank** *of a formula* $\varphi \in \mathbf{K}_t$*, denoted respectively by* $\mathrm{ar}_\Box(\varphi)$ *and* $\mathrm{ar}_\Diamond(\varphi)$*, are defined by simultaneous induction as follows, where* $\triangle \in \{\Box, \Diamond\}$*:*

1. *if* p *is an atomic proposition, then* $\mathrm{ar}_\triangle(p) = 0$ *and* $\mathrm{ar}_\triangle(\neg p) = 0$;
2. $\mathrm{ar}_\triangle(\psi_1 \vee \psi_2) = \mathrm{ar}_\triangle(\psi_1 \wedge \psi_2) = \max\{\mathrm{ar}_\triangle(\psi_1), \mathrm{ar}_\triangle(\psi_2)\}$;
3. $\mathrm{ar}_\Diamond(\langle R\rangle \psi) = \mathrm{ar}_\Box(\psi) + 1$ *and* $\mathrm{ar}_\Box(\langle R\rangle \psi) = \mathrm{ar}_\Box(\psi)$.
   Likewise for $\mathrm{ar}_\Diamond(\langle R^{-1}\rangle \psi)$ *and* $\mathrm{ar}_\Box(\langle R^{-1}\rangle \psi)$.
4. $\mathrm{ar}_\Box([R]\psi) = \mathrm{ar}_\Diamond(\psi) + 1$ *and* $\mathrm{ar}_\Diamond([R]\psi) = \mathrm{ar}_\Diamond(\psi)$.
   Likewise for $\mathrm{ar}_\Diamond([R^{-1}]\psi)$ *and* $\mathrm{ar}_\Box([R^{-1}]\psi)$.

Finally, the **alternation rank** *of* φ*, denoted* $ar(\varphi)$ *is defined to be*

$$ar(\varphi) = \max\{\mathrm{ar}_\Box(\varphi), \mathrm{ar}_\Diamond(\varphi)\}.$$

For instance, $\mathrm{ar}_\Box([R](\langle R\rangle [R]p \vee [R][R^{-1}]\neg q)) = 3$ and $\mathrm{ar}_\Diamond([R](\langle R\rangle [R]p \vee [R][R^{-1}]\neg q)) = 2$, hence $ar([R](\langle R\rangle [R]p \vee [R][R^{-1}]\neg q)) = 3$.

6.2 Formula Complexity

We measure the size of a finite automaton or transducer $\mathcal{M}$ by the number of transition edges in it, denoted $|\mathcal{M}|$.

Proposition 1. *If* $\mathcal{A}$ *is an automaton recognizing the regular language* X *and* $\mathcal{T}$ *a transducer recognizing the rational relation* R*, then the time complexity of computing an automaton recognizing* $\langle R\rangle^m X$ *is in* $O(|\mathcal{T}|^m|\mathcal{A}|)$.

Proof. The size of the synchronized product $\mathcal{T} \curlywedge \mathcal{A}$ of $\mathcal{T}$ and $\mathcal{A}$ is bounded above by $|\mathcal{T}||\mathcal{A}|$ and it can be computed in time $O(|\mathcal{T}||\mathcal{A}|)$. The claim now follows by iterating that procedure m times. ■

However, we are going to show that the time complexity of computing an automaton recognizing $[R]X$ is far worse.

For a regular language X recognized by an automaton $\mathcal{A}$, we define $R_X = \{(u, \epsilon) \,|\, u \in X\}$. A transducer $\mathcal{T}$ recognizing R_X can be constructed from $\mathcal{A}$ by simply replacing every edge (q, x, p) in $\mathcal{A}$ with the edge (q, x, ϵ, p).

Lemma 4. *Let* X *be a regular language. Then the complementation* $\overline{X}$ *of* X *equals* $[R_X]\varnothing$.

Proof. Routine verification. ■

Consequently, computing $[R_X]\varnothing$ cannot be done in less than exponential time in the size of the (non-deterministic) automaton $\mathcal{A}$ for X. This result suggests the following conjecture.

Conjecture 1. The formula complexity of global model checking of a $\mathbf{K}_t$-formula is non-elementary in terms of the alternating box rank of the formula.

6.3 Structure Complexity

Next we analyze the **structure complexity**, i.e. the complexity of global model checking a fixed formula $\varphi \in \mathbf{K}_t$ on an input rational Kripke model. Here the input is assumed to be the transducer and automata presenting the model.

Fix a formula $\varphi \in \mathbf{K}_t$ in negation normal form, then for any input rational Kripke model $\mathcal{M}$ there is a fixed number of operations to perform on the input transducer and automata that can lead to subsequent exponential blowups of the size of the automaton computing $[\![\varphi]\!]_{\mathcal{M}}$. That number is bounded by the modal rank $\mathrm{mr}(\varphi)$ of the formula φ, and therefore the structure complexity is bounded above by an exponential tower of a height not exceeding that modal rank:

$$2^{\cdots^{(\mathrm{mr}(\varphi)\ \text{ times})^{\cdots}2^{|\mathcal{T}||\mathcal{A}|}}}$$

However, using the alternation rank of φ and Proposition 1 we can do better.

Proposition 2. *The structure complexity of global model checking for a fixed formula $\varphi \in \mathbf{K}_t$ on an input rational Kripke model $\mathcal{M}$, presented by the transducer and automata $\{\mathcal{T}, \mathcal{A}_1, \ldots, \mathcal{A}_n\}$, is bounded above by*

$$2^{\cdots^{(\mathrm{ar}(\varphi)\ \text{ times})^{\cdots}2^{P(|\mathcal{T}|)}}}$$

where $P(|\mathcal{T}|)$ is a polynomial in $|\mathcal{T}|$ with leading coefficient not greater that $n2^c$ where $c \leq \max\{|\mathcal{A}_i| \mid i = 1, \ldots n\}$ and degree no greater than $\mathrm{mr}(\varphi)$.

Proof. The number of steps in the computation of $[\![\varphi]\!]_{\mathcal{M}}$, following the structure of φ, that produce nested exponential blow-ups can be bounded by the alternation rank, since nesting of any number of diamonds does not cause an exponential blow-up, while nesting of any number of boxes can be reduced by double complementation to nesting of diamonds; e.g., $[R]([R][R]p \vee [R^{-1}]\neg q)$ can be equivalently re-written as $\neg\langle R\rangle(\langle R\rangle\langle R\rangle\neg p \wedge \langle R^{-1}\rangle q)$. The initial synchronized product construction (when a diamond or box is applied to a boolean formula) produces an automaton of size at most $2^c|\mathcal{T}|$, the number of nested product constructions is bounded above by $\mathrm{mr}(\varphi)$, and each of these multiplies the size of the current automaton by $|\mathcal{T}|$. In the worst case, all alternations would take place after all product constructions, hence the upper bound. ■

7 Model Checking Extensions of $\mathbf{K}_t$ on Rational Models

7.1 Model Checking Hybrid Extensions of $\mathbf{K}_t$

A major limitation of the basic modal language is its inability to refer explicitly to states in a Kripke model, although the modal semantics evaluates modal formulae at states. Hybrid logics provide a remedy for that problem. We will

only introduce some basic hybrid logics of interest here; for more details consult, e.g., [3,4].

The *basic hybrid tense logic* $\mathbf{H}_t$ extends the basic tense logic $\mathbf{K}_t$ with a set of new atomic symbols Θ called *nominals* which syntactically form a second type of atomic formulae, which are evaluated in Kripke models in *singleton sets* of states. The unique state in the valuation of a nominal is called its *denotation*. Thus, nominals can be used in $\mathbf{H}_t$ to refer directly to states.

Here is the formal definition of the set of formulae of $\mathbf{H}_t$:

$$\varphi = p \mid i \mid \neg\varphi \mid \varphi \vee \phi \mid \langle R\rangle\, \varphi \mid \left\langle R^{-1}\right\rangle \varphi,$$

where $i \in \Theta$ and $p \in \Phi$.

The basic hybrid logic $\mathbf{H}_t$ can be further extended to $\mathbf{H}_t\,(@)$ by adding the *satisfaction operator* $@$, where the formula $@_i\varphi$ means 'φ *is true at the denotation of* i'. A more expressive extension of $\mathbf{H}_t$ is $\mathbf{H}_t(U)$ involving the *universal modality* with semantics $\mathcal{M}, v \models [U]\varphi$ iff $\mathcal{M}, w \models \varphi$ for every $w \in \mathcal{M}$. The operator $@$ is definable in $\mathbf{H}_t(U)$ by $@_i\varphi := [U](i \rightarrow \varphi)$. Moreover, $\mathbf{H}_t$ can be extended with the more expressive *difference modality* $\langle D\rangle$ (and its dual $[D]$), where $\mathcal{M}, v \models \langle D\rangle\varphi$ iff there exists a $w \neq v$ such that $\mathcal{M}, w \models \varphi$. Note that $[U]$ is definable in $\mathbf{H}_t(D)$ by $[U]\varphi := \varphi \wedge [D]\varphi$.

Yet another extension of $\mathbf{H}_t\,(@)$ is $\mathbf{H}_t\,(@, \downarrow)$ which also involves *state variables* and *binders* that bind these variables to states. Thus, in addition to $\mathbf{H}_t\,(@)$, formulae also include $\downarrow x.\varphi$ for x a state variable. For a formula φ possibly containing free occurrences of a state variable x, and w a state in a given model, let $\varphi\,[x \leftarrow i_w]$ denote the result of substitution of all free occurrences of x by a nominal i_w in φ, where w is the denotation of i_w. Then the semantics of $\downarrow x.\varphi$ is defined by: $\mathcal{M}, w \models \downarrow x.\varphi$ iff $\mathcal{M}, w \models \varphi\,[x \leftarrow i_w]$.

Proposition 3. *For every formula* φ *of the hybrid language* $\mathbf{H}_t(D)$ *(and therefore, of* $\mathbf{H}_t\,(@)$ *and of* $\mathbf{H}_t\,(U)$*) and every rational Kripke model* $\mathcal{M}$*, the set* $[\![\varphi]\!]_{\mathcal{M}}$ *is an effectively computable rational language.*

Proof. The claim follows from Theorem 1 since the valuations of nominals, being singletons, are rational sets, and the difference relation D is a rational relation. The latter can be shown by explicitly constructing a transducer recognizing D in a given rational set, or by noting that it is the complement of the automatic relation of equality, hence it is automatic itself, as the family of automatic relations is closed under complements (cf., e.g., [14] or [6]). ■

Corollary 2. *Global and local model checking, as well as satisfiability checking, of formulae of the hybrid language* $\mathbf{H}_t(D)$ *(and therefore, of* $\mathbf{H}_t\,(@)$ *and* $\mathbf{H}_t\,(U)$*, too) in rational Kripke models are decidable.*

Proposition 4. *Model checking of the* $\mathbf{H}_t\,(@, \downarrow)$*-formula* $\downarrow x.\,\langle R\rangle\, x$ *in* $\mathbf{H}_t\,(@, \downarrow)$ *on a given input rational Kripke model is not decidable.*

Proof. Immediate consequence from Morvan's earlier mentioned reduction [20] of the model checking of $\exists x Rxx$ to the Post Correspondence Problem. ■

Proposition 5. *There is a rational Kripke model on which model checking formulae from the hybrid language is undecidable.*

Proof. (Sketch) The rational graph constructed by Thomas [22] can be used to prove this undecidability, since the first-order properties queried there are also expressible in $\mathbf{H}_t(@, \downarrow)$. ■

7.2 Counting Modalities

We now consider extensions of $\mathbf{K}_t$ with counting (or, graded) modalities:

- $\Diamond^{\geq k}\varphi$ with semantics: 'there exist at least k successors where φ holds';
- $\Diamond^{\leq k}\varphi$ with semantics: 'there exist at most k successors where φ holds';
- $\Diamond^{k}\varphi$ with semantics: 'there exist exactly k successors where φ holds';
- $\Diamond^{\infty}\varphi$ with semantics: 'there exist infinitely many successors where φ holds'.

Clearly, some of these are inter-definable: $\Diamond^{k}\varphi := \Diamond^{\geq k}\varphi \wedge \Diamond^{\leq k}\varphi$, while $\Diamond^{\geq k}\varphi := \neg\Diamond^{\leq k-1}\varphi$ and $\Diamond^{\leq k}\varphi := \neg\Diamond^{\geq k+1}\varphi$.

We denote by $\mathbf{C}_t$ the extension of $\mathbf{K}_t$ with $\Diamond^{\infty}\varphi$ and all counting modalities for all integers $k \geq 0$. Further, we denote by $\mathbf{C}_t^0$ the fragment of $\mathbf{C}_t$ where no occurrence of a counting modality is in the scope of any modal operator.

Proposition 6. *Local model checking of formulae in the language $\mathbf{C}_t^0$ in rational Kripke models is decidable.*

Proof. First we note that each of the following problems: '*Given an automaton A, does its language contain at most / at least / exactly k / finitely / infinitely many words?*' is decidable. Indeed, the case of finite (respectively infinite) language is well-known (cf., e.g., [18], pp. 186–189). A decision procedure[3] for recognizing if the language of a given automaton $\mathcal{A}$ contains at least k words can be constructed recursively on k. When $k = 1$ that boils down to checking non-emptiness of the language (*ibid*). Suppose we have such a procedure P_k for a given k. Then, a procedure for $k + 1$ can be designed as follows: first, test the language $L(\mathcal{A})$ of the given automaton for non-emptiness by looking for any word recognized by it (by searching for a path from the initial state to any accepting state). If such a word w is found, modify the current automaton to exclude (only) w from its language, i.e. construct an automaton for the language $L(\mathcal{A}) \setminus \{w\}$, using the standard automata constructions. Then, apply the procedure P_k to the resulting automaton.

Testing $L(\mathcal{A})$ for having at most k words is reduced to testing for at least $k + 1$ words; likewise, testing for exactly k words is a combination of these.

Now, the claim follows from Theorem 1. Indeed, given a RKM $\mathcal{M}$ and a formula $\varphi \in \mathbf{C}_t^0$, for every subformula $\Diamond^c\psi$ of φ, where $\Diamond^c$ is any of the counting

[3] The procedure designed here is perhaps not the most efficient one. but, it will not make the complexity of the model checking worse, given the high overall complexity of the latter.

modalities listed above, the subformula ψ is in $\mathbf{K}_t$, and therefore an automaton for the regular language $[\![\psi]\!]_{\mathcal{M}}$ is effectively computable, and hence the question whether $\Diamond^c\psi$ is true at the state where the local model checking is performed can be answered effectively. It remains to note that every formula of $\mathbf{C}_t^0$ is a boolean combination of subformulae $\Diamond^c\psi$ where $\psi \in \mathbf{K}_t$. ■

At present, we do not know whether any of the counting modalities preserves regularity in rational models, and respectively whether global model checking in rational models of either of these languages is decidable.

7.3 A Presentation Based Extension

Here we consider a 'presentation-based' extension of the multi-modal version of $\mathbf{K}_t$, where the new modalities are defined in terms of word operations, so they only have meaning in Kripke models where the states are labeled by words (such as the rational Kripke models) hereafter called *Kripke word-models*.

To begin with, for a given alphabet Σ, with every language $X \subseteq \Sigma^*$ we can uniformly associate the following binary relations in Σ^*:

$X? := \{(u,u)\,|u \in X\}$;
$\overrightarrow{X} := \{(uv,v)\,|u \in X, v \in \Sigma^*\}$.

Proposition 7. *For every regular language $X \subseteq \Sigma^*$ the relations $X?$ and $\overrightarrow{X}$ are rational.*

Proof. For each of these, there is a simple uniform construction that produces from the automaton recognizing X a transducer recognizing the respective relation. For instance, the transducer for $\overrightarrow{X}$ is constructed as composition of the transducers (defined just like the composition of finite automata) for the rational relations $\{(u,\varepsilon) \mid u \in X\}$ and $\{(v,v) \mid v \in \Sigma^*\}$. The former is constructed from the automaton $\mathcal{A}$ for X by converting every a-transition in $\mathcal{A}$, for $a \in \Sigma$, to (a,ε)-transition, and the latter is constructed from an automaton recognizing Σ^* by converting every a-transition, for $a \in \Sigma$, to (a,a)-transition. ■

This suggests a natural extension of (multi-modal) $\mathbf{K}_t$ with an infinite family of new modalities associated with relations as above defined over the extensions of formulae. The result is a richer, PDL-like language which extends the star-free fragment of PDL with test and converse by additional program constructions corresponding to the regularity preserving operations defined above. We call that language '*word-based star-free PDL (with test and converse)*', hereafter denoted **WPDL**.

Formally, **WPDL** has two syntactic categories, viz., *programs* PROG and *formulae* FOR, defined over given alphabet Σ, set of atomic propositions AP, and set of atomic programs (relations) REL, by mutual induction as follows:

Formulae FOR:

$$\varphi ::= p \mid \ell_a \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \langle\alpha\rangle\varphi$$

for $p \in \mathsf{AP}$, $a \in \Sigma$, and $\alpha \in \mathsf{PROG}$, where for each $a \in \Sigma$ we have added a special new atomic proposition ℓ_a, used further to translate extended star-free regular expressions to **WPDL**-formulae.

Programs PROG:

$$\alpha ::= \pi \mid \alpha\prime \mid \alpha_1 \cup \alpha_2 \mid \alpha_1 \circ \alpha_2 \mid \varphi? \mid \overrightarrow{\varphi}$$

where $\pi \in \mathsf{REL}$ and $\varphi \in \mathsf{FOR}$.

We note that **WPDL** is not a purely logical language, as it does not have semantics on abstract models but only on word-models (including rational Kripke models), defined as follows. Let $\mathcal{M} = (S, \{R_\pi\}_{\pi \in \mathsf{REL}}, V)$ be a Kripke word-model over an alphabet Σ, with a set of states $S \subseteq \Sigma^*$, a family of basic relations indexed with REL, and a valuation V of the atomic propositions from AP. Then every formula $\varphi \in \mathsf{FOR}$ is associated with the language $[\![\varphi]\!]_{\mathcal{M}} \subseteq \Sigma^*$, defined as before, where $[\![p]\!]_{\mathcal{M}} := V(p)$ for every $p \in \mathsf{AP}$ and $[\![\ell_a]\!] := \{a\} \cap S$ for every $a \in \Sigma$. Respectively, every program α is associated with a binary relation R_α in Σ^*, defined inductively as follows (where $\circ$ is composition of relations):

- $R_{\alpha\prime} := R_\alpha^{-1}$,
- $R_{\alpha_1 \cup \alpha_2} := R_{\alpha_1} \cup R_{\alpha_2}$,
- $R_{\alpha_1 \circ \alpha_2} := R_{\alpha_1} \circ R_{\alpha_2}$,
- $R_{\varphi?} := [\![\varphi]\!]?$,
- $R_{\overrightarrow{\varphi}} := \overrightarrow{[\![\varphi]\!]}$.

Lemma 5. *For every* **WPDL***-formulae φ, ψ and a Kripke word-model $\mathcal{M}$:*

1. $[\![\langle\varphi?\rangle\psi]\!]_{\mathcal{M}} = [\![\varphi]\!]_{\mathcal{M}} \cap [\![\psi]\!]_{\mathcal{M}}$.
2. $[\![\langle\overrightarrow{\varphi}\rangle\psi]\!]_{\mathcal{M}} = [\![\varphi]\!]_{\mathcal{M}}; [\![\psi]\!]_{\mathcal{M}}$ *(where ; denotes concatenation of languages).*

Proof. Routine verification:

1. $[\![\langle\varphi?\rangle\psi]\!]_{\mathcal{M}} = \{w \in \Sigma^* \mid wR_{\varphi?}v \text{ for some } v \in [\![\psi]\!]_{\mathcal{M}}\}$
   $= \{w \in \Sigma^* \mid w = v \text{ for some } v \in [\![\varphi]\!]_{\mathcal{M}} \text{ and } v \in [\![\psi]\!]_{\mathcal{M}}\} = [\![\varphi]\!]_{\mathcal{M}} \cap [\![\psi]\!]_{\mathcal{M}}$.
2. $[\![\langle\overrightarrow{\varphi}\rangle\psi]\!]_{\mathcal{M}} = \{w \in \Sigma^* \mid wR_{\overrightarrow{\varphi}}v \text{ for some } v \in [\![\psi]\!]_{\mathcal{M}}\}$
   $= \{uv \in \Sigma^* \mid u \in [\![\varphi]\!]_{\mathcal{M}}, v \in [\![\psi]\!]_{\mathcal{M}}\} = [\![\varphi]\!]_{\mathcal{M}}; [\![\psi]\!]_{\mathcal{M}}$. ■

Corollary 3. *For every* **WPDL***-formula φ and a rational Kripke model $\mathcal{M}$, the language $[\![\varphi]\!]_{\mathcal{M}}$ is an effectively computable from φ regular language.*

Corollary 4. *Local and global model checking, as well as satisfiability checking, of* **WPDL***-formulae in rational Kripke models is decidable.*

Extended star-free regular expressions over an alphabet Σ are defined as follows:

$$E := a \mid \neg E \mid E_1 \cup E_2 \mid E_1; E_2,$$

where $a \in \Sigma$. Every such expression E defines a regular language $L(E)$, where $\neg, \cup, ;$ denote respectively complementation, union, and concatenation of languages. The question whether two extended star-free regular expressions define the same language has been proved to have a non-elementary complexity in [19].

Every extended star-free regular expression can be linearly translated to an **WPDL**-formula:

- $\tau(a) := \ell_a$,
- $\tau(\neg E) := \neg\tau(E)$,
- $\tau(E_1 \cup E_2) := \tau(E_1) \vee \tau(E_2)$,
- $\tau(E_1; E_2) := \langle\overrightarrow{\tau(E_1)}\rangle\tau(E_2)$.

Lemma 6. *Given an alphabet Σ, consider the rational Kripke model $\mathcal{M}^\Sigma$ with set of states Σ^*, over empty sets of basic relations and atomic propositions. Then, for every extended star-free regular expression E,*

$$L(E) = [\![\tau(E)]\!]_{\mathcal{M}^\Sigma}.$$

Proof. Straightforward induction on E. The only non-obvious case $E = E_1; E_2$ follows from Lemma 5. ■

Consequently, for any extended star-free regular expressions E_1 and E_2, we have that $L(E_1) = L(E_2)$ iff $[\![\tau(E_1)]\!]_{\mathcal{M}^\Sigma} = [\![\tau(E_2)]\!]_{\mathcal{M}^\Sigma}$ iff $\mathcal{M}^\Sigma \models \tau(E_1) \leftrightarrow \tau(E_2)$. Thus, we obtain the following.

Corollary 5. *Global model checking of* **WPDL***-formulae in rational Kripke models has non-elementary formula-complexity.*

Remark: Since the $\overrightarrow{\varphi}$-free fragment of **WPDL** is expressively equivalent to $\mathbf{K}_t$, a translation of bounded exponential blow-up from the family of extended star-free regular expressions to the latter fragment would prove Conjecture 1.

8 Concluding Remarks

We have introduced the class of rational Kripke models and shown that all formulae of the basic tense logic $\mathbf{K}_t$, and various extensions of it, have effectively computable rational extensions in such models, and therefore global model checking and local model checking of such formulae on rational Kripke models are decidable, albeit probably with non-elementary formula complexity.

Since model checking reachability on such models is generally undecidable, an important direction for further research would be to identify natural large subclasses of rational Kripke models on which model checking of K_t extended with the reachability modality $\langle R\rangle^*$ is decidable. Some such cases, defined in terms of the presentation, are known, e.g., rational models with length-preserving or length-monotone transition relation [20]; the problem of finding structurally defined large classes of rational models with decidable reachability is still essentially open.

Other important questions concern deciding bisimulation equivalence between rational Kripke models, as that would allow us to transfer model checking of any property definable in the modal mu-calculus from one to the other. These questions are studied in a follow-up to the present work.

References

1. Berstel, J.: Transductions and Context-Free Languages. Teubner Studienbücher Informatik. B.G. Teubner, Stuttgart (1979)
2. Biere, A., Cimatti, A., Clarke, E., Strichman, O., Zhu, Y.: Bounded model checking. Advances in Computers 58, 118–149 (2003)
3. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. Cambridge Tracts in Theoretical Computer Science, vol. 53. CUP (2001)
4. Blackburn, P.: Representation, reasoning, and relational structures: a hybrid logic manifesto. Logic Journal of the IGPL 8(3), 339–365 (2000)
5. Blumensath, A.: Automatic structures. Diploma thesis, RWTH Aachen (1999)
6. Blumensath, A., Grädel, E.: Automatic structures. In: Abadi, M. (ed.) Proc. LICS 2000, pp. 51–62 (2000)
7. Bouajjani, A., Jonsson, B., Nilsson, M., Touili, T.: Regular model checking. In: Emerson, E.A., Sistla, A.P. (eds.) CAV 2000. LNCS, vol. 1855, pp. 403–418. Springer, Heidelberg (2000)
8. Bouajjani, A., Esparza, J., Maler, O.: Reachability analysis of pushdown automata: Application to model-checking. In: Mazurkiewicz, A., Winkowski, J. (eds.) CONCUR 1997. LNCS, vol. 1243, pp. 135–150. Springer, Heidelberg (1997)
9. Eilenberg, S.: Automata, Languages and Machines, vol. A. Academic Press, New York (1974)
10. Elgot, C., Mezei, J.: On relations defined by finite automata. IBM J. of Research and Development 9, 47–68 (1965)
11. Esparza, J., Kučera, A., Schwoon, S.: Model-Checking LTL with Regular Valuations for Pushdown Systems. In: Kobayashi, N., Pierce, B.C. (eds.) TACS 2001. LNCS, vol. 2215, pp. 316–339. Springer, Heidelberg (2001)
12. Frougny, C., Sakarovitch, J.: Synchronized rational relations of finite and infinite words. Theor. Comput. Sci. 108(1), 45–82 (1993)
13. Johnson, J.H.: Rational equivalence relations. Theor. Comput. Sci. 47(3), 39–60 (1986)
14. Khoussainov, B., Nerode, A.: Automatic presentations of structures. In: Leivant, D. (ed.) LCC 1994. LNCS, vol. 960, pp. 367–392. Springer, Heidelberg (1994)
15. Kesten, Y., Maler, O., Marcus, M., Pnueli, A., Shahar, E.: Symbolic model checking with rich assertional languages. Theor. Comput. Sci. 256(1-2), 93–112 (2001)
16. Kuper, G.M., Vardi, M.Y.: On the complexity of queries in the logical data model. In: Gyssens, M., Van Gucht, D., Paredaens, J. (eds.) ICDT 1988. LNCS, vol. 326, pp. 267–280. Springer, Heidelberg (1988)
17. Kupferman, O., Vardi, M.Y., Wolper, P.: An automata-theoretic approach to branching-time model checking. Journal of the ACM 47(2), 312–360 (2000)
18. Martin, J.C.: Introduction to Languages and the Theory of Computation, 3rd edn., pp. 186–189. McGraw-Hill, Inc., New York (2002)
19. Meyer, A.R., Stockmeyer, L.J.: Word problems requiring exponential time: Preliminary report. In: Aho, A.V., et al. (eds.) Proc. STOC 1973, pp. 1–9 (1973)
20. Morvan, C.: On Rational Graphs. In: Tiuryn, J. (ed.) FOSSACS 2000. LNCS, vol. 1784, pp. 252–266. Springer, Heidelberg (2000)
21. Reisig, W.: Petri nets: An Introduction. Springer, New York (1985)

22. Thomas, W.: Constructing infinite graphs with a decidable mso-theory. In: Rovan, B., Vojtáš, P. (eds.) MFCS 2003. LNCS, vol. 2747, pp. 113–124. Springer, Heidelberg (2003)
23. Vardi, M.: An automata-theoretic approach to linear temporal logic. In: Moller, F., Birtwistle, G. (eds.) Logics for Concurrency: Structure versus Automata (8th Banff Higher Order Workshop). LNCS, vol. 1043, pp. 238–266. Springer, Heidelberg (1996)
24. Walukiewicz, I.: Model checking CTL properties of pushdown systems. In: Kapoor, S., Prasad, S. (eds.) FST TCS 2000. LNCS, vol. 1974, pp. 127–138. Springer, Heidelberg (2000)