



UvA-DARE (Digital Academic Repository)

Epistemic modelling and protocol dynamics

Wang, Y.

Publication date
2010

[Link to publication](#)

Citation for published version (APA):

Wang, Y. (2010). *Epistemic modelling and protocol dynamics*. [Thesis, fully internal, Universiteit van Amsterdam]. Institute for Logic, Language and Computation.

General rights

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations

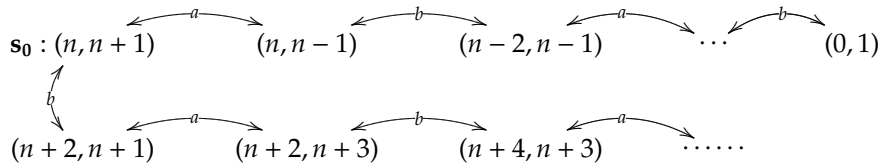
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

8.1 Introduction

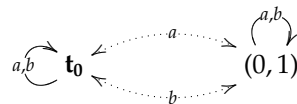
In this chapter we look at a particular technique of PDL_{Σ} abstraction. Since DEL can be translated back to PDL, the technique we will develop here can be adapted to DEL model checking. As demonstrated in the previous chapter, we can use three-valued logic to reason about properties using abstract models with *may*- and *must*-transitions. According to the 3-valued semantics of modal logic, universal (safety) properties $\Box\phi$ are checked w.r.t the over-approximation (may-transitions), while existential (liveness) properties $\Diamond\phi$ are checked w.r.t the under-approximation (must-transitions). This works fine for *safety* properties, but the verification of *liveness* properties is problematic. The problem comes from the lack of guaranteed (“must”) behaviours, due to the non-determinism introduced by abstraction. Consider the following example:

8.1.1. EXAMPLE. (Guessing the other number) Agents a and b have the natural numbers n and $n + 1$ respectively. They only know their own numbers. They are told that what they have are two consecutive natural numbers, but they do not know who has the bigger one. Ω

We can build the following model (suppose n is an even number):

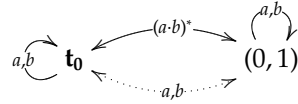


If we want to abstract away all the intermediate states except the *terminal* one $(0, 1)$ then we have the following model:



where t_0 is the abstraction of all the states in the original model except $(0, 1)$. It is not hard to see that there cannot be any must relations in such an abstract model except the reflexive ones at $(0, 1)$. Therefore we cannot get a definite answer to the model checking problem of the formula $\widehat{C}_{a,b}has_a0$ (in PDL: $\langle a+b \rangle^*has_a0$) on the abstract model.

To deal with this problem, Espada and van de Pol proposed *accelerated modal LTS* (accModal-LTS), a new formalism to represent abstractions [EvdP06]. They enhance KMLTSs by labelling must-transitions with *regular expressions* over basic actions. These so-called *accelerated* transitions capture the idea that a state *must* be reached from another state by *some* finite computation contained in the language of the corresponding regular expression. This extension of the abstraction enables us to capture the concrete models more accurately and infer more *liveness* properties. Consider example 8.1.1 again. One could introduce an accelerated must-transition from t_0 to $(0, 1)$ in the abstract model as follows:



Intuitively, this must-transition means, in any state abstracted by t_0 there is a finite $\xrightarrow{a} \xrightarrow{b} \xrightarrow{a} \xrightarrow{b} \dots$ path to the state $(0, 1)$. According to the 3-valued PDL $_{\Sigma}$ semantics defined in [EvdP06], $\langle \pi \rangle \phi$ is true at a point s if there is a *must*-path $s \xrightarrow{\pi_1} s_1 \xrightarrow{\pi_2} \dots \xrightarrow{\pi_n} s_n$ to a state where ϕ is true, such that $\mathcal{L}(\pi_1 \dots \pi_n) \subseteq \mathcal{L}(\pi)$. Thus we can verify that $\langle (a+b)^* \rangle has_a0$ is indeed true on the above abstract model. Based on a suitable abstraction relation, [EvdP06] shows that we can safely reason about the properties of the concrete models by model checking the same properties on their abstractions.

In [EvdP06], the authors also gave a non-trivial model checking algorithm for 3-valued PDL $_{\Sigma}$ on accModal-LTSs, showing decidability of the model checking problem. The most intricate part of the algorithm deals with finding must-paths that comply with the regular expression π in a diamond formula $\langle \pi \rangle \phi$ which is quite different from the usual PDL $_{\Sigma}$ model checking algorithm (see, e.g., [Lan06]). A hard problem left open in [EvdP06] is the precise complexity and optimality of the algorithm.

To understand the behaviour of the accelerated transitions better, in this chapter, we consider PDL $_{\Sigma}$ defined on 2-valued models with accelerated transitions only, which we call *Accelerated Kripke Models (AKM)*. Note that in our AKM, we have only one type of relation as in the standard Kripke models. Thus an AKM can be viewed as the *must* part of an accModal-LTS with 2-valued valuations on each state. The model checking algorithm of PDL $_{\Sigma}$ on AKM can be easily adapted to the original three-valued setting as in [EvdP06].

Main contributions. As we will see, PDL $_{\Sigma}$ interpreted over an AKM behaves quite differently from standard PDL. Developing a model checking algorithm is of utmost importance. Moreover, for an in-depth understanding of the logic, axiomatization and satisfiability checking are two central questions. We address all of these problems.

In Section 3, we first reduce model checking PDL_{Σ} on AKM to model checking PDL on standard Kripke models, by exploiting the notion of *regular expression rewriting* studied extensively in [CDGLV02]. We then provide an automata theoretical model checking algorithm whose complexity can be easily analyzed, namely, in EXPSpace . Furthermore, we prove an EXPSpace lower bound for the model checking problem. These results solve the open problem on model checking left in [EvdP06] and establish a strong link between model checking PDL_{Σ} over AKM and regular expression rewriting. In Section 4, we provide an axiomatization of PDL_{Σ} on AKM, which employs *Kleene Algebra* [Koz91] as an oracle. The soundness and completeness of this system are shown. This result shows very clearly the differences with standard PDL_{Σ} on Kripke models. Furthermore, in Section 5, we study the decision problem of satisfiability. For satisfiability, again, by resorting to the notion of regular expression rewriting, we reduce this problem to the satisfiability of PDL_{Σ} in the standard semantics over Kripke models. We show that the satisfiability of a PDL_{Σ} formula over AKM can be checked in 3-EXPTIME .

Related work. Finite-state automata that allow more complex transition labels recently received a resurgence of attention. These include *generalized automata* [GM99] (a.k.a. string or lazy automata) with strings (or blocks) as transition labels rather than merely characters or the null string and *expression automata* [HW05], finite-state automata whose transition labels are regular expressions over the input alphabet. However, these have been studied from the automata and language perspectives. In particular, the determinism and minimization problems are explored there. In logic, [Mat03] studies μ -calculus with regular expressions in the modalities. It is shown that in this case, regular expressions in formulae can be easily eliminated by the fixpoint construction. [LS07] introduces the notion of regular linear temporal logic, which is a logic that generalizes linear temporal logic with the ability to use regular expressions arbitrarily as sub-expressions. The expressiveness and satisfiability of this logic are investigated there. These works are orthogonal to the use of regular expressions in LTSs, which is the main focus of this chapter. Another work which extends the transitions in LTS is [Lod95], in which the authors study PDL_{Σ} on *distributed transition systems* where the transition relations are labelled with a finite set of actions, representing the fact that the actions occur as a concurrent step. However, the semantics of PDL_{Σ} in [Lod95] is quite different from ours, due to the different interpretation of the non-standard transitions.

8.2 Preliminaries

8.2.1 PDL on AKM

8.2.1. DEFINITION. (Accelerated Kripke model) An *Accelerated Kripke model* (AKM) is a tuple $\mathcal{M} = (S, \mathbf{P}, \Sigma, \rightarrow, V)$ where:

- S, \mathbf{P}, Σ, V are as usual;

- \rightarrow is a possibly infinite set of *accelerated* transitions of the form $s \xrightarrow{\pi} s'$ with $s, s' \in S$, and $\pi \in \text{Reg}_{\Sigma}$ where recall that Reg_{Σ} is the set of (test-free) regular expressions over alphabet Σ (with $a \in \Sigma$):

$$\pi ::= \mathbf{0} \mid \mathbf{1} \mid a \mid \pi + \pi \mid \pi \cdot \pi \mid \pi^*$$

Following the tradition in modal logic, we shall call $\mathcal{F} = (S, \mathbf{P}, \Sigma, \rightarrow)$ an AKM *frame*. As usual, a pointed AKM is an AKM with a designated state $s_0 \in S$: $(S, \mathbf{P}, \Sigma, \rightarrow, V, s_0)$. \mathfrak{M} .

In this chapter, we fix a vocabulary \mathbf{P} , thus we refer to an AKM as $(S, \Sigma, \rightarrow, V)$.

Recall the *test-free* PDL language:

$$\phi ::= \top \mid p \mid \phi \wedge \phi \mid \neg\phi \mid \langle \pi \rangle \phi$$

where π is a regular expression over some alphabet Σ . When Σ is not fixed, we use PDL_{Σ} to denote the test-free PDL language w.r.t the action set Σ .

$\langle \pi \rangle \phi$ is intended to express that there *must* be an execution of π which entails ϕ . Recall that an accelerated transition $s \xrightarrow{\pi} t$ intuitively means that there must be an execution of π from s to t , however, we do not know which execution can do the job. Therefore, assuming $s \xrightarrow{\pi} t$, we can only be sure that there must be an execution of π' to a t world if $\mathcal{L}(\pi) \subseteq \mathcal{L}(\pi')$. The following semantics fleshes out this intention:

$\mathcal{M}, s \vDash p$	\Leftrightarrow	$p \in V(s)$
$\mathcal{M}, s \vDash \neg\phi$	\Leftrightarrow	$s \not\vDash \phi$
$\mathcal{M}, s \vDash \phi \wedge \psi$	\Leftrightarrow	$\mathcal{M}, s \vDash \phi$ and $\mathcal{M}, s \vDash \psi$
$\mathcal{M}, s \vDash \langle \pi \rangle \phi$	\Leftrightarrow	there exists a path $s = s_0 \xrightarrow{\pi_1} s_1 \xrightarrow{\pi_2} \dots \xrightarrow{\pi_n} s_n$ in \mathcal{M} such that $\mathcal{M}, s_n \vDash \phi$ and $\mathcal{L}(\pi_0 \cdot \pi_1 \dots \pi_n) \subseteq \mathcal{L}(\pi)$

To illustrate the semantics, we present two simple examples:

8.2.2. EXAMPLE.

$s \xrightarrow{a+b} \bullet$ $\mathcal{M}, s \vDash \langle a + b + c \rangle \top$ $\mathcal{M}, s \not\vDash \langle a \rangle \top$	$t \xrightarrow{a} \bullet$ $t \xrightarrow{b} \bullet$ $\mathcal{M}, t \vDash \langle a + b + c \rangle \top$ $\mathcal{M}, t \vDash \langle a \rangle \top \wedge \langle b \rangle \top$
---	--

Ω

It is clear that on Kripke models (AKM with only atomic actions as labels), the above semantics coincides with the standard PDL_{Σ} semantics.

8.2.2 Regular Expression Rewriting

The notion of *regular expression rewriting* is introduced in [CDGLV02], and turns out to play an essential role in solving model checking and satisfiability checking problems in this chapter.

Given a regular expression π over an alphabet Σ and a finite set $\mathcal{E} = \{\pi_0, \pi_1, \dots, \pi_n\}$ of regular expressions over the same alphabet Σ , the goal of regular expression rewriting of π over \mathcal{E} is to re-express π , if possible, by a suitable combination of π_0, \dots, π_n using operations \cdot , $+$, and $*$. Given \mathcal{E} , let $\Sigma_{\mathcal{E}}$ be an alphabet containing exactly one unique symbol e_{π} for each π in \mathcal{E} . We shall use $exp_{\Sigma}(e)$ to denote the regular expression associated with the symbol $e \in \Sigma_{\mathcal{E}}$. We can lift exp_{Σ} to any regular expression α over alphabet $\Sigma_{\mathcal{E}}$ in a straightforward way: $exp_{\Sigma}(\alpha)$ is the regular expression over Σ obtained by replacing each occurrence of $e \in \Sigma_{\mathcal{E}}$ with $exp_{\Sigma}(e)$. We let $\mathcal{L}_{\mathcal{E}}(\alpha)$ be the language of α over $\Sigma_{\mathcal{E}}$ (see Definition 2.1.2). Given a regular expression α over $\Sigma_{\mathcal{E}}$, $exp_{\Sigma}(\alpha)$ is called the *expansion* of α . It is clear that $exp_{\Sigma}(e_{\pi_1} \cdot e_{\pi_2} \cdots e_{\pi_n}) = \mathcal{L}(\pi_1 \cdot \pi_2 \cdots \pi_n)$ for $\{\pi_1, \dots, \pi_n\} \subseteq \mathcal{E}$.

Now we define the concept of regular expression rewriting formally:

8.2.3. DEFINITION. (Regular Expression Rewriting) Given a regular expression π over Σ , a set of regular expressions over Σ : $\mathcal{E} = \{\pi_0, \pi_1, \dots, \pi_n\}$, and another regular expression α over the alphabet $\Sigma_{\mathcal{E}}$, we say α is an \mathcal{E} -rewriting of π if $exp_{\Sigma}(\alpha) \subseteq \mathcal{L}(\pi)$. α is called a *maximal \mathcal{E} -rewriting* (notation: $\widehat{\pi}_{\mathcal{E}}$) if for any other \mathcal{E} -rewriting β of π : $\mathcal{L}_{\mathcal{E}}(\beta) \subseteq \mathcal{L}_{\mathcal{E}}(\alpha)$ (thus $exp_{\Sigma}(\beta) \subseteq exp_{\Sigma}(\alpha)$). We say that a rewriting α is *empty* if $\mathcal{L}_{\mathcal{E}}(\alpha) = \emptyset$. \blacksquare

Note that given \mathcal{E} and π , there is a unique maximal \mathcal{E} -rewriting of π (modulo language equivalence over $\Sigma_{\mathcal{E}}$), for otherwise suppose there are two different maximal rewritings α, β . Then $\alpha + \beta$ is also an \mathcal{E} -rewriting of π , which contradicts the maximality of α and β . Moreover, we have the following straightforward result:

8.2.4. PROPOSITION. *If $\mathcal{L}(\pi_1 \cdot \pi_2 \cdots \pi_n) \subseteq \mathcal{L}(\pi)$ and $\{\pi_1, \pi_2, \dots, \pi_n\} \subseteq \mathcal{E}$ then $e_{\pi_1} \cdots e_{\pi_n} \in \mathcal{L}_{\mathcal{E}}(\widehat{\pi}_{\mathcal{E}})$,*

PROOF Towards a contradiction suppose $\mathcal{L}(\pi_1 \cdots \pi_n) \subseteq \mathcal{L}(\pi)$ and $\{\pi_1, \dots, \pi_n\} \subseteq \mathcal{E}$, but $e_{\pi_1} \cdots e_{\pi_n} \notin \mathcal{L}_{\mathcal{E}}(\widehat{\pi}_{\mathcal{E}})$. Let $\alpha = \widehat{\pi}_{\mathcal{E}} + (e_{\pi_1} \cdots e_{\pi_n})$. It is clear that α is another \mathcal{E} -rewriting of π such that $\mathcal{L}_{\mathcal{E}}(\widehat{\pi}_{\mathcal{E}}) \subset \mathcal{L}_{\mathcal{E}}(\alpha)$, which contradicts the maximality of $\widehat{\pi}_{\mathcal{E}}$. \spadesuit

The following two theorems are from [CDGLV02]:

8.2.5. THEOREM. ([CDGLV02]) *The problem of verifying the existence of a non-empty rewriting of a regular expression π' w.r.t. a set \mathcal{E} of regular expressions is EXPSPACE-complete.*

8.2.6. THEOREM. ([CDGLV02]) *There is an essentially optimal algorithm to compute the maximal \mathcal{E} -rewriting of a given π w.r.t. a given set \mathcal{E} in 2-EXPTIME.*

8.3 Model Checking

In this section, we tackle the model checking problem. At first sight, one might think this is a simple problem: an immediate idea might be to first transform an AKM into a Kripke model by replacing every accelerated transition labelled by π with the corresponding (deterministic) automaton of π , then run a traditional model checking algorithm. However, this does *not* work, at least not in a naive way. Let us look at the left figure in Example 8.2.2. Suppose one wants to check $\langle a \rangle \top$, which is *false* at s , following the above idea, one can obtain a Kripke model in the right figure. However, the result will be *true*. This example suggests that the model checking cannot be performed in a very simple way.

8.3.1 A Reduction to Standard PDL $_{\Sigma}$ Model Checking

We now present a non-trivial method to reduce the model checking problem of PDL $_{\Sigma}$ over AKM to the one over Kripke models. Here, as said, the notion of regular expression rewriting is crucially exploited.

Given an AKM $\mathcal{M} = (S, \Sigma, \rightarrow, V)$, let :

$$\langle \rangle_{\mathcal{M}} = \{\pi \mid \pi \in \text{Reg}_{\Sigma} \text{ and } \pi \text{ appears as a label for some transition in } \mathcal{M}\}$$

We define $\ulcorner \mathcal{M} \urcorner$ as $(S, \{e_{\pi} \mid \pi \in \langle \rangle_{\mathcal{M}}\}, \rightarrow', V)$ where $s \xrightarrow{e_{\pi}} s'$ iff $s \xrightarrow{\pi} s'$. If $\langle \rangle_{\mathcal{M}}$ is finite then we can compute the maximal $\langle \rangle_{\mathcal{M}}$ -rewriting of any regular expressions $\pi \in \text{Reg}_{\Sigma}$ in 2-EXPTIME, according to Theorem 8.2.6. Then we can rewrite a PDL $_{\Sigma}$ -formula w.r.t to a model as follows:

8.3.1. DEFINITION. (Rewriting w.r.t an AKM) Given an AKM \mathcal{M} and a PDL $_{\Sigma}$ formula ϕ , $\mathfrak{R}_{\mathcal{M}}(\phi)$ is the rewriting of ϕ in the language PDL $_{\Sigma, \langle \rangle_{\mathcal{M}}}$ defined by :

- $\mathfrak{R}_{\mathcal{M}}(p) = p$ for $p \in \mathbf{P}$;
- $\mathfrak{R}_{\mathcal{M}}(\neg\psi) = \neg\mathfrak{R}_{\mathcal{M}}(\psi)$;
- $\mathfrak{R}_{\mathcal{M}}(\psi_1 \wedge \psi_2) = \mathfrak{R}_{\mathcal{M}}(\psi_1) \wedge \mathfrak{R}_{\mathcal{M}}(\psi_2)$;
- $\mathfrak{R}_{\mathcal{M}}(\langle \pi \rangle \psi) = \langle \widehat{\pi}_{\langle \rangle_{\mathcal{M}}} \rangle \mathfrak{R}_{\mathcal{M}}(\psi)$.

□

Let \models denote the standard PDL $_{\Sigma}$ semantics on Kripke models (cf. Section 2.3.1), then we have:

8.3.2. THEOREM. For any pointed AKM \mathcal{M}, s and any PDL $_{\Sigma}$ formula ϕ ,

$$\mathcal{M}, s \models \phi \iff \ulcorner \mathcal{M} \urcorner, s \models \mathfrak{R}_{\mathcal{M}}(\phi).$$

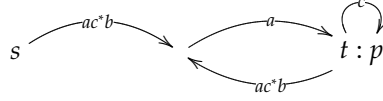


Figure 8.1: Accelerated LTS

PROOF By induction on the structure of ϕ . The only interesting case is $\phi = \langle \pi \rangle \psi$.

(\Rightarrow): Suppose $\mathcal{M}, s \models \langle \pi \rangle \psi$ then there exists some t in \mathcal{M} such that $s \xrightarrow{\pi_1} \dots \xrightarrow{\pi_n} t$ in \mathcal{M} , $\mathcal{L}(\pi_1 \dots \pi_n) \subseteq \mathcal{L}(\pi)$, and $\mathcal{M}, t \models \psi$. From proposition 8.2.4, it is not hard to see that $e_{\pi_1} \dots e_{\pi_n} \in \mathcal{L}_{\langle \rangle_{\mathcal{M}}}(\widehat{\pi}_{\langle \rangle_{\mathcal{M}}})$. By the induction hypothesis, $\ulcorner \mathcal{M} \urcorner, t \Vdash \mathfrak{R}_{\mathcal{M}}(\psi)$ and thus $\ulcorner \mathcal{M} \urcorner, s \Vdash \langle \widehat{\pi}_{\langle \rangle_{\mathcal{M}}} \rangle \mathfrak{R}_{\mathcal{M}}(\psi)$. Namely $\ulcorner \mathcal{M} \urcorner, s \Vdash \mathfrak{R}_{\mathcal{M}}(\phi)$.

(\Leftarrow): Suppose $\ulcorner \mathcal{M} \urcorner, s \Vdash \langle \widehat{\pi}_{\langle \rangle_{\mathcal{M}}} \rangle \mathfrak{R}_{\mathcal{M}}(\psi)$, then there exists a path $s \xrightarrow{e_{\pi_1}} \dots \xrightarrow{e_{\pi_n}} t$ in $\ulcorner \mathcal{M} \urcorner$ such that $e_{\pi_1} \dots e_{\pi_n} \in \mathcal{L}_{\langle \rangle_{\mathcal{M}}}(\widehat{\pi}_{\langle \rangle_{\mathcal{M}}})$ with $\{\pi_1, \dots, \pi_n\} \subseteq \langle \rangle_{\mathcal{M}}$. It follows that $\text{exp}_{\Sigma}(e_{\pi_1} \dots e_{\pi_n}) \subseteq \text{exp}_{\Sigma}(\widehat{\pi}_{\langle \rangle_{\mathcal{M}}})$. Since $\widehat{\pi}_{\langle \rangle_{\mathcal{M}}}$ is a $\langle \rangle_{\mathcal{M}}$ -rewriting, $\mathcal{L}(\pi_1 \dots \pi_n) \subseteq \mathcal{L}(\pi)$. By the definition of $\ulcorner \mathcal{M} \urcorner$, $s \xrightarrow{\pi_1} \dots \xrightarrow{\pi_n} t$ in \mathcal{M} . By the induction hypothesis, $\mathcal{M}, t \models \psi$, and thus $\mathcal{M}, s \models \langle \pi \rangle \psi$. \spadesuit

Theorem 8.3.2 allows us to use the standard PDL_{Σ} model checking algorithm (e.g. [Lan06]) to solve the problem over AKM in a straightforward manner. We present an example here. Let us consider the AKM \mathcal{M} depicted in Fig. 8.1. Suppose we need to check whether the formula $\phi = \langle a \cdot (b \cdot a + c)^* \rangle p$ holds at state s . We first collect the set $\langle \rangle_{\mathcal{M}} = \{a, a \cdot c^* \cdot b, c\}$; then we compute the maximal rewriting of $a \cdot (b \cdot a + c)^*$ w.r.t $\langle \rangle_{\mathcal{M}}$, following the algorithm of generating the maximal rewriting in [CDGLV02]. It follows that $\mathfrak{R}_{\mathcal{M}}(\phi) = \langle (e_{a \cdot c^* \cdot b})^* \cdot e_a \cdot (e_c)^* \rangle p^1$. According to Theorem 8.3.2, we only need to check whether $\ulcorner \mathcal{M} \urcorner, s \Vdash \mathfrak{R}_{\mathcal{M}}(\phi)$, where $\ulcorner \mathcal{M} \urcorner$ is the same graph as in Fig.8.1 except that the labels become $e_{a \cdot c^* \cdot b}$, e_a and e_c respectively. A standard PDL_{Σ} model checking algorithm will return **TRUE** and thus we can conclude that $\mathcal{M}, s \models \phi$.

8.3.2 A Direct Algorithm

Due to Theorem 8.2.6, the above translation $\mathfrak{R}_{\mathcal{M}}$ is quite expensive (2-EXPTIME). To avoid generating the maximal rewriting explicitly, we may process the rewriting and the model checking at the same time and check non-emptiness of the rewriting when needed. Based on this idea, we now present a more efficient direct algorithm, which shares the same basic structure as those proposed in literature for branching-time temporal logic (typically CTL, see e.g. [CGP99] for a clear exposition). In a nutshell, given a formula ϕ , the algorithm recursively evaluates the truth-values of the subformulas ψ of ϕ at all states, starting from the propositional formulas of ϕ and following the recursive definitions of each modality. The whole process will be gathered up in a global labelling algorithm. It turns out that the central part of the algorithm is to solve the following question:

¹Actually $e_{a \cdot c^* \cdot b}^* \cdot e_a \cdot e_c^*$ is an exact $\langle \rangle_{\mathcal{M}}$ -rewriting of $a \cdot (b \cdot a + c)^*$ (cf. [CDGLV02]).

Exists(\mathcal{M}, s, T, π_0): Given a pointed AKM $\mathcal{M} = (S, \Sigma, \rightarrow, V, s_0)$, a set of states $T \subseteq S$ and a regular expression π_0 , check whether there exists a sequence of transitions $s \xrightarrow{\pi_1} s_1 \cdots \xrightarrow{\pi_n} t$ such that $\mathcal{L}(\pi_1 \cdots \pi_n) \subseteq \mathcal{L}(\pi_0)$ and $t \in T$. **Exists**(\mathcal{M}, s, T, π_0) returns \top if the answer is yes and returns \perp otherwise.

In the sequel, we deal with this problem by an automata-theoretic approach. A sketch is as follows:

1. Construct a *deterministic* automaton (DFA) A_{π_0} such that $\mathcal{L}(A_{\pi_0}) = \mathcal{L}(\pi_0)$;
2. Define a suitable product $\mathcal{M} \otimes A_{\pi_0}$ of \mathcal{M} and A_{π_0} ;
3. Run the emptiness check on $\mathcal{M} \otimes A_{\pi_0}$.

Now, we present the detailed construction step by step. Step 1 is standard (cf., e.g., [Con71]). We start from Step 2.

8.3.3. DEFINITION. (Product \otimes_T) Given a pointed AKM $\mathcal{M} = (S, Act, \rightarrow, s_0)$ and $T \subseteq S$, a DFA $A = (Q, \Sigma, \delta, q_0, F)$ (cf., Definition. 2.1.1), define $\mathcal{M} \otimes_T A$ as the nondeterministic automaton $(G, \Sigma', \rho, q'_0, F')$ where:

- $G = S \times \mathcal{P}(Q)$;
- $\Sigma' = \{e_\pi \mid \pi \text{ appears in the transition of } \mathcal{M}\}$;
- $(\langle s, U \rangle, e_\pi, \langle t, R \rangle) \in \rho \iff s \xrightarrow{\pi} t \text{ and } R = \bigcup_{u \in U} \bigcup_{w \in \mathcal{L}(\pi)} \{\delta^*(u, w)\}$;
- $q'_0 = \langle s_0, \{q_0\} \rangle$; and
- $F' = \{\langle s, U \rangle \mid s \in T \text{ and } U \subseteq F\}$.

δ^* is the extended transition function in a deterministic automaton such that given $u \in Q$ and word $w \in \Sigma^*$, $\delta^*(q, w)$ gives the unique state that can be reached from q by a w path. □

In the above construction, although $\mathcal{L}(\pi)$ may be infinite, we can still compute $\bigcup_{w \in \mathcal{L}(\pi)} \{\delta^*(u, w)\}$. Given an $u \in Q_A$ and a regular expression π , let A^u be the DFA just as A but with the new start state u , and let A' be a nondeterministic automaton such that $\mathcal{L}(A') = \mathcal{L}(\pi)$. It is not hard to see that we can compute $\bigcup_{w \in \mathcal{L}(\pi)} \delta^*(u, w)$ by collecting the $u' \in Q_A$ in the reachable final states of the standard product of A^u and A' (cf. e.g., [Con71]).

Step 3, the emptiness checking for a nondeterministic automaton can be done in a standard and efficient way. We are in a position to present the correctness of the whole procedure.

8.3.4. PROPOSITION. *Given a pointed AKM $\mathcal{M} = (S, \Sigma, \rightarrow, V, s_0)$ and $T \subseteq S$, if $T = \{t \mid \mathcal{M}, t \models \phi\}$, then we have:*

$$\mathcal{M}, s_0 \models \langle \pi_0 \rangle \phi \iff \mathcal{L}(\mathcal{M} \otimes_T \mathbf{A}_{\pi_0}) \neq \emptyset$$

where \mathbf{A}_{π_0} denotes the deterministic automaton corresponding to π_0 .

PROOF Let $\mathbf{A}_{\pi_0} = (Q, \Sigma, \delta, q_0, F)$ be the deterministic automaton of π_0 .

(\Rightarrow): Since $\mathcal{M}, s_0 \models \langle \pi_0 \rangle \phi$, there is a path $s_0 \xrightarrow{\pi_1} s_1 \xrightarrow{\pi_2} \dots \xrightarrow{\pi_n} s_n$ in \mathcal{M} such that $s_n \in T$ and $\mathcal{L}(\pi_1 \dots \pi_n) \subseteq \mathcal{L}(\pi_0)$. It follows that for any $w_1 \dots w_n \in \mathcal{L}(\pi_1 \dots \pi_n)$ where $w_i \in \mathcal{L}(\pi_i)$, $w_1 \dots w_n \in \mathcal{L}(\mathbf{A}_{\pi_0})$.

We define $U_0 = \{q_0\}$ and $U_{i+1} = \bigcup_{u \in U_i} \bigcup_{w \in \mathcal{L}(\pi_i)} \{\delta^*(u, w)\}$. Now consider the trace:

$$\langle s_0, U_0 \rangle \xrightarrow{e_{\pi_1}} \langle s_1, U_1 \rangle \xrightarrow{e_{\pi_2}} \dots \xrightarrow{e_{\pi_n}} \langle s_n, U_n \rangle \quad (\#)$$

Clearly, this is a path in $\mathcal{M} \otimes_T \mathbf{A}_{\pi_0}$. Moreover, for each state $q \in U_n$, there must exist $q_0, q_1, \dots, q_n = q$ with w_1, w_2, \dots, w_n such that for each i , $q_i \in U_i$ and $q_{i+1} = \delta^*(q_i, w_i)$ and $w_i \in \mathcal{L}(\pi_i)$. Therefore $w_1 \cdot w_2 \dots w_n \in \mathcal{L}(\pi_0)$. and then $w_1 \cdot w_2 \dots w_n \in \mathcal{L}(\pi_0)$. Thus $w_1 \cdot w_2 \dots w_n$ is accepted by \mathbf{A}_{π_0} . Since \mathbf{A}_{π_0} is deterministic, q must be an accept state, namely, $q \in F$. It follows that $U_n \subseteq F$. Since $s_n \in T$, (s_n, U_n) is an accept state in $\mathcal{M} \otimes_T \mathbf{A}_{\pi_0}$. Therefore the above path (#) is an accepting path, i.e. $\mathcal{L}(\mathcal{M} \otimes_T \mathbf{A}_{\pi_0}) \neq \emptyset$.

(\Leftarrow): Suppose $\mathcal{L}(\mathcal{M} \otimes_T \mathbf{A}_{\pi_0}) \neq \emptyset$, then there exists some path:

$$\langle s_0, U_0 \rangle \xrightarrow{e_{\pi_1}} \langle s_1, U_1 \rangle \xrightarrow{e_{\pi_2}} \dots \xrightarrow{e_{\pi_n}} \langle s_n, U_n \rangle$$

such that (s_n, U_n) is an accept state, namely, $s_n \in T$ and $U_n \subseteq F$. It follows from the definition that $s_0 \xrightarrow{\pi_1} s_1 \xrightarrow{\pi_2} \dots \xrightarrow{\pi_n} s_n$ in \mathcal{M} , and for each $w_i \in \mathcal{L}(\pi_i)$, each $u_i \in U_i$, $\delta^*(u_i, w_i) \in U_{i+1}$. Hence for any $w_1 \dots w_n \in \mathcal{L}(\pi_1 \dots \pi_n)$, we can construct a sequence of states q_0, q_1, \dots, q_n such that $\delta^*(q_i, w_i) = q_{i+1}$ and $q_i \in U_i$. Since $U_n \subseteq F$, $q_n \in F$. Namely, $w_1 \dots w_n$ is accepted by \mathbf{A}_{π_0} and thus $w_1 \dots w_n \in \mathcal{L}(\pi_0)$. Therefore $\mathcal{L}(\pi_1 \dots \pi_n) \subseteq \mathcal{L}(\pi_0)$, namely $\mathcal{M}, s_0 \models \langle \pi_0 \rangle \phi$. \spadesuit

Note that in the above correctness proof, we rely on the property of *determinism*. It is crucial that the transition function assigns to each state and each label one and only one successor.

Algorithm. We have presented how to construct the function $\text{Exists}(\mathcal{M}, s, T, \pi)$ and now we are in the position to give the full algorithm, as defined in Algorithm 1. The termination of the algorithm is clear and thus the correctness can be ensured by the following theorem:

8.3.5. THEOREM. *Given a pointed AKM $\mathcal{M} = (S, \text{Act}, \rightarrow, s_0)$, and a PDL formula ϕ .*

$$s_0 \in \text{eval}(\phi) \iff s_0 \models \phi$$

Algorithm 1 Model Checking Algorithm

Input: A pointed AKM $\mathcal{M} = (S, Act, \rightarrow, s_0)$, and a PDL $_{\Sigma}$ formula ϕ .

return $s_0 \in eval(\phi)$;

where

Function $eval(\phi)$

If $\phi = \top$, then **return** S ;

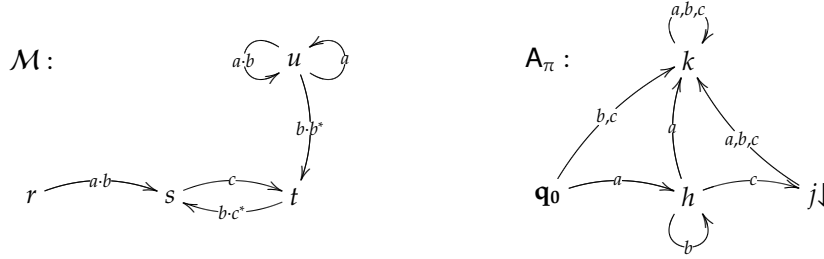
If $\phi = p$, then **return** $\{s \in S \mid p \in V(s)\}$;

If $\phi = \neg\phi'$, then **return** $S \setminus eval(\phi')$;

If $\phi = \phi' \wedge \phi''$, then **return** $eval(\phi') \cap eval(\phi'')$;

If $\phi = \langle \pi \rangle \phi'$, then **return**
 $\{s \in S \mid \mathbf{Exists}(\mathcal{M}, s, eval(\phi'), \pi) = \top\}$;

We end this section by presenting an example, originally appearing in [EvdP06]. Let us consider the AKM \mathcal{M} below (the valuation function is not essential for this example and we name the states as r, s, t , etc.):



We demonstrate how to compute $\mathbf{Exists}(\mathcal{M}, r, T, \pi)$, where $\pi = a \cdot b^* \cdot c$ and $T = \{t\}$. The first step is to transform π into a complete deterministic automaton A_{π} (the right graph above, where q_0 is the start state and j is the accept state). Then we can compute the product, which is simply:

$$(r, \{q_0\}) \xrightarrow{e_{a \cdot b}} (s, \{h\}) \xrightarrow{e_c} (t, \{j\}) \xrightarrow{e_{b \cdot c^*}} (s, \{k\}) \xrightarrow[e_{b \cdot c^*}]{e_c} (t, \{k\})$$

with $(t, \{j\})$ the accept state (note that by an on-the-fly construction, the unreachable parts are omitted). At last, the emptiness checking yields YES. It is clear that our algorithm is much simpler than the one presented in [EvdP06].

8.3.3 Complexity Analysis

Upper Bounds. Let us analyse the complexity of the algorithm presented above for model checking PDL $_{\Sigma}$ over AKMs. As we mentioned before, the essential part of the algorithm is the oracle $\mathbf{Exists}(\mathcal{M}, s, T, \pi)$. We observe that

- (1) for a regular expression π , the deterministic automaton A_{π} is of exponential size $O(2^{|\pi|})$;

- (2) the product $\mathcal{M} \otimes_T \mathbf{A}_\pi$ is of the size $\mathcal{O}(|\mathcal{M}| \cdot 2^{2^{|\pi|}})$;
- (3) Checking emptiness can be done in nondeterministic logarithmic space.

To glue them together, we obtain a nondeterministic ExpSPACE bound, and using Savitch's theorem, we get a deterministic ExpSPACE bound for the oracle². Moreover, as in the traditional algorithm for CTL, the main algorithm presented in Algorithm 1 can be done in P time with an ExpSPACE -bounded oracle. So the complexity is P^{ExpSPACE} , which is ExpSPACE .

One might think the complexity is a bit too high in practice. However, Lichtenstein and Pnueli argued that when analyzing the complexity of model checking, a distinction should be made between complexity in the size of the input structure and complexity in the size of the input formula. And it is often the complexity in the size of the structure that is typically the computational bottleneck [LP85]. In a nutshell, *program complexity* refers to the complexity of the problem in terms of the size of the input module, assuming the formula is *fixed*. Clearly, in our case, the program complexity turns out to be LogSPACE . This is important in practice since people might argue that the complexity of our algorithm is too high to be practical. However, in practice, usually the logic formula is small and in this case the algorithm still performs very well.

Lower Bound. We show that the upper bound established above is essentially optimal. We shall exploit the regular expression rewriting problem (see Section 8.2.2) to prove the ExpSPACE lower bound of the problem of model checking AKM w.r.t. a PDL_Σ formula.

We present a reduction as follows:

8.3.6. LEMMA. *Given a set of non-empty regular expressions $\mathcal{E} = \{\pi_1, \dots, \pi_k\}$ and another regular expression π over Σ , there exists a pointed AKM model $(\mathcal{M}_\mathcal{E}, s)$ and a PDL_Σ formula ϕ such that:*

$$\mathcal{M}_\mathcal{E}, s \models \langle \pi \rangle \phi \iff \text{there is a non-empty rewriting of } \pi \text{ w.r.t. } \mathcal{E}.$$

PROOF Given $\mathcal{E} = \{\pi_1, \dots, \pi_k\}$ and π , we define the AKM $\mathcal{M}_\mathcal{E}$ as

$$(\{s\}, \mathcal{E}, \rightarrow, V)$$

where $\rightarrow = \{(s, \pi_i, s) \mid \pi_i \in \mathcal{E}\}$, V is an arbitrary valuation. Let $\phi = \langle \pi \rangle \top$.

(\Rightarrow .) Suppose $\mathcal{M}_\mathcal{E}, s \models \langle \pi \rangle \top$. According to the semantics, there is a path in $\mathcal{M}_\mathcal{E}$ with $s \xrightarrow{\pi'_1} s \cdots \xrightarrow{\pi'_m} s$ where $\{\pi'_1, \dots, \pi'_m\} \subseteq \mathcal{E}$ and $\mathcal{L}(\pi'_1 \cdots \pi'_m) \subseteq \mathcal{L}(\pi)$. It follows that $e_{\pi'_1} \cdots e_{\pi'_m}$ is a non-empty rewriting of π w.r.t. \mathcal{E} .

²Note that some care is needed to get the claimed space bound. We cannot simply construct \mathbf{A}_π since it is of doubly exponential size. Instead, we construct \mathbf{A}_π on the fly.

(\Leftarrow): Suppose there is a non-empty rewriting α of π w.r.t. \mathcal{E} . Since α is non-empty, there is a possibly empty word $e_{\pi'_1} \cdots e_{\pi'_m} \in \mathcal{L}(\alpha)$ where for each $1 \leq i \leq m$, $\pi'_i \in \mathcal{E}$. It is easy to see that $\text{exp}_{\Sigma}(e_{\pi'_1} \cdots e_{\pi'_m}) \subseteq \text{exp}_{\Sigma}(\alpha)$. Furthermore, according to the definition of the rewriting, $\text{exp}_{\Sigma}(\alpha) \subseteq \mathcal{L}(\pi)$ and thus $\mathcal{L}(\pi'_1 \cdots \pi'_m) \subseteq \mathcal{L}(\pi)$. Clearly there exists a path in $\mathcal{M}_{\mathcal{E}}$ with $s \xrightarrow{\pi'_1} s \cdots \xrightarrow{\pi'_m} s$ thus $\mathcal{M}_{\mathcal{E}, s} \models \langle \pi \rangle \top$. This completes the proof. \spadesuit

Based on the EXPSpace upper bound, Theorem 8.2.5 and Lemma 8.3.6 yield the main result of the current section, as follows:

8.3.7. THEOREM. *The problem of model checking a PDL_{Σ} formula w.r.t. an AKM is EXPSpace -complete.*

8.4 Axiomatization

In this section, we give a complete axiomatization of PDL_{Σ} over AKM. Although the syntax of PDL_{Σ} does not change, the interpretation over AKM results in a new semantics which differs from standard PDL_{Σ} considerably. For instance, the following axioms are valid in standard PDL_{Σ} . However, most of them are *not* valid any more (if a \Leftarrow appears in the right column, this indicates that \leftrightarrow should be replaced by \Leftarrow to keep the formula valid).

Axioms	In our semantics
$[\pi](\phi \rightarrow \psi) \rightarrow ([\pi]\phi \rightarrow [\pi]\psi)$	valid
$\langle \pi_1 \cdot \pi_2 \rangle \phi \leftrightarrow \langle \pi_1 \rangle \langle \pi_2 \rangle \phi$	\Leftarrow
$\langle \pi_1 + \pi_2 \rangle \phi \leftrightarrow \langle \pi_1 \rangle \phi \vee \langle \pi_2 \rangle \phi$	\Leftarrow
$\langle \pi^* \rangle \phi \leftrightarrow (\phi \vee \langle \pi \rangle \langle \pi^* \rangle \phi)$	\Leftarrow
$[\pi^*](\phi \rightarrow [\pi]\phi) \rightarrow (\phi \rightarrow [\pi^*]\phi)$	invalid

In view of this, instead of the standard PDL_{Σ} axioms we propose the following new conditional axiomatization.

8.4.1. DEFINITION. A deductive system AS

TAUTOLOGY	all the tautologies
K	$[\pi](\phi \rightarrow \phi') \rightarrow ([\pi]\phi \rightarrow [\pi]\phi')$
SEQ	$[\pi_1 \cdot \pi_2]\phi \rightarrow [\pi_1][\pi_2]\phi$
STAR	$[\pi^*]\phi \rightarrow \phi$
Rules	
\square	$\frac{\phi}{[\pi]\phi}$
MP	$\frac{\phi, \phi \rightarrow \psi}{\psi}$
INCL	$\frac{\vdash_{\text{KA}} \pi + \pi' = \pi'}{[\pi']\phi \rightarrow [\pi]\phi}$

where \mathbf{KA} is a complete Kleene algebra, for example as in [Koz91], acting as an oracle.

The rest of this section is devoted to showing that \mathbf{AS} is sound and complete w.r.t the class of all AKM frames. First let us consider a special class of AKM frames on which we can use an equivalent simple semantics for technical convenience. An AKM frame is called *normal* if it satisfies the following properties:

- **sequentiality:** For any $\pi, \pi' \in \text{Reg}_\Sigma : \pi \circ \pi' \xrightarrow{\subseteq} \pi \xrightarrow{\subseteq} \pi \circ \pi'$ where \circ is concatenation of binary relations;
- ***-reflexivity:** For any $\pi \in \text{Reg}_\Sigma$: if $\{\mathbf{1}\} \in \mathcal{L}(\pi)$ then $s \xrightarrow{\pi} s$ for any $s \in S$;
- **regularity:** For any $\pi, \pi' \in \text{Reg}_\Sigma$: $\mathcal{L}(\pi) \subseteq \mathcal{L}(\pi')$ implies that $\pi \xrightarrow{\subseteq} \pi'$.

Models based on the normal AKM frames are called *normal* AKM models. Now we can define an equivalent semantics \vDash_0 on the normal AKM models as follows:

- For boolean cases: as before;
- For modal case:
 $\mathcal{M}, s \vDash_0 \langle \pi \rangle \phi \iff \exists t : s \xrightarrow{\pi} t \text{ and } t \vDash_0 \phi.$

We can saturate an arbitrary AKM frame of PDL_π : $\mathcal{F} = (S, \Sigma, \rightarrow)$ into a normal frame $R(\mathcal{F}) = (S, \Sigma, \rightarrow_r)$ by adding transitions³:

$$s \xrightarrow{\pi}_r t \iff \exists s \xrightarrow{\pi_1} s_1 \xrightarrow{\pi_2} \dots \xrightarrow{\pi_n} s_n \text{ and } \mathcal{L}(\pi_1 \pi_2 \dots \pi_n) \subseteq \mathcal{L}(\pi)$$

$R(\mathcal{M})$ is the saturated model which keeps the valuation the same but saturates the frame of \mathcal{M} . It is easy to see that \vDash_0 coincides with \vDash on normal models:

8.4.2. PROPOSITION. *Given an AKM $\mathcal{M} = (S, \Sigma, \rightarrow, V)$, for any PDL_Σ formula ϕ :*

$$\mathcal{M}, s \vDash \phi \iff R(\mathcal{M}), s \vDash_0 \phi \iff R(\mathcal{M}), s \vDash \phi$$

Since all the normal AKM frames are AKM frames and all the AKM frames can be saturated into normal AKM frames, it follows from the above proposition that $\Delta \vDash \phi \iff \Delta \vDash_0 \phi$, where Δ is a set of PDL_Σ formulas.

It is easy to check the following lemmata:

8.4.3. LEMMA. *For any normal AKM frame \mathcal{F} and any two regular expressions π and π' , if $\vdash_{\mathbf{KA}} \pi + \pi' = \pi'$ then $\mathcal{F} \vDash_0 [\pi']p \rightarrow [\pi]p$.*

Here $\mathcal{F} \vDash_0 \phi$ iff for any model \mathcal{M} based on \mathcal{F} : $\mathcal{M} \vDash_0 \phi$.

8.4.4. LEMMA. *For any normal AKM frame \mathcal{F} : \mathcal{F} satisfies sequentiality $\iff \mathcal{F} \vDash_0 \text{SEQ}$.*

8.4.5. LEMMA. *For any normal AKM frame \mathcal{F} : \mathcal{F} satisfies *-reflexivity implies $\mathcal{F} \vDash_0 \text{STAR}$.*

³Note that $\mathbf{1}$ denotes for the empty sequence, and for any s : $s \xrightarrow{\mathbf{1}} s$.

From the above lemmata, and the completeness of Kleene Algebra [Koz91], it is straightforward to establish:

8.4.6. THEOREM (SOUNDNESS). *AS is sound for normal AKM frames.*

Note that the STAR axiom does not correspond to $*$ -reflexivity by itself, but in the presence of the other two properties⁴:

8.4.7. LEMMA. *If an AKM frame \mathcal{F} satisfies regularity, sequentiality and $\mathcal{F} \models \text{STAR}$ then \mathcal{F} is normal.*

PROOF Suppose \mathcal{F} satisfies regularity and sequentiality, we only need to show that \mathcal{F} satisfies $*$ -reflexivity: for any regular expression $\pi \in \text{Reg}_\Sigma$, if $\mathbf{1} \in \mathcal{L}(\pi)$ then $\xrightarrow{\pi}$ is reflexive. We prove this by induction on the structure of π .

- If $\pi = \pi'^*$ then it is straightforward to check that $\xrightarrow{\pi}$ is reflexive since $\mathcal{F} \models \text{STAR}$.
- If $\pi = \pi_1 + \pi_2$ then $\mathbf{1} \in \mathcal{L}(\pi_1)$ or $\mathbf{1} \in \mathcal{L}(\pi_2)$. By the induction hypothesis $\xrightarrow{\pi_1}$ is reflexive or $\xrightarrow{\pi_2}$ is reflexive. From regularity, $\xrightarrow{\pi_1} \subseteq \xrightarrow{\pi}$ and $\xrightarrow{\pi_2} \subseteq \xrightarrow{\pi}$. So $\xrightarrow{\pi}$ is reflexive.
- If $\pi = \pi_1 \cdot \pi_2$ then $\mathbf{1} \in \mathcal{L}(\pi_1)$ and $\mathbf{1} \in \mathcal{L}(\pi_2)$. By the induction hypothesis $\xrightarrow{\pi_1}$ and $\xrightarrow{\pi_2}$ are reflexive. From sequentiality, $\xrightarrow{\pi_1} \circ \xrightarrow{\pi_2} \subseteq \xrightarrow{\pi}$. So $\xrightarrow{\pi}$ is reflexive.

✕

Completeness follows from the standard canonical model construction.

8.4.8. THEOREM (COMPLETENESS). *For any set of PDL $_\Sigma$ formulas $\Delta \cup \{\phi\}$: $\Delta \models_0 \phi \implies \Delta \vdash_{\text{AS}} \phi$. Namely AS is strongly complete for normal AKM frames w.r.t \models_0 . Thus AS is strongly complete for all AKM frames.*

PROOF Recall that a logic theory is a *normal modal logic* if it contains all the instances of tautologies, the K axiom and is closed under MP and \Box . Therefore AS induces a normal modal logic. Thus it is strongly complete with respect to its canonical model $\mathcal{M}^c = (S^c, \text{Reg}_\Sigma, \xrightarrow{c}, V^c)$ according to the canonical model theorem (see e.g., [BdRV02, Theorem 4.22]), where S^c is the set of all AS-maximal consistent sets, $s \xrightarrow{\pi} t$ if for all ψ , $\psi \in s \implies \langle \pi \rangle \psi \in t$, $V^c(s) = \{p \mid p \in s\}$. We only need to show that the canonical model \mathcal{M}^c is indeed a model based on a normal AKM frame. Since S^c is the set of AS-maximal consistent sets, $\mathcal{M}^c \models_0 \text{STAR} \wedge \text{SEQ}$. From Lemma 8.4.4 and 8.4.7, we only need to show the canonical model satisfies regularity:

$$\text{For any } \pi, \pi' \in \pi^*, \mathcal{L}(\pi) \subseteq \mathcal{L}(\pi') \text{ implies } \xrightarrow{\pi} \subseteq \xrightarrow{\pi'}$$

⁴That is why we don't include a rule like: $\frac{[\pi]\phi}{\phi}$ if $\epsilon \in \mathcal{L}(\pi)$.

Suppose there are regular expressions π, π' such that $\mathcal{L}(\pi) \subseteq \mathcal{L}(\pi')$ and $\exists s, t : s \xrightarrow{\pi}^c t$ in the canonical model. From the definition of $\xrightarrow{\pi}^c$, we have for all $\psi : \psi \in t \Rightarrow \langle \pi \rangle \psi \in s$. Since $\mathcal{L}(\pi) \subseteq \mathcal{L}(\pi')$ and KA is complete, we have $\vdash_{\text{KA}} \pi + \pi' = \pi'$. Since s is a maximal consistent set, then from INCL we have for all $\psi : \langle \pi \rangle \psi \rightarrow \langle \pi' \rangle \psi \in s$. Therefore by applying MP we have for all $\psi \in t : \langle \pi' \rangle \psi \in s$. It follows, by definition, that $s \xrightarrow{\pi'}^c t$. \times

Strong completeness implies compactness:

8.4.9. COROLLARY (COMPACTNESS). PDL_{Σ} w.r.t AKM is model compact. Namely if all the finite subsets of Γ are satisfiable then Γ is satisfiable.

8.4.10. REMARK. Recall that the standard PDL_{Σ} is not model compact: considering the set $\Gamma = \{\langle a^* \rangle p, \neg p, \neg \langle a \rangle p, \neg \langle a \rangle \langle a \rangle p, \dots\}$, any finite subset of Γ is satisfiable, yet not the whole Γ . However, Γ is satisfiable in the following AKM model:

$$\neg p \xrightarrow{a^*} p$$

8.5 Satisfiability

In this section, we turn to the satisfiability checking problem. The basic idea is to reduce this problem to traditional PDL_{Σ} satisfiability checking. However, clearly this can not be done in a straightforward way, since their semantics do not coincide, as observed in the previous section.

For technical convenience, let us consider the equivalent positive PDL_{Σ}^+ language

$$\phi ::= \top \mid \perp \mid p \mid \bar{p} \mid \phi \wedge \phi \mid \phi \vee \phi \mid [\pi]\phi \mid \langle \pi \rangle \phi$$

where p and \bar{p} (negation of p) are in a set *lit* of literals of basic propositions and $\pi \in \text{Reg}_{\Sigma}$. It is a standard exercise to transform a PDL_{Σ} formula to an equivalent PDL_{Σ}^+ formula and vice versa.

Given a PDL_{Σ}^+ formula ϕ , let $\langle \rangle_{\phi}$ be the set $\{\pi \mid \langle \pi \rangle \text{ appears in } \phi\}$. We now prove that if a formula is satisfiable then it is satisfiable in a certain class of models.

8.5.1. PROPOSITION. *Given a PDL_{Σ}^+ formula ϕ , ϕ is satisfiable on an AKM $\iff \phi$ is satisfiable in an AKM that only contains π -transitions for $\pi \in \langle \rangle_{\phi}$.*

PROOF \Leftarrow is straightforward. We now prove \Rightarrow :

Suppose there is an AKM $\mathcal{M} = (S, \Sigma, \rightarrow, V)$ such that $\mathcal{M}, s \vDash \phi$ for some $s \in S$. From proposition 8.4.2, $R(\mathcal{M}), s \vDash \phi$. Based on $R(\mathcal{M})$ we build the model $\mathcal{M}' = (S, \Sigma', \rightarrow', V)$ where:

$$\Sigma' = \langle \rangle_{\phi} \text{ and } s \xrightarrow{\pi}' t \text{ in } \mathcal{M}' \iff s \xrightarrow{\pi}_r t \text{ in } R(\mathcal{M}).$$

Namely we eliminate all the transitions in $R(\mathcal{M})$ except the ones labelled by some $\pi \in \langle \rangle_{\phi}$. We claim: $\mathcal{M}', s \vDash \phi$. We prove it by induction on the structure of ϕ :

- For atomic and boolean cases, trivial.
- $\phi = \langle \pi \rangle \psi$: since $R(\mathcal{M}), s \models \phi$ then $\exists t \in S$ such that $R(\mathcal{M}), t \models \psi$ and $s \xrightarrow{\pi}_r t$.
By the definition of $\xrightarrow{\cdot}'$, $s \xrightarrow{\pi}' t$. Now by the induction hypothesis we have $\mathcal{M}', t \models \psi$, thus $\mathcal{M}', s \models \phi$.
- $\phi = [\pi] \psi$: since $R(\mathcal{M}), s \models \phi$ then for all t such that $s \xrightarrow{\pi}_r t$, $R(\mathcal{M}), t \models \psi$.
By the induction hypothesis, $\mathcal{M}', t \models \psi$. Note that if there exists t such that $s \xrightarrow{\pi_1}' \dots \xrightarrow{\pi_n}' t$ in \mathcal{M}' , and $\mathcal{L}(\pi_1 \dots \pi_n) \subseteq \mathcal{L}(\pi)$ then $s \xrightarrow{\pi}_r t$ in $R(\mathcal{M})$.
Therefore for all π -reachable states t in S , $\mathcal{M}', t \models \psi$. It follows that $\mathcal{M}', s \models \phi$.

✱

Given a PDL_{Σ}^+ formula ϕ , we define a rewriting of ϕ , obtained by replacing every instance of π in $[\pi] \psi$ with its maximal $\langle \rangle_{\phi}$ -rewriting $\widehat{\pi}_{\langle \rangle_{\phi}}$. Recall that $\widehat{\pi}_{\langle \rangle_{\phi}}$ is a regular expression over the alphabet $\Sigma_{\langle \rangle_{\phi}} = \{e_{\pi} \mid \pi \in \langle \rangle_{\phi}\}$ where each e_{π} is a new symbol.

8.5.2. DEFINITION. (Rewriting of a PDL_{Σ}^+ formula) Given a PDL_{Σ}^+ formula ϕ , $\mathfrak{R}(\phi)$ is the rewriting of ϕ in the language $\text{PDL}_{\Sigma_{\langle \rangle_{\phi}}}^+$ defined by:

- $\mathfrak{R}(p) = p$ where $p \in \text{lit} \cup \{\top, \perp\}$.
- $\mathfrak{R}(\psi_1 \wedge \psi_2) = \mathfrak{R}(\psi_1) \wedge \mathfrak{R}(\psi_2)$.
- $\mathfrak{R}(\psi_1 \vee \psi_2) = \mathfrak{R}(\psi_1) \vee \mathfrak{R}(\psi_2)$.
- $\mathfrak{R}(\langle \pi \rangle (\psi)) = \langle e_{\pi} \rangle \mathfrak{R}(\psi)$.
- $\mathfrak{R}([\pi] \psi) = [\widehat{\pi}_{\langle \rangle_{\phi}}] \mathfrak{R}(\psi)$.

Ⓜ

8.5.3. THEOREM. Given a PDL_{Σ}^+ formula ϕ , ϕ is satisfiable on an AKM $\iff \mathfrak{R}(\phi)$ is satisfiable on a Kripke model w.r.t. the standard PDL_{Σ} semantics.

PROOF

(\implies): Suppose ϕ is satisfiable, then from proposition 8.5.1, we know that ϕ is satisfiable in an AKM model \mathcal{M} that only contains π -transitions for $\pi \in \langle \rangle_{\phi}$. Note that we can also treat \mathcal{M} as a Kripke model over the action set $\Sigma_{\langle \rangle_{\phi}}$, which we denote by \mathcal{G} . Namely, \mathcal{G} is the same as \mathcal{M} except that the transition is renamed. Assuming $\mathcal{M}, s \models \phi$, we now show $\mathcal{G}, s \models \mathfrak{R}(\phi)$ by induction on the structures of $\mathfrak{R}(\phi)$:

- For atomic and boolean cases, trivial.
- Suppose $\phi = \langle \pi \rangle \psi$ thus $\mathfrak{R}(\phi) = \langle e_{\pi} \rangle \mathfrak{R}(\psi)$, where $\pi \in \langle \rangle_{\phi}$. Since $\mathcal{M}, s \models \phi$, there exists some $s \xrightarrow{\pi} s'$ with $\mathcal{M}, s' \models \psi$. According to our construction, in $\mathcal{G}, s \xrightarrow{e_{\pi}} s'$. By the induction hypothesis, $\mathcal{G}, s' \models \mathfrak{R}(\psi)$ in \mathcal{G} . It follows from the standard semantics of PDL_{Σ} that $\mathcal{G}, s \models \mathfrak{R}(\phi)$.

- Suppose $\phi = [\pi]\psi$ thus $\mathfrak{R}(\phi) = [\widehat{\pi}_{\langle \rangle_\phi}]\mathfrak{R}(\psi)$. Since $\mathcal{M}, s \models \phi$, for any sequence of transitions $s \xrightarrow{\pi_1} \dots \xrightarrow{\pi_n} s'$ with $n \geq 0$, $\mathcal{L}(\pi_1 \dots \pi_n) \subseteq \mathcal{L}(\pi)$ implies $\mathcal{M}, s' \models \psi$. Now let us consider any sequence of transitions $s \xrightarrow{e_{\pi'_1}} \dots \xrightarrow{e_{\pi'_m}} s_m$ with $e_{\pi'_1} \dots e_{\pi'_m} \in \mathcal{L}_{\langle \rangle_\phi}(\widehat{\pi}_{\langle \rangle_\phi})$ in \mathcal{G} . It is clear that $\text{exp}_\Sigma(e_{\pi'_1} \dots e_{\pi'_m}) \subseteq \text{exp}_\Sigma(\widehat{\pi}_{\langle \rangle_\phi})$. Since $\widehat{\pi}_{\langle \rangle_\phi}$ is a $\langle \rangle_\phi$ -rewriting of π , we have:

$$\mathcal{L}(\pi'_1 \dots \pi'_m) = \text{exp}_\Sigma(e_{\pi'_1} \dots e_{\pi'_m}) \subseteq \mathcal{L}(\pi)$$

Therefore $\mathcal{M}, s_m \models \psi$. By the induction hypothesis, $\mathcal{G}, s_m \Vdash \mathfrak{R}(\psi)$. It follows that $\mathcal{G}, s \Vdash \phi$.

(\Leftarrow): Suppose $\mathfrak{R}(\phi)$ is satisfiable at a pointed Kripke model (\mathcal{G}, s) over action set $\Sigma_{\langle \rangle_\phi}$ such that $\mathcal{G}, s \Vdash \mathfrak{R}(\phi)$. Clearly, we can construct a corresponding AKM \mathcal{M} which is the same as \mathcal{G} except that for any transition $e_\pi \in \Sigma_{\langle \rangle_\phi}$ in \mathcal{G} , we rename the label e_π by π . We now show $\mathcal{M}, s \models \phi$ by the induction on the structure of ϕ :

- For atomic and boolean cases, trivial.
- Suppose $\phi = \langle \pi \rangle \psi$, where $\pi \in \langle \rangle_\phi$. Since $\mathcal{G}, s \Vdash \mathfrak{R}(\phi)$, namely $\mathcal{G}, s \Vdash \langle e_\pi \rangle \mathfrak{R}(\psi)$, there exists some $s \xrightarrow{e_\pi} s'$ in \mathcal{G} with $s' \Vdash \mathfrak{R}(\psi)$. According to our construction, $s \xrightarrow{\pi} s'$ in \mathcal{M} . By induction hypothesis, $\mathcal{M}, s' \models \psi$. It follows from our semantics that $\mathcal{M}, s \models \phi$.
- Suppose $\phi = [\pi]\psi$: Since $\mathcal{G}, s \Vdash \mathfrak{R}(\phi)$, namely $\mathcal{G}, s \Vdash [\widehat{\pi}_{\langle \rangle_\phi}]\mathfrak{R}(\psi)$, we have $s \xrightarrow{e_{\pi_1}} \dots \xrightarrow{e_{\pi_m}} s'$ and $e_{\pi_1} \dots e_{\pi_m} \in \mathcal{L}_{\langle \rangle_\phi}(\widehat{\pi}_{\langle \rangle_\phi})$ implies $\mathcal{G}, s' \Vdash \mathfrak{R}(\psi)$. Take an arbitrary t such that $s \xrightarrow{\pi'_1} \dots \xrightarrow{\pi'_n} t$ in \mathcal{M} and $\mathcal{L}(\pi'_1 \dots \pi'_n) \subseteq \mathcal{L}(\pi)$. Since $\widehat{\pi}_{\langle \rangle_\phi}$ is the maximal $\langle \rangle_\phi$ -rewriting of π , from Proposition 8.2.4 we have $e_{\pi'_1} \dots e_{\pi'_n} \in \mathcal{L}_{\langle \rangle_\phi}(\widehat{\pi}_{\langle \rangle_\phi})$. Hence $\mathcal{G}, t \Vdash \mathfrak{R}(\psi)$. By the induction hypothesis, $\mathcal{M}, t \models \psi$. Therefore $\mathcal{M}, s \models \phi$.

✕

8.5.4. REMARK. This result is somewhat surprising. Note that our semantics and traditional PDL_Σ semantics differs as shown in the previous section. However, they coincide after the rewriting. For example, $\phi = \langle a \cdot b \rangle p \wedge [a][b]\neg p$ is satisfiable w.r.t our semantics, but not in standard PDL, while $\mathfrak{R}(\phi) = \langle e_{a \cdot b} \rangle p \wedge [\mathbf{0}][\mathbf{0}]\neg p$ is satisfiable in traditional PDL semantics, where constant $\mathbf{0}$ denotes the empty language.

From Theorem 8.5.3, we can reduce satisfiability checking of PDL_Σ over AKM to standard PDL_Σ satisfiability checking, which has been extensively studied in the literature (see, e.g., [HKT00]) and is known to be EXPTIME -complete. Note that the regular expression rewriting can be done in 2-EXPTIME as in Theorem 8.2.6. These entail that the satisfiability checking of PDL_Σ over AKM can be done in 3-EXPTIME ⁵.

⁵The length of the output of a 2-EXPTIME algorithm is essentially at most doubly exponential of the size of the input. A moment of reflection should confirm the desired complexity.

8.6 Conclusion and Future Work

We have performed a thorough study of PDL_Σ over accelerated labelled transition systems. We investigated three problems: model checking, axiomatization and satisfiability checking. We showed that the model checking problem of this logic is EXSPACE -complete while the program complexity turns out to be NLOGSPACE -complete. This answers an open question in [EvdP06]. We also provided a sound and complete axiomatization for PDL_Σ which involves Kleene Algebra as an Oracle. Furthermore, we show the satisfiability problem is decidable in 3-EXPTIME by giving a reduction to the satisfiability of PDL_Σ w.r.t. the standard PDL_Σ semantics on Kripke models.

There are many avenues for future study. First, although we conjecture that our reduction method is optimal, the exact complexity of the satisfiability problem is left open. In [CvdPW08], we claimed the satisfiability problem is EXSPACE -complete. However, the argument was, in retrospect, based on a misunderstanding of Theorem 8.2.6. There are a number of extensions of PDL_Σ (e.g. the test operator) and we are interested in what will happen if the accelerated transitions are labelled by expressions containing extra operators. Furthermore, some open problems remain in applying AKM to abstract model checking of liveness properties, as sketched in [EvdP06]. For instance, how can an abstraction with accelerated transitions be computed automatically? [EvdP06] hints at the relation to automated termination provers. Our study shows that the model checking problem with accelerated transitions is hard. So another interesting question is how to add the minimal number of accelerated transitions, in order to prove a certain liveness property.