Copy and Remove as Dynamic Operators

Carlos Areces[a], Hans van Ditmarsch[b], Raul Fervari[a], Bastien Maubert[c] and François Schwarzentruber[d]

[a]CONICET & FAMAF, Universidad Nacional de Córdoba, Argentina;
[b]Open University, The Netherlands;
[c]Università degli Studi di Napoli "Federico II", Italy;
[d]Univ Rennes, CNRS, IRISA, France

ARTICLE HISTORY
Compiled August 17, 2021

ABSTRACT
In this article we present a modal logic that extends the basic modal logic ML with two dynamic operators: *copy* (cp), which replicates the current model, labelling each copy with a different propositional symbol and respecting accessibility relations even between distinct copies; and *remove* (rm), which deletes paths in the model that satify certain intermediate conditions. We call the resulting logic ML(cp, rm). We study its computational complexity, and its relative expressivity with respect to (static) modal logics ML and ML($\Box^-$), and the dynamic epistemic Action Model Logic, AML.

KEYWORDS
Modal logic, dynamic epistemic logic, complexity, expressivity.

1. Introduction

Modal logic [16, 17] is a family of languages conceived to reason about different modes of truth. The most popular semantics of these logics is usually given in terms of relational models [34], as many modes of truth can be represented in this setting. Some examples are: necessity, obligation, knowledge, belief and temporality, just to name a few. Thus, the evolution or change in those concepts can be seen as transformations in the underlying relational model.

Over the last decades, several approaches appeared in which modalities are interpreted by a transformation of the model. In these logics, the semantics of such *dynamic* modality is given in terms of a binary relation between pointed relational models, where the second argument of the relation is the transformed model. Some examples of this kind of logics are the so-called *sabotage logic* [45], wherein states or arrows are deleted from a model; *dynamic epistemic logics* [50] that focus on such model changing operators in view of modelling change of knowledge or belief (the standard interpretation

C. Areces. Email: carlos.areces@unc.edu.ar
H. van Ditmarsch. Email: hans.van-ditmarsch@loria.fr
R. Fervari. Email: rfervari@unc.edu.ar
B. Maubert. Email: bastien.maubert@gmail.com
F. Schwarzentruber. Email: francois.schwarzentruber@ens-rennes.fr

for the basic modalities in that setting); *graph modifiers* [10], gathering different kinds of modifications on the model; or the family of *memory/hybrid logics* [39, 7], in which the dynamic operators change the valuation of the model. In [2, 3, 23, 4] a new line of contributions to model-update logics, motivated by van Benthem's sabotage logic, is developed. Therein, three kinds of updates on the accessibility relation are considered: deleting an edge (such as in sabotage), adding an edge, and swapping around an edge. These modifications are studied both globally (anywhere in the model) and locally (from the evaluation point), giving rise to the so-called *relation-changing logics*. Our contributions in this article advance this last line of work.

In, e.g., [4], it is pointed out that, when dealing with relation-changing operators, there is a trade-off between expressive power and computational complexity. More precisely, all the concrete logics studied therein are very expressive (for instance, they lack the tree model property and can enforce infinite models). As a downside, the model checking problem for these logics is PSpace-complete, whereas their satisfiability problem is undecidable [6, 5]. Another example of very expressive relation-changing logics with untractable reasoning tasks is the *modal separation logic* family from [19, 20, 21, 22]. Separating connectives are, in essence, relation-changing operators. In the aforementioned works, it is shown that even interpreted over weakly functional models (the most common models in separation logics), these logics are very expressive and their reasoning tasks become untractable very quickly. Over arbitrary models they are easily shown untractable, since the separating conjunction $*$ combined with the emptiness test emp is able to express the undecidable global sabotage operator from [2, 4].

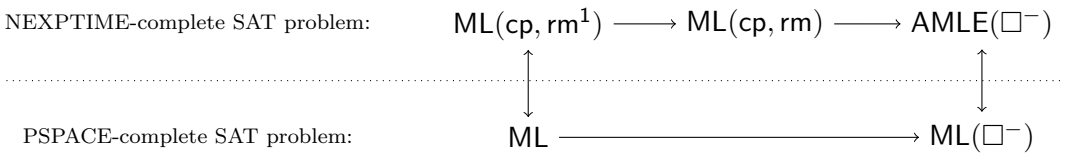
From the literature discussed in the last paragraph, we can extrapolate that reasoning with relation-changing operators is computationally expensive. This is a fair conclusion, since the above-mentioned papers explore thoroughly the properties of a wide range of logics. However, there are more specific instances of relation-changing logics which are more tractable. Arrow update logic [32], a dynamic framework in doxastic logic, is a relation-changing logic that can be embedded, via reduction axioms, into the basic modal logic ML. Even though such embedding might involve an exponential blow-up on the size of the formula, the satisfiability problem can be solved, at worst, in ExpSpace. In [24, 25], a family of relation-changing operators that represent epistemic steps of inference is presented. In particular, the operators model the fact that an agent gains introspective information. These logics are also effectively embedded into decidable logics (e.g., the propositional dynamic logic PDL [30]); therefore their satisfiability problem is decidable. The particularity in both approaches is that they were designed in order to represent very specific phenomena. Also, updates are performed under certain conditions that can be expressed syntactically in the languages, which makes them more controlled. By following these, by no mean exhaustive, examples of updates, in this paper we focus on the design of a dynamic logic with good expressivity and computational properties. Moreover, we will see how different fragments of this logic are related to action model logic [13].

Our Contributions. In this article, we extend the modal logic $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ from [8, 9], which is obtained from ML with the addition of two primitive operators called *copy* and *remove*. The 'remove' primitive is a relation-changing operator that deletes paths in a model, according to intermediate conditions defined syntactically. This operation generalises the 'weaker remove' from [8, 9], which only allows sequential removal, one at a time. Herein, we extend this feature with the simultaneous deletion

according to a set of path descriptions. Moreover, in such descriptions, intermediate tests are formulas from the full language $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$, whereas in [8, 9] the remove operator uses only propositional tests. The 'copy' primitive is a domain, relation and valuation changing operator; it replicates the original model, labels each copy with a new propositional symbol, and adds new edges between the worlds of each copy, emulating the original accessibility relation. We study model theoretic properties of this logic and its complexity.

As an example of what one can do with this logic, we show an embedding of action model logic (AML) into $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$: we show that every action model can be simulated by a combination of the copy and remove operators. This is in line with the previously known result that, on the class of finite models, action model execution corresponds to model restriction ('remove') on a bisimilar copy ('copy') of the initial model [46]. The 'remove' operator we propose is akin to the generalised arrow updates of [33], continuing the work started in [32]. Their approach is also known to have equal expressivity with action model logic. But the copy and remove operators we propose are more procedural (whereas the operators in [33] are more of declarative nature). We also provide a translation from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ to AML extended with the inverse operator $\Box^-$, and where actions in action models have postconditions (also called effects). These results combined give us a way to, from any formula in action model logic, obtain an equivalent formula in action model logics such that action models are in a normal form, in the sense of the action emulation relation of [52]. Moreover, action models in the obtained formula are of size at most four. It is worthwhile to notice that the generalisation of the remove operator to handle simultaneous deletions from a given set of path descriptions, and tests from the full language, is crucial at this point. As we will see later on the paper, both new features are key for establishing a relation between $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ and AML, making a fair expressivity comparison.

Interestingly, $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ falls between the basic modal logic ML and ML extended with $\Box^-$, in terms of expressive power, but it is exponentially more succinct. The picture below illustrates these results.



$\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ is the fragment of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ restricted to deletions of length 1. $\mathsf{AMLE}(\Box^-)$ is the extension of AML with effects and the converse modality $\Box^-$. From left to right, expressivity increases: arrows represent *strictly more expressive than*, two-sided arrows represent *equally expressive as*. From bottom to top, succinctness increases.

A version of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ was first introduced in [8, 9]. In this article, we extend previous results and present some new. Our contributions are summarised below:

- We present the modal logic of copy and remove, denoted $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$. This logic extends the one presented in [8, 9] by allowing: simultaneous deletions; and path expressions defined with test formulas of the full language.
- We introduce the notion of path bisimulation and prove that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ is invariant under such notion.
- We prove that the expressivity of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ lies strictly between the basic modal logic ML and the modal logic with inverse $\mathsf{ML}(\Box^-)$.

- We provide a translation from the action model logic AML into $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$.
- Conversely, $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ formulas can be translated into $\mathsf{AMLE}(\Box^-)$.
- By composing the two aforementioned translations, any AML formula can be translated into an AMLE formula with action models of size at most 4.
- We establish the computational complexity of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ and some of its fragments. We also establish the complexity of $\mathsf{AMLE}(\Box^-)$ and provide a tableaux calculus for this logic.

Outline. In Section 2, we introduce the main definitions to tackle the rest of the paper. Section 3 is devoted to introduce $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$, generalising and reviewing the work in [8, 9]. In Section 4 we introduce AML and intuitively show its relation with $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$. Sections 5 and 6 investigate translations between AML into $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$. Section 7 discusses our complexity results. Finally, Section 8 positions our work with respect to related results, whereas Section 9 provides final remarks.

2. Technical Preliminaries

In this section we introduce central definitions concerning models, bisimulations and the syntax and semantics of the basic modal logic ML and its extension with the converse modality $\Box^-$. In the rest of the paper, let Prop be a countable set of propositional symbols and Mod be a finite set of modal symbols.

We start by introducing the structures in which we interpret formulas of any of the logics we consider.

Definition 2.1 (Relational Models)**.** A *relational model* (or model) is a tuple $\mathcal{M} = \langle W, R, V\rangle$, where: W is a non-empty set of states or worlds; $R \subseteq \mathsf{Mod} \times W \times W$ is the accessibility relation; and $V : \mathsf{Prop} \to 2^W$ is the valuation function. For $R \subseteq \mathsf{Mod} \times W \times W$, we will denote $R_a = \{(w, w') \mid (a, w, w') \in R\}$.

Let $w \in W$; we call $(\mathcal{M}, w)$ a *pointed model*, with parentheses usually dropped.

Syntax of ML and $\mathsf{ML}(\Box^-)$ is defined as follows:

Definition 2.2 (Syntax)**.** The set of formulas of $\mathsf{ML}(\Box^-)$ is defined by the following grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_a\varphi \mid \Box_a^-\varphi,$$

where $p \in \mathsf{Prop}$ and $a \in \mathsf{Mod}$.

Let $\Diamond_a\varphi$ be a shorthand for $\neg\Box_a\neg\varphi$, and $\Diamond_a^-\varphi$ for $\neg\Box_a^-\neg\varphi$. Other Boolean operators are defined in the usual way. ML is the fragment of $\mathsf{ML}(\Box^-)$ where $\Box_a^-$ does not appear. Also, for any formula φ we define $\mathsf{Mod}(\varphi) := \{a \in \mathsf{Mod} \mid \Box_a \text{ or } \Box_a^- \text{ appears in } \varphi\}$. Similarly, $\mathsf{Prop}(\varphi) := \{p \in \mathsf{Prop} \mid p \text{ appears in } \varphi\}$.

We now present the semantics.

Definition 2.3 (Semantics)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ be a relational model with $w \in W$; the satisfaction relation $\models$ between models and $\mathsf{ML}(\Box^-)$ formulas is inductively defined

as follows:

$$\begin{array}{lll} \mathcal{M}, w \models p & \text{iff} & w \in V(p) \\ \mathcal{M}, w \models \neg\varphi & \text{iff} & \mathcal{M}, w \not\models \varphi \\ \mathcal{M}, w \models \varphi \wedge \psi & \text{iff} & \mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, w \models \psi \\ \mathcal{M}, w \models \Box_a \varphi & \text{iff} & \text{for all } v \in W, wR_a v \text{ implies } \mathcal{M}, v \models \varphi \\ \mathcal{M}, w \models \Box_a^- \varphi & \text{iff} & \text{for all } v \in W, vR_a w \text{ implies } \mathcal{M}, v \models \varphi. \end{array}$$

We say that a formula φ is satisfiable if there is a pointed model $\mathcal{M}, w$ such that $\mathcal{M}, w \models \varphi$, and it is valid (notation: $\models \varphi$) if φ holds in every pointed model.

We will be interested in investigating and comparing the expressive power of the different languages we will introduce. Hence, suitable notions of bisimulation are crucial. A bisimulation is a relation between models that share similar structural properties. The main goal is to characterise the exact structural properties that a logic can capture. Now, we introduce the basic notion of bisimulation [41], together with some of its variants (see., e.g. [40]).

Definition 2.4 (Bisimulations)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ and $\mathcal{M}' = \langle W', R', V'\rangle$ be two relational models; a non-empty relation $Z \subseteq W \times W'$ is a *bisimulation* between $\mathcal{M}$ and $\mathcal{M}$ if wZw' implies:

(atom) $w \in V(p)$ if and only if $w' \in V'(p)$, for all $p \in \mathsf{Prop}$;
(zig) if $wR_a v$ then there is $v' \in W'$ such that $w'R'_a v'$ and vZv';
(zag) if $w'R'_a v'$ then there is $v \in W$ such that $wR_a v$ and vZv'.

We write $\mathcal{M}, w \leftrightarroweq \mathcal{M}', w'$ if there is a bisimulation Z between $\mathcal{M}$ and $\mathcal{M}'$ with wZw'.

Definition 2.5 (Two-way bisimulations)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ and $\mathcal{M}' = \langle W', R', V'\rangle$ be two relational models; a non-empty relation $Z \subseteq W \times W'$ is a *two-way bisimulation* between $\mathcal{M}$ and $\mathcal{M}$ if wZw' implies that all conditions in Definition 2.4 hold, and:

(zig$^-$) if $vR_a w$ then there is $v' \in W'$ such that $v'R'_a w'$ and vZv';
(zag$^-$) if $v'R'_a w'$ then there is $v \in W$ such that $vR_a w$ and vZv'.

We write $\mathcal{M}, w \leftrightarroweq^- \mathcal{M}', w'$ if there is a two-way bisimulation Z between $\mathcal{M}$ and $\mathcal{M}'$ with wZw'.

Definition 2.6 (P-bisimulations and P-two-way bisimulations)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ and $\mathcal{M}' = \langle W', R', V'\rangle$ be two relational models, and let $P \subseteq \mathsf{Prop}$; a non-empty relation $Z \subseteq W \times W'$ is a *P-bisimulation* (resp., a *P-two-way bisimulation*) between $\mathcal{M}$ and $\mathcal{M}'$ if Z is as in Definition 2.4 (resp., as in Definition 2.5), with the (atom) condition restricted to propositional symbols in P.

We write $\mathcal{M}, w \leftrightarroweq_P \mathcal{M}', w'$ (resp., $\mathcal{M}, w \leftrightarroweq_P^- \mathcal{M}', w'$) if there is a P-bisimulation (resp., a P-two-way bisimulation) Z between $\mathcal{M}$ and $\mathcal{M}'$ with wZw'.

The following proposition states the correspondence between the different notions of bisimulation and formula equivalence (see e.g., [16, 17, 40]).

Proposition 2.7 (Invariance)**.** *Let $\mathcal{M} = \langle W, R, V\rangle$ and $\mathcal{M}' = \langle W', R', V'\rangle$ be two relational models, with $w \in W$ and $w' \in W'$, and let $P \subseteq \mathsf{Prop}$; then,*

- *$\mathcal{M}, w \leftrightarroweq \mathcal{M}', w'$ implies $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}', w' \models \varphi$, for all $\varphi \in \mathsf{ML}$;*
- *$\mathcal{M}, w \leftrightarroweq^- \mathcal{M}', w'$ implies $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}', w' \models \varphi$, for all $\varphi \in \mathsf{ML}(\Box^-)$;*
- *$\mathcal{M}, w \leftrightarroweq_P \mathcal{M}', w'$ implies $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}', w' \models \varphi$, for all $\varphi \in \mathsf{ML}$ with*

$\mathsf{Prop}(\varphi) \subseteq P$;
- $\mathcal{M}, w \leftrightarrows^{-}_{P} \mathcal{M}', w'$ *implies* $\mathcal{M}, w \models \varphi$ *iff* $\mathcal{M}', w' \models \varphi$, *for all* $\varphi \in \mathsf{ML}(\Box^{-})$ *with* $\mathsf{Prop}(\varphi) \subseteq P$.

We now define how to compare the expressive power of two logics.

Definition 2.8 ($\mathsf{L}_1 \preceq \mathsf{L}_2$)**.** Let L_1 and L_2 be two logical languages. We say that L_1 is at most as expressive as L_2, denoted $\mathsf{L}_1 \preceq \mathsf{L}_2$, if for every L_1 formula φ there exists an L_2 formula φ' such that for every model $\mathcal{M}, w$,

$$\mathcal{M}, w \models \varphi \text{ if and only if } \mathcal{M}, w \models \varphi'.$$

$\mathcal{M}, w$ is seen as a model of L_1 on the left and as a model of L_2 on the right, and we use in each case the appropriate semantic relation. The existence of the formula φ' above will be usually proved by defining a function $t : \mathsf{L}_1 \to \mathsf{L}_2$ such that $t(\varphi) = \varphi'$.

We say that L_1 is strictly less expressive than L_2 (notation $\mathsf{L}_1 \prec \mathsf{L}_2$) if $\mathsf{L}_1 \preceq \mathsf{L}_2$ but $\mathsf{L}_2 \not\preceq \mathsf{L}_1$; and L_1 and L_2 are equally expressive ($\mathsf{L}_1 \approx \mathsf{L}_2$) if $\mathsf{L}_1 \preceq \mathsf{L}_2$ and $\mathsf{L}_2 \preceq \mathsf{L}_1$.

Finally, we define the size of formulas. While this notion is usually defined as the size of a syntactic tree representing the formula, we will use instead a notion of size based on a more compact representation of formulas, namely where identical subformulas are merged. The syntactic tree thus becomes a directed acyclic graph (DAG).

Definition 2.9 (DAG size)**.** The DAG size of a formula φ is the number of distinct subformulas of φ. For $\mathsf{ML}(\Box^{-})$, the set $\mathrm{Sub}(\varphi)$ of subformulas of a formula φ is inductively defined as follows:

$$\begin{aligned}
\mathrm{Sub}(p) &= \{p\} \\
\mathrm{Sub}(\neg\varphi) &= \mathrm{Sub}(\varphi) \cup \{\neg\varphi\} \\
\mathrm{Sub}(\varphi \wedge \psi) &= \mathrm{Sub}(\varphi) \cup \mathrm{Sub}(\psi) \cup \{\varphi \wedge \psi\} \\
\mathrm{Sub}(\Box_a\varphi) &= \mathrm{Sub}(\varphi) \cup \{\Box_a\varphi\} \\
\mathrm{Sub}(\Box^{-}_a\varphi) &= \mathrm{Sub}(\varphi) \cup \{\Box^{-}_a\varphi\}.
\end{aligned}$$

The size of an $\mathsf{ML}(\Box^{-})$ formula φ is then defined as $\|\varphi\| = |\mathrm{Sub}(\varphi)|$.

3. The Modal Logic of Copy and Remove

In this section we introduce $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$, which is the basic modal logic ML extended with two operators, cp and rm. These operators are able to perform changes on the model in which they are evaluated: the cp operator creates copies of the model, whereas the rm operator removes edges in the model. The logic follows the lines of other model-update logics like sabotage [45, 12], graph modifiers [10], arrow updates [32] and relation-changing logics [2, 4]. Moreover, as we will discuss in detail later, it has connections with model updates used in dynamic epistemic logics [50].

3.1. Syntax and Semantics

The syntax of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ *formulas* and of *path expressions* are defined by mutual induction. Path expressions are used in the rm operator to define what edges of a

model should be removed. For defining path expressions we borrow ideas from PDL-programs [30] and path expressions in query languages such as XPath (see e.g., [43]).

Definition 3.1 (Syntax)**.** The set of path expressions and formulas are defined by mutual induction in the non-terminal symbols π and φ, respectively, as follows:

$$\pi ::= a \mid \varphi \mid \pi;\pi,$$
$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_a\varphi \mid \mathsf{rm}(\Pi)\varphi \mid \mathsf{cp}(Q,q)\varphi,$$

where: $p \in \mathsf{Prop}$; $a \in \mathsf{Mod}$; Π is a finite set of path expressions; Q is a finite set of propositional symbols; and $q \in Q$. Other operators are defined as usual. Sometimes we write $\mathsf{rm}(\pi)$ and $\mathsf{cp}(q,q)$, when $\Pi = \{\pi\}$ and $Q = \{q\}$.

The set of all formulas is denoted by $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$. $\mathsf{ML}(\mathsf{cp})$ (resp., $\mathsf{ML}(\mathsf{rm})$) is the fragment of $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ that does not use the remove (resp., copy) operator. $\mathsf{ML}(\mathsf{cp},\mathsf{rm}^1)$, is the fragment of $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ that only involves path expressions of the shape $\varphi_1;a;\varphi_2$, i.e., that only removes paths of length one (see the semantics of the rm operator below). Finally, $\mathsf{ML}(\mathsf{rm}^1)$ is the fragment of $\mathsf{ML}(\mathsf{cp},\mathsf{rm}^1)$ that does not use the copy operator.

Intuitively, a path expression of the form a refers to all a-edges in a given model; one of the form φ refers to all worlds that verify φ, a world being a path of length 0; and an expression of the form $\pi;\pi'$ denotes all the paths which are concatenations of a path denoted by π with a path denoted by π'.

The intuitive meaning of the operation $\mathsf{rm}(\Pi)\varphi$ is that φ holds after having deleted (simultaneously) all edges that appear in paths matching some path expression in Π. The operator $\mathsf{cp}(Q,q)\varphi$ means that after creating one copy of the initial model for each proposition $p \in Q$, φ is true in the copy associated with q.

Below we introduce the notion of path in a model. Intuitively, a path is a sequence of alternations of worlds and modal symbols from Mod. This definition is crucial in the semantics of our remove operator.

Definition 3.2. (Path) Let $\mathcal{M} = \langle W, R, V\rangle$ be a model; a path in $\mathcal{M}$ is a sequence $w_0a_1w_1\ldots a_nw_n \in W \times (\mathsf{Mod} \times W)^*$ such that, for all $0 \leq i < n$, $(w_i, w_{i+1}) \in R_{a_{i+1}}$. Given two paths $\sigma = w_0a_1\ldots a_nw_n$ and $\sigma' = w'_0a'_1\ldots a'_mw'_m$ such that $w_n = w'_0$, we define their concatenation $\sigma;\sigma' = w_0a_1\ldots a_nw_na'_1\ldots a'_mw'_m$. Finally, we define $\mathsf{edges}(\sigma) = \{(a_i, w_{i-1}, w_i) \mid 1 \leq i \leq n\}$, i.e., $\mathsf{edges}(\sigma)$ is the set of edges in σ.

We now define the interpretation of path expressions in a model.

Definition 3.3 (Semantics of path expressions)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ be a model and π a path expression. We define the set of paths $\mathcal{P}^{\mathcal{M}}(\pi)$ by induction on π as follows:

$$\begin{array}{lcl}
\mathcal{P}^{\mathcal{M}}(a) & = & \{wau \mid (w,u) \in R_a\} \\
\mathcal{P}^{\mathcal{M}}(\varphi) & = & \{w \mid \mathcal{M}, w \models \varphi\} \\
\mathcal{P}^{\mathcal{M}}(\pi;\pi') & = & \{\sigma;\sigma' \mid \sigma \in \mathcal{P}^{\mathcal{M}}(\pi), \sigma' \in \mathcal{P}^{\mathcal{M}}(\pi') \text{ and } \sigma;\sigma' \text{ is defined}\}.
\end{array}$$

Given a set Π of path expressions, $\mathcal{P}^{\mathcal{M}}(\Pi)$ is defined as follows:

$$\mathcal{P}^{\mathcal{M}}(\Pi) = \bigcup_{\pi\in\Pi} \mathcal{P}^{\mathcal{M}}(\pi).$$

We define how the two dynamic operators cp and rm modify a model.

Definition 3.4 (Updated Models)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ be a model and $Q \subseteq \mathsf{Prop}$; we define the updated model $\mathcal{M}_{\mathsf{cp}(Q)}$ as:

$$\begin{array}{l}\mathcal{M}_{\mathsf{cp}(Q)} = \langle W_{\mathsf{cp}(Q)}, R_{\mathsf{cp}(Q)}, V_{\mathsf{cp}(Q)}\rangle, \text{ where}\\ \quad W_{\mathsf{cp}(Q)} \;=\; W \times Q\\ \quad R_{\mathsf{cp}(Q)} \;=\; \{(a,(w,q),(w',q')) \mid (a,w,w') \in R\}\\ \quad V_{\mathsf{cp}(Q)}(p) \;=\; \{(w,q) \mid w \in V(p)\} \;\text{ for }\; p \in \mathsf{Prop} \setminus Q\\ \quad V_{\mathsf{cp}(Q)}(q) \;=\; \{(w,q) \mid w \in W\} \;\text{ for }\; q \in Q.\end{array}$$

Let Π be a set of path expressions, we define the updated model $\mathcal{M}_{\mathsf{rm}(\Pi)}$ as:

$$\begin{array}{l}\mathcal{M}_{\mathsf{rm}(\Pi)} = \langle W, R_{\mathsf{rm}(\Pi)}, V\rangle, \text{ where}\\ \quad R_{\mathsf{rm}(\Pi)} \;=\; R \setminus \bigcup_{\sigma \in \mathcal{P}^{\mathcal{M}}(\Pi)} \mathsf{edges}(\sigma).\end{array}$$

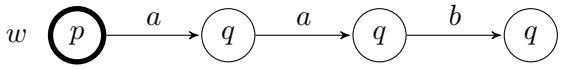
We can now define the semantics of $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$.

Definition 3.5 (Semantics)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ be a relational model, and $w \in W$; the satisfiability relation $\models$ between models and $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ formulas is inductively defined as follows. We only provide the semantics for the new operators, the others remain as before.

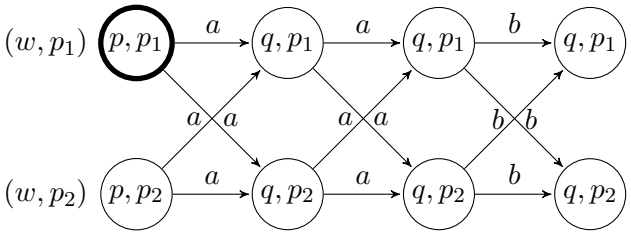
$$\begin{array}{lll}\mathcal{M}, w \models \mathsf{rm}(\Pi)\varphi & \text{iff} & \mathcal{M}_{\mathsf{rm}(\Pi)}, w \models \varphi\\ \mathcal{M}, w \models \mathsf{cp}(Q,q)\varphi & \text{iff} & \mathcal{M}_{\mathsf{cp}(Q)}, (w,q) \models \varphi.\end{array}$$

We discuss some examples to illustrate the behaviour of the new operators.

Example 3.6. Let us consider the following model $\mathcal{M}$:



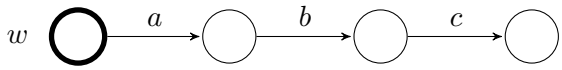
Evaluating the formula $\mathsf{cp}(\{p_1,p_2\},p_1)\Diamond_a\Diamond_a\Diamond_b\top$, we first get the following model, in which we have two copies of the original one: one satisfying p_1 and one satisfying p_2:



Next we have to evaluate $\Diamond_a\Diamond_a\Diamond_b\top$ at (w,p_1), since the copy labelled by p_1 is the one designated as the 'current copy'. Notice that this formula is satisfied at w in the original model. Notice also that $\mathsf{cp}(\{p_1,p_2\},p_1)$ does not introduce new information about successors: each copy of a successor in the original model is also a successor in the new model. It means that modal information remains unchanged, hence $\Diamond_a\Diamond_a\Diamond_b\top$ holds at (w,p_1) in the new model.

Now, let us see some properties of rm. The example shows the non-commutativity of the operator. Moreover, we illustrate the difference between sequential and simultaneous deletion.

Example 3.7. Let us consider the following model $\mathcal{M}$:



Changing the order in which deletions are performed changes the effects of the deletions. Consider $\mathsf{rm}(b;c)\mathsf{rm}(a;b)\Diamond_a\top$. After evaluating $\mathsf{rm}(b;c)$, we get the model

When we evaluate $\mathsf{rm}(a;b)$, no deletions are done (the current model has no a-edges followed by b-edges), then $\Diamond_a\top$ holds at w. On the other hand, if we consider $\mathsf{rm}(a;b)\mathsf{rm}(b;c)\Diamond_a\top$, after evaluating $\mathsf{rm}(a;b)$ we get:

The operation $\mathsf{rm}(b;c)$ does not remove edges, and the formula $\Diamond_a\top$ does not hold at w. Finally, with the formula $\mathsf{rm}(\{a;b,b;c\})$, we remove all the edges in $\mathcal{M}$. It is clear that the effect of sequential deletions with a single path expression inside of modalities can be captured by the logics from [8, 9]. However, the simultaneous deletion w.r.t. a set of path expressions cannot be captured in that setting. Thus, the logic in this paper is more expressive than the one in [8, 9].

To define the DAG size of a $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ formula, in addition to the number of different subformulas, we also have to take into account the size of sets Q in copy operators, and the length of path expressions in remove operators.

Definition 3.8 (DAG size in $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$)**.** For $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$, $\mathrm{Sub}(\varphi)$ is inductively defined as for $\mathsf{ML}(\Box^-)$ together with the following clauses:

$$\mathrm{Sub}(\mathsf{rm}(\Pi)\varphi) = \mathrm{Sub}(\varphi) \cup \bigcup_{\pi\in\Pi}\bigcup_{\varphi'\in\pi} \mathrm{Sub}(\varphi') \cup \{\mathsf{rm}(\Pi)\varphi\}$$
$$\mathrm{Sub}(\mathsf{cp}(Q,q)\varphi) = \mathrm{Sub}(\varphi) \cup \{\mathsf{cp}(Q,q)\varphi\}.$$

The length $|\pi|$ of a path expression reflects the length of paths it denotes, and is defined as the number of modal symbols that appear in it:

$$|a| = 1$$
$$|\varphi| = 0$$
$$|\pi;\pi'| = |\pi| + |\pi'|.$$

The size of an $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ formula φ is then defined as $\|\varphi\| = |\mathrm{Sub}(\varphi)| + \max_{\pi\in\varphi}|\pi|$.

3.2. Bisimulations and Expressive Power

We start with a lemma that states that a model resulting from $\mathsf{cp}(Q)$ is bisimilar to the original model if we ignore propositional symbols in Q.

Lemma 3.9. *Let $\mathcal{M},w$ be a pointed model, $Q \subseteq \mathsf{Prop}$ and $P = \mathsf{Prop} \setminus Q$. It holds that, for every $q \in Q$, $\mathcal{M},w \leftrightarrows_P \mathcal{M}_{\mathsf{cp}(Q)},(w,q)$.*

Proof. The lemma follows from Definition 3.4. Define the relation $Z = \{(w,(w,q)) \mid w \in W \text{ and } q \in Q\}$. We prove that Z is a P-bisimulation. Let $(w,(w,q)) \in Z$.

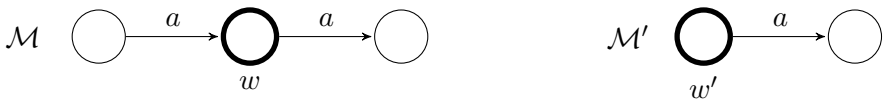
(P-atom) For $p \in P$, $w \in V(p)$ if, and only if, $(w,q) \in V_{\mathsf{cp}(Q)}(p)$.
(zig) If there is w' such that wR_aw', then $(w,q)R_a(w',q)$, and $w'Z(w',q)$.
(zag) If there is (w',q') such that $(w,q)R_{\mathsf{cp}(Q)_a}(w',q')$, then wR_aw', and $w'Z(w',q')$.

This proves the lemma. □

The expressive power of several modal logics with model-update operators has been extensively studied in [2, 3, 23, 4]. In general, adding relation-changing operators to ML increases the expressive power.

Proposition 3.10. $\mathsf{ML} \prec \mathsf{ML}(\mathsf{rm}) \preceq \mathsf{ML}(\mathsf{cp},\mathsf{rm})$.

Proof. Since ML is a fragment of ML(rm) it is trivial than $\mathsf{ML} \preceq \mathsf{ML}(\mathsf{rm})$ (also that $\mathsf{ML}(\mathsf{rm}) \preceq \mathsf{ML}(\mathsf{cp},\mathsf{rm})$). To prove that it is strictly less expressive, consider the following models.



According to Definition 2.4, $\mathcal{M}, w \leftrightarrows \mathcal{M}', w'$, and thus no formula of ML can distinguish the two. However, $\mathcal{M}, w \not\models \mathsf{rm}(a;a)\Diamond_a\top$ while $\mathcal{M}', w' \models \mathsf{rm}(a;a)\Diamond_a\top$. □

The question of whether $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ is strictly more expressive than $\mathsf{ML}(\mathsf{rm})$ is still unanswered. We conjecture that this is actually the case.

As our goal is to show bisimulation invariance for our logic, we need to introduce a suitable notion of bisimulation for $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$, which we name *path-bisimulations*.

Definition 3.11 (Path bisimulations)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ and $\mathcal{M}' = \langle W', R', V'\rangle$ be two models; a non-empty relation $Z \subseteq (W \times 2^{\mathsf{Mod}\times W\times W}) \times (W' \times 2^{\mathsf{Mod}\times W'\times W'})$ is a path bisimulation between $\mathcal{M}$ and $\mathcal{M}'$, if $(w,S)Z(w',S')$ implies:

(atom) $w \in V(p)$ if and only if $w' \in V'(p)$, for all $p \in \mathsf{Prop}$;
(zig) if wS_av then there is $v' \in W'$ such that $w'S'_av'$ and $(v,S)Z(v',S')$;
(zag) if $w'S'_av'$ then there is $v \in W$ such that wS_av and $(v,S)Z(v',S')$;
(path) for all $\Pi \subseteq \mathsf{Path}$, $(w,T)Z(w',T')$
where $T = S \setminus \bigcup_{\sigma\in\mathcal{P}^{\langle W,S,V\rangle}(\Pi)} \mathsf{edges}(\sigma)$ and $T' = S \setminus \bigcup_{\sigma\in\mathcal{P}^{\langle W',S',V'\rangle}(\Pi)} \mathsf{edges}(\sigma)$.

Notice how, in the (path) condition, once (w,S) and (w',S') are related by Z, w and w' should also enter the bisimulation with any possible update of S and S' by an arbitrary path deletion on the models where the accessibility relations are S and S'.

Given $\mathcal{M} = \langle W, R, V\rangle$ and $\mathcal{M}' = \langle W', R', V'\rangle$, we write $\mathcal{M}, w \leftrightarrows^{path} \mathcal{M}', w'$ if there is a path bisimulation Z between $\mathcal{M}$ and $\mathcal{M}'$ where $(w,R)Z(w',R')$.

The following lemma states that cp preserves path bisimulations.

Lemma 3.12. *Let $\mathcal{M} = \langle W, R, V\rangle$ and $\mathcal{M}' = \langle W', R', V'\rangle$ be two models, $w \in W$ and $w' \in W'$. Then, $\mathcal{M}, w \leftrightarrows^{path} \mathcal{M}', w'$ implies $\mathcal{M}_{\mathsf{cp}(Q)}, (w,q) \leftrightarrows^{path} \mathcal{M}'_{\mathsf{cp}(Q)}, (w',q)$.*

Proof. We have to define a bisimulation Z between $\mathcal{M}_{\mathsf{cp}(Q)}$ and $\mathcal{M}'_{\mathsf{cp}(Q)}$. Because we

have $\mathcal{M}, w \leftrightarrows^{path} \mathcal{M}', w'$, there exists Z such that $(w, R)Z(w', R')$. We define:

$$Z' = \{(((v, q), S_{\mathsf{cp}(Q)}), ((v', q), S'_{\mathsf{cp}(Q)})) \mid (v, S)Z(v', S')\}.$$

Take (v, q), (v', q), S and S' such that $((v, q), S_{\mathsf{cp}(Q)})Z'((v', q), S'_{\mathsf{cp}(Q)})$. By definition, $(v, S)Z(v', S')$. The condition (atom) holds for Z' because it holds for Z, and (v, q) and (v', q) are both labelled by the symbol q. For (zig), assume $((v, q), S_{\mathsf{cp}(Q)})Z'((v', q), S'_{\mathsf{cp}(Q)})$ and $((v, q), (u, r)) \in (S_{\mathsf{cp}(Q)})_a$. Then we know $(v, u) \in S_a$. Because $(v, S)Z(v', S')$, by (zig) there is some u' such that $(v', u') \in S'_a$. Hence, we have $((v', q), (u', r)) \in (S'_{\mathsf{cp}(Q)})_a$ such that $((u, q), S_{\mathsf{cp}(Q)})Z'((u', q), S'_{\mathsf{cp}(Q)})$. The proof for (zag) is analogous to the one for (zig). For (path), notice that for all $\Pi \subseteq \mathsf{Form}$, $(v, S \setminus \bigcup_{\sigma \in \mathcal{P}^{\langle W,S,V\rangle}(\Pi)} \mathsf{edges}(\sigma))Z(v', S' \setminus \bigcup_{\sigma \in \mathcal{P}^{\langle W',S',V'\rangle}(\Pi)} \mathsf{edges}(\sigma))$. For $S_{\mathsf{cp}(Q)}$ and $S'_{\mathsf{cp}(Q)}$, we need to consider also path expressions π involving propositional symbols in Q. It is easy to see that by definition of Z', we also have $((v, q), S \setminus \bigcup_{\sigma \in \mathcal{P}^{\langle W,S,V\rangle}(\Pi)} \mathsf{edges}(\sigma))Z'((v', q), S' \setminus \bigcup_{\sigma \in \mathcal{P}^{\langle W',S',V'\rangle}(\Pi)} \mathsf{edges}(\sigma))$.

Therefore, we can conclude that $\mathcal{M}_{\mathsf{cp}(Q)}, (w, q) \leftrightarrows^{path} \mathcal{M}'_{\mathsf{cp}(Q)}, (w', q)$. □

Now, the intended property:

Proposition 3.13 ($\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ invariance for $\leftrightarrows^{path}$)**.** *Let $\mathcal{M}, w$ and $\mathcal{M}', v$ be two pointed models. Then, $\mathcal{M}, w \leftrightarrows^{path} \mathcal{M}', v$ implies that, for all $\varphi \in \mathsf{ML}(\mathsf{cp}, \mathsf{rm})$, $\mathcal{M}, w \models \varphi$ if and only if $\mathcal{M}', v \models \varphi$.*

Proof. The inductive hypothesis needs to be generalised as follows. Let $\mathcal{M} = \langle W, R, V\rangle$ and $\mathcal{M}' = \langle W', R', V'\rangle$ be two relational models, with $w \in W$, $w' \in W'$. Let $Q_1, \ldots, Q_n$ be any enumeration of finite subsets of Prop. For $i < n$, we define W_i inductively as: $W_0 := W$ and $W_{i+1} := W_i \times Q_{i+1}$ (W'_i is defined analogously). For simplicity, sometimes for $((w, p_1), \ldots, p_i) \in W_i$, we will write $wp_1, \ldots, p_i$. The valuation V_i is defined as follows (V'_i is analogous):

$$V_i(p) := \{w\alpha \mid \alpha \in (Q_1 \times \ldots \times Q_i) \text{ and } (w \in V(p) \text{ or } p \text{ appears in } \alpha)\}.$$

We will show that for all $i \in \mathbb{N}$, $S_i \subseteq W_i^2$, $S'_i \subseteq W_i'^2$ and $\varphi \in \mathsf{ML}(\mathsf{cp}, \mathsf{rm})$, if $\langle W_i, S_i, V_i\rangle, w_i \leftrightarrows^{path} \langle W'_i, S'_i, V'_i, \rangle, w'_i$ then

$$\langle W_i, S_i, V_i\rangle, w_i \models \varphi \text{ if and only if } \langle W'_i, S'_i, V'_i, \rangle, w'_i \models \varphi.$$

The proof is by induction on the structure of φ. The cases for $p \in \mathsf{Prop}$ and Boolean connectives are straightforward. Also, for $\varphi = \Diamond_a \psi$, it is as for ML but using (zig), (zag) and $(S_i)_a$ and $(S'_i)_a$ as accessibility relations. Let us consider cp and rm.

[$\varphi = \mathsf{cp}(Q, q)\psi$]**:** Suppose $\langle W_i, S_i, V_i\rangle, w_i \models \mathsf{cp}(Q_{i+1}, q_{i+1})\psi$. Then, by $\models$ for cp, $\langle W_i, S_i, V_i\rangle_{\mathsf{cp}(Q_{i+1})}, (w_i, q_{i+1}) \models \psi$. Since $\langle W_i, S_i, V_i\rangle, w_i \leftrightarrows^{path}_{P} \langle W'_i, S'_i, V'_i\rangle, w'_i$, by Lemma 3.12, $\langle W_i, S_i, V_i\rangle_{\mathsf{cp}(Q_{i+1})}, (w_i, q_{i+1}) \leftrightarrows^{path} \langle W'_i, S'_i, V'_i\rangle_{\mathsf{cp}(Q_{i+1})}, (w'_i, q_{i+1})$, then by IH we get $\langle W'_i, S'_i, V'_i\rangle_{\mathsf{cp}(Q_{i+1})}, (w'_i, q_{i+1}) \models \psi$. Hence, $\langle W'_i, S'_i, V'_i\rangle, w'_i \models \mathsf{cp}(Q_{i+1}, q_{i+1})\psi$.

[$\varphi = \mathsf{rm}(\Pi)\psi$]**:** Suppose $\langle W_i, S_i, V_i\rangle, w_i \models \mathsf{rm}(\Pi)$. Then, by $\models$ for rm, we have $\langle W_i, S_i \setminus \bigcup_{\sigma \in \mathcal{P}^{\langle W_i,S_i,V_i\rangle}(\Pi)} \mathsf{edges}(\sigma), V_i\rangle, w_i \models \psi$.
By (path), $(w_i, S_i \setminus \bigcup_{\sigma \in \mathcal{P}^{\langle W_i,S_i,V_i\rangle}(\Pi)} \mathsf{edges}(\sigma))Z(w'_i, S_i \setminus \bigcup_{\sigma \in \mathcal{P}^{\langle W'_i,S'_i,V'_i\rangle}(\Pi)} \mathsf{edges}(\sigma))$.

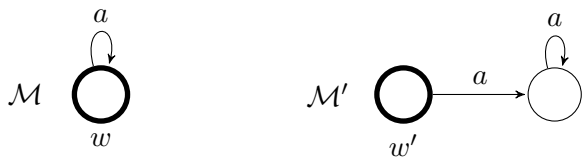
Hence, by IH, $\langle W_i', S_i' \setminus \bigcup_{\sigma \in \mathcal{P}^{\langle W_i', S_i', V_i' \rangle}(\Pi)} \mathsf{edges}(\sigma), V_i' \rangle, w_i' \models \psi$. As a consequence, $\langle W_i', S_i', V_i' \rangle, w_i' \models \mathsf{rm}(\Pi)$.

This concludes the proof. □

The clause (path) in the definition of path bisimulations is 'syntactic', since it explicitly mentions a set of path expressions Π. For model theoretical purposes, a purely semantic condition would be nicer. Moreover, the condition makes it difficult to check path bisimilarity. However, it serves our purpose: to characterise the relative expressive power of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ with respect to ML and $\mathsf{ML}(\Box^-)$. In order to show $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \prec \mathsf{ML}(\Box^-)$, we need to show that that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \preceq \mathsf{ML}(\Box^-)$ and that $\mathsf{ML}(\Box^-) \not\preceq \mathsf{ML}(\mathsf{cp}, \mathsf{rm})$. To show that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \preceq \mathsf{ML}(\Box^-)$ we need to provide an equivalence preserving translation from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ into $\mathsf{ML}(\Box^-)$. This translation exists, and it will be proved later on the paper. At this stage we only show that $\mathsf{ML}(\Box^-) \not\preceq \mathsf{ML}(\mathsf{cp}, \mathsf{rm})$.

Proposition 3.14. $\mathsf{ML}(\Box^-) \not\preceq \mathsf{ML}(\mathsf{cp}, \mathsf{rm})$.

Proof. In order to show the proposition, we need to show that there are two path bisimilar models that can be distinguished in $\mathsf{ML}(\Box^-)$. Consider the following models:



To check that $\mathcal{M}, w \leftrightarrows^{path} \mathcal{M}', w'$, notice that any path on one model can be mimicked in the other. Removing paths of any size, leads to the empty relation in both models. Thus, (zig) and (zag) are satisfied both before and after every possible removal. But the models are distinguishable in $\mathsf{ML}(\Box^-)$: $\mathcal{M}, w \models \Diamond^- \top$, but $\mathcal{M}', w' \not\models \Diamond^- \top$. □

4. Action Model Logic

Action model logic (AML) [13] is a well-known dynamic epistemic logic, originally designed to model information updates. It is an extension of the basic epistemic logic with a dynamic operator $\langle \mathcal{E}, e \rangle$ for the execution of actions. This operator is parameterised by a pointed action (or event) model $(\mathcal{E}, e)$. Formally, an action model is a multi-graph, where nodes can be decorated with formulas that act as pre and post-conditions (the latter are also called *effects*). An action model usually represents a multi-agent information changing scenario, and they are treated as syntactic objects in modal operators.

It is known that this modality can be eliminated into an exponentially larger formula in the basic epistemic language. As a consequence, deciding model checking for AML is PSpace-complete, while deciding satisfiability is NExpTime-complete [11].

4.1. Syntax and Semantics

In this section we present the main ingredients of action model logic. For simplicity, we present definitions of action models and formulas that are mutually dependent,

but formal definitions by double induction can be found in [50, p.149]. Notice that we consider as the base language an extended version of AML, and then we define fragments. The full language is denoted $\mathsf{AMLE}(\Box^-)$.

Definition 4.1 (Action Models)**.** An action model $\mathcal{E}$ is a tuple $\mathcal{E} = \langle E, \to, \mathsf{pre}, \mathsf{eff}\rangle$, where: E is a non-empty finite set of action points; $\to$ is a set of relations $\to_a \subseteq E \times E$, with $a \in \mathsf{Mod}$; $\mathsf{pre} : E \to \mathsf{AMLE}(\Box^-)$ is a precondition function; and $\mathsf{eff} : E \to \mathsf{Prop} \to \mathsf{AMLE}(\Box^-)$ is an effect function (also known as the postcondition function), such that for every $e \in E$, $\mathsf{eff}(e)$ is the identity function on all but a finite set of propositional symbols, that we write $\mathsf{dom}(\mathsf{eff}(e))$.

If $\mathcal{E} = \langle E, \to, \mathsf{pre}, \mathsf{eff}\rangle$ is an action model and $F \subseteq E$ a subset of action points, $(\mathcal{E}, F)$ is a multi-pointed action model. When $F = \{e\}$ is a singleton we simply write $(\mathcal{E}, e)$ and say pointed action model. We will also often drop parentheses. If for every $e \in \mathcal{E}$ $\mathsf{eff}(e)$ is the identity function, then $\mathcal{E}$ is called a purely informative action model. This means that it does not change the worlds in which it occurs, but only modifies the accessibility relations. We may therefore omit the effect function in the description of such action models.

Finally, for an action model $\mathcal{E} = \langle E, \to, \mathsf{pre}, \mathsf{eff}\rangle$ we define its set of modal and propositional symbols:

$$\mathsf{Mod}(\mathcal{E}) = \bigcup_{\to_a \in \to} \{a\} \quad \mathsf{Prop}(\mathcal{E}) = \bigcup_{e \in E} \{\mathsf{Prop}(\mathsf{pre}(e))\} \cup \bigcup_{e \in E, p \in \mathsf{Prop}} \{\mathsf{Prop}(\mathsf{eff}(e)(p))\},$$

and let its size be the number of action points: $|\mathcal{E}| = |E|$.

We can now define the language of action model logic.

Definition 4.2 (Syntax)**.** The set of formulas of action model logic is defined by the following grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_a \varphi \mid \Box_a^- \varphi \mid [\mathcal{E}, F]\varphi,$$

where $p \in \mathsf{Prop}$, $a \in \mathsf{Mod}$ and $(\mathcal{E}, F)$ is a multi-pointed action model.

As usual, $\langle\alpha\rangle\varphi$ is a shorthand for $\neg[\alpha]\neg\varphi$, $\Diamond_a^-\varphi$ for $\neg\Box_a^-\neg\varphi$, and other operators are defined in the usual way.

The set of formulas generated by the non-terminal φ above is denoted by $\mathsf{AMLE}(\Box^-)$. Notice, in particular, that there is no restriction on how complex action models can be in an expression for the form $[\mathcal{E}, F]\varphi$.

We are also interested in the following fragments. AMLE is the fragment of $\mathsf{AMLE}(\Box^-)$ without the $\Box^-$ modality. $\mathsf{AML}(\Box^-)$ (resp., AML) is the fragment of $\mathsf{AMLE}(\Box^-)$ (resp., AMLE) where all action models appearing in modalities of the shape $[\mathcal{E}, F]$ are purely informative.

The definitions of $\mathsf{Mod}(\varphi)$ and $\mathsf{Prop}(\varphi)$ for φ an $\mathsf{AMLE}(\Box^-)$ formula are as in Definition 2.2, except that $\mathsf{Mod}([\mathcal{E}, F]\varphi) = \mathsf{Mod}(\mathcal{E}) \cup \mathsf{Mod}(\varphi)$ and $\mathsf{Prop}([\mathcal{E}, F]\varphi) = \mathsf{Prop}(\mathcal{E}) \cup \mathsf{Prop}(\varphi)$.

We now define three products between models. The first one represents the update of a relational model with an action model.

Definition 4.3 (Update product)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ be a relational model, and let $\mathcal{E} = \langle E, \to, \mathsf{pre}, \mathsf{eff}\rangle$ be an action model; we define their update product as $\mathcal{M} \otimes \mathcal{E} =$

$\langle W^{\otimes}, R^{\otimes}, V^{\otimes}\rangle$, where:

$$\begin{array}{lcl} W^{\otimes} & = & \{(w,e) \in W \times E \mid \mathcal{M}, w \models \mathsf{pre}(e)\} \\ R_a^{\otimes} & = & \{((w,e),(v,f)) \mid (w,v) \in R_a \text{ and } e \to_a f\} \\ V^{\otimes}(p) & = & \{(w,e) \mid \mathcal{M}, w \models \mathsf{eff}(e)(p)\}. \end{array}$$

The second product represents the sequential composition of two action models.

Definition 4.4 (Sequential composition product)**.** The sequential composition product of two action models $\mathcal{E} = \langle E, \to, \mathsf{pre}, \mathsf{eff}\rangle$ and $\mathcal{E}' = \langle E', \to', \mathsf{pre}', \mathsf{eff}'\rangle$ is defined as the action model $\mathcal{E} \times \mathcal{E}' = \langle E^{\times}, \to^{\times}, \mathsf{pre}^{\times}, \mathsf{eff}^{\times}\rangle$, where:

$$\begin{array}{lcl} E^{\times} & = & E \times E' \\ \to_a^{\times} & = & \{((e,e'),(f,f')) \mid (e,f) \in \to_a \text{ and } (e',f') \in \to_a\} \\ \mathsf{pre}^{\times}(e,e') & = & \mathsf{pre}(e) \wedge [\mathcal{E},e]\mathsf{pre}'(e') \\ \mathsf{eff}^{\times}(e,e')(p) & = & \begin{cases} \mathsf{eff}(e)(p) & \text{if } p \notin \mathsf{dom}(\mathsf{eff}'(e')), \\ [\mathcal{E},e]\mathsf{eff}'(e')(p) & \text{otherwise.} \end{cases} \end{array}$$

The third product represents parallel composition of purely informative events and will be used in Section 6.

Definition 4.5 (Parallel composition product)**.** The parallel composition product of two purely informative action models $\mathcal{E} = \langle E, \to, \mathsf{pre}\rangle$ and $\mathcal{E}' = \langle E', \to', \mathsf{pre}'\rangle$ is defined as the action model $\mathcal{E} \| \mathcal{E}' = \langle E^{\|}, \to^{\|}, \mathsf{pre}^{\|}, \mathsf{eff}^{\|}\rangle$, where:

$$\begin{array}{lcl} E^{\|} & = & E \times E' \\ \to_a^{\|} & = & \{((e,e'),(f,f')) \mid (e,f) \in \to_a \text{ and } (e',f') \in \to_a\} \\ \mathsf{pre}^{\|}(e,e') & = & \mathsf{pre}(e) \wedge \mathsf{pre}'(e'). \end{array}$$

Note that the parallel composition of events with effects is more delicate to define, as the postconditions of two concurrent events may be in conflict on the value to assign to certain propositions. This question has been investigated in e.g., [49, 53, 38].

We now introduce the semantics of action model logic.

Definition 4.6 (Semantics)**.** Let $\mathcal{M} = \langle W, R, V\rangle$ be a relational model with $w \in W$, and let $(\mathcal{E}, F)$ be a multi-pointed action model with precondition function pre; the satisfiability relation is defined as in Definition 3.5 for Boolean operators and $\Box_a$, and:

$$\mathcal{M}, w \models [\mathcal{E}, F]\varphi \quad \text{iff} \quad \text{for all } e \in F,\ \mathcal{M}, w \models \mathsf{pre}(e) \text{ implies } (\mathcal{M} \otimes \mathcal{E}), (w,e) \models \varphi.$$

The following example illustrates the effects of performing an update on a relational model via action models.

Example 4.7. In Figure 1 we show an epistemic model (a relational model), an action model, and the result of executing that action model in that epistemic model.

Recall that in epistemic logic, labels in the edges (i.e., members of Mod) are interpreted as agents, and edges state the uncertainty of an agent (see, e.g., [50] for details). In the figure, the epistemic model represents that agents a and b are uncertain whether an atomic proposition p is true (and that they have common knowledge of that uncertainty). The actual world, or designated state, of the model is where p is true (shown

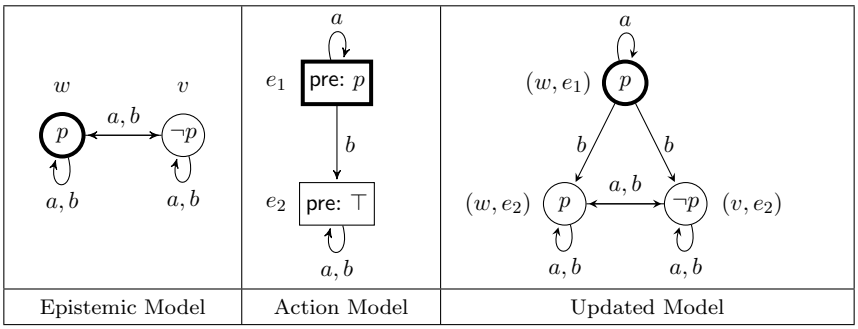


Figure 1. Agent a privately learns that p.

with a thick circle in the figure). The action model represents that agent a learns that p is true, whereas agent b (incorrectly) believes that nothing happens—of which a is aware. In short: a privately learns that p. In action models, nodes are labelled with preconditions, in this case p and $\top$.

The action model in this example does not have effects. As defined before, action models update relational models by mean of a restricted modal product, where the domain is limited to the state-action pairs where the preconditions of the actions hold. Hence, there are only three (and not four) pairs in the updated model: the pair (v, e_1) is missing as the precondition of e_1, the formula p, is not true in state v. The arrows in the restricted product are updated according to the principle that there is a (labelled) arrow between two state-action pairs if there was such an arrow linking both the first arguments and the second arguments, on each respective model. One can now establish that in the resulting model a knows that p (there is only an a-arrow from w to itself), whereas b still consider possible that a, b are ignorant about p.

Performing updates with action models is intuitive when we deal with epistemic scenarios. An action model represents some epistemic information to be incorporated in the underlying relational model via a product with such action model. However, if we are interested in making more arbitrary updates, or if we want to understand, step by step, a transformation of a model into another one, we might want to use more basic operators. In this direction, [26] presents a language with basic actions to represent action models updates. As we will discuss later, $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ turns out to be an alternative to this kind of proposals.

By means of the copy and remove actions $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ can describe the effect of the action model in Figure 1. This is in Figure 2, from left to right. First, we replicate the original epistemic model (the leftmost model in the figure) as many times as there are actions in the action model (in this case, twice). We identify each copy with a (fresh) propositional variable corresponding to an action in the action model (e.g., p_{e_1} corresponds to e_1). Thus we obtain the second model in Figure 2.

Then, we first remove all the edges that point to state-action alternatives wherein the action cannot be executed in the state. This is depicted in the third model of Figure 2. Finally, between the remaining state-action pairs we remove all edges that are ruled out according to the accessibility relation in the action model. Thus we obtain the 'updated model' on the right in Figure 2, which is bisimilar to the one in Figure 1.

Before moving on, we define the DAG size of $\mathsf{AMLE}(\Box^-)$ formulas. With respect to $\mathsf{ML}(\Box^-)$, the main addition is that we have to take into account the size of the event

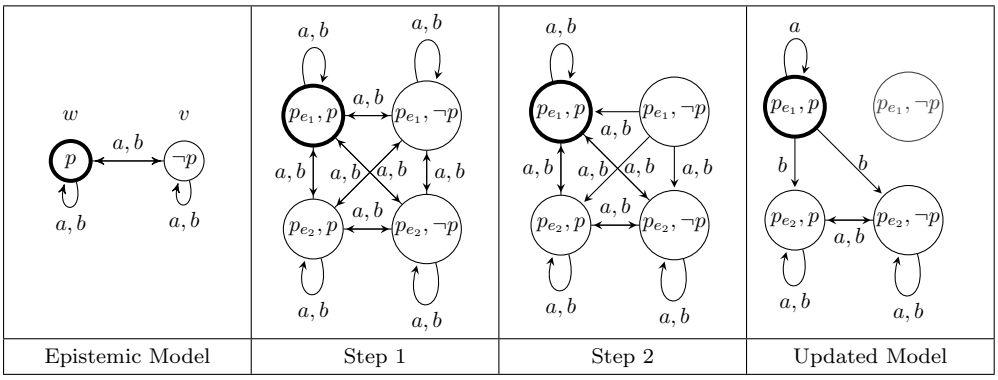


Figure 2. Step by step transformation using copy and remove.

models that appear in a formula, and recursively the size of the preconditions and postconditions that appear in these models.

Definition 4.8. For $\mathsf{AMLE}(\Box^-)$, the set $\mathrm{Sub}(\varphi)$ of subformulas of a formula φ is inductively defined as for $\mathsf{ML}(\Box^-)$ together with the following clause:

$$\mathrm{Sub}([\mathcal{E},F]\varphi) = \mathrm{Sub}(\varphi) \cup \bigcup_{e\in\mathcal{E},p\in\mathsf{Prop}} \mathrm{Sub}(\mathsf{pre}(e)) \cup \mathrm{Sub}(\mathsf{eff}(e)(p)) \cup \{[\mathcal{E},F]\varphi\}$$

The *DAG size* of φ is defined as $|\mathrm{Sub}(\varphi)| + \sum_{\mathcal{E}\in\varphi} |\mathcal{E}|$, where $\mathcal{E} \in \varphi$ means that $\mathcal{E}$ appears in a dynamic modality in one of the subformulas of φ as defined above, i.e., including recursively all pre and postconditions.

4.2. Expressive Power and Completeness

AMLE can be translated into ML (see, e.g., [50]), and it is easy to see that $\mathsf{AMLE}(\Box^-)$ can be translated into $\mathsf{ML}(\Box^-)$. This is achieved via reduction axioms as follows. But first, we introduce a lemma regarding the composition product, whose proof is routine.

Lemma 4.9 ([48])**.** *For every pair of pointed action models $(\mathcal{E},e)$, $(\mathcal{E}',e')$ with postconditions, it holds that: $\models [\mathcal{E},e][\mathcal{E}',e']\varphi \leftrightarrow [(\mathcal{E}\times\mathcal{E}'),(e,e')]\varphi$.*

The following are reduction axioms from $\mathsf{AMLE}(\Box^-)$ to $\mathsf{ML}(\Box^-)$.

Proposition 4.10 (Completeness)**.** *Let $\mathcal{A}$ be any sound and complete proof system for $\mathsf{ML}(\Box^-)$ (see e.g., [16]). A complete proof system for $\mathsf{AMLE}(\Box^-)$ is obtained by adding to $\mathcal{A}$ the following reduction axioms allowing the elimination of $[\mathcal{E},F]$ modalities:*

$$\begin{array}{lcl}
[\mathcal{E},F]\varphi & \leftrightarrow & \bigwedge_{e\in F}[\mathcal{E},e]\varphi \\
[\mathcal{E},e]p & \leftrightarrow & (\mathsf{pre}(e) \to \mathsf{eff}(e)(p)) \\
[\mathcal{E},e]\neg\varphi & \leftrightarrow & (\mathsf{pre}(e) \to \neg[\mathcal{E},e]\varphi) \\
[\mathcal{E},e](\varphi\wedge\psi) & \leftrightarrow & ([\mathcal{E},e]\varphi \wedge [\mathcal{E},e]\psi) \\
[\mathcal{E},e]\Box_a\varphi & \leftrightarrow & (\mathsf{pre}(e) \to \bigwedge\{\Box_a[\mathcal{E},f]\varphi \mid e \to_a f\}) \\
[\mathcal{E},e]\Box_a^-\varphi & \leftrightarrow & (\mathsf{pre}(e) \to \bigwedge\{\Box_a^-[\mathcal{E},f]\varphi \mid f \to_a e\}) \\
[\mathcal{E},e][\mathcal{E}',e']\varphi & \leftrightarrow & [(\mathcal{E}\times\mathcal{E}'),(e,e')]\varphi.
\end{array}$$

Proof. We only need to show that the reduction axioms eliminating $[\mathcal{E}, F]$ and $[\mathcal{E}, e]$ modalities are valid. The axiom for $[\mathcal{E}, F]$ is clearly valid by definition of the semantics (notice that since action models are finite, the conjunction in the axiom is also finite). For $[\mathcal{E}, e]$, we only need to treat the case for $\Box^-$, since the rest are the classical reduction axioms for AMLE. In particular, the case $[\mathcal{E}, e][\mathcal{E}', e']\varphi$ follows by Lemma 4.9.

Suppose that $\mathcal{M}, w \models [\mathcal{E}, e]\Box_a^- \varphi$. If $\mathcal{M}, w \not\models \mathsf{pre}(e)$, thus $\mathcal{M}, w \models [\mathcal{E}, e]\Box_a^- \varphi$ trivially. On the other hand, if $\mathcal{M}, w \models \mathsf{pre}(e)$, then we have (by $\models$) $(\mathcal{M} \otimes \mathcal{E}), (w, e) \models \Box_a^- \varphi$. As a consequence, for all f and v such that $e \to_a f$ and $(v, w) \in R_a$, $(\mathcal{M} \otimes \mathcal{E}), (v, f) \models \varphi$. Again by $\models$, we get $\mathcal{M}, v \models [\mathcal{E}, f]\varphi$. Moreover, notice that this holds for all v such that $(v, w) \in R_a$, hence $\mathcal{M}, w \models \Box_a[\mathcal{E}, f]\varphi$, and since it holds for all f such that $e \to_a f$, therefore $\mathcal{M}, w \models \bigwedge\{\Box_a[\mathcal{E}, f]\varphi \mid e \to_a f\}$.

The right-to-left direction follows similar steps. □

As a corollary, $\mathsf{AMLE}(\Box^-)$ and $\mathsf{ML}(\Box^-)$ are equally expressive. In the next proposition, we will prove the result stating that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ is strictly less expressive than $\mathsf{ML}(\Box^-)$. Notice that the proof relies on Theorem 6.14, which will be proved in Section 6. However, we introduce the proposition here in order to complete the picture about expressive power of the languages.

Proposition 4.11. $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \prec \mathsf{ML}(\Box^-)$.

Proof. In order to show that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \preceq \mathsf{ML}(\Box^-)$ we need to provide an equivalence preserving translation from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ into $\mathsf{ML}(\Box^-)$. This translation exists, and it can be proved in two steps. In Section 6 we will introduce Tr_3, a translation from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ into the logic $\mathsf{AMLE}(\Box^-)$, which establishes that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \preceq \mathsf{AMLE}(\Box^-)$ (see Theorem 6.14, page 26). Also, by Proposition 4.10, there exists a translation from $\mathsf{AMLE}(\Box^-)$ into $\mathsf{ML}(\Box^-)$, thus $\mathsf{AMLE}(\Box^-) \preceq \mathsf{ML}(\Box^-)$. By composing both translations we obtain $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \preceq \mathsf{ML}(\Box^-)$. From that, together with Proposition 3.14 that states $\mathsf{ML}(\Box^-) \not\preceq \mathsf{ML}(\mathsf{cp}, \mathsf{rm})$, we obtain the required $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \prec \mathsf{ML}(\Box^-)$. □

Notice that by Proposition 4.11 we can conclude that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ is strictly less expressive than $\mathsf{AMLE}(\Box^-)$.

Corollary 4.12. $\mathsf{AMLE}(\Box^-) \approx \mathsf{ML}(\Box^-)$ *and* $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \prec \mathsf{AMLE}(\Box^-)$.

We conclude this section by stating an invariance (under bisimulation) result for AMLE and $\mathsf{AMLE}(\Box^-)$. Since $\mathsf{ML}(\Box^-)$ is invariant for two-way bisimulations (Proposition 2.7), and by Proposition 4.10 above, the following proposition holds:

Proposition 4.13 (AML invariance for $\leftrightarrows$; $\mathsf{AML}(\Box^-)$ invariance for $\leftrightarrows^-$)**.** *Let $\mathcal{M}, w$ and $\mathcal{M}', w'$ be two pointed models. Then, $\mathcal{M}, w \leftrightarrows \mathcal{M}', w'$ implies that, for all $\varphi \in \mathsf{AMLE}$, $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}', w' \models \varphi$. Also, $\mathcal{M}, w \leftrightarrows^- \mathcal{M}', w'$ implies that, for all $\varphi \in \mathsf{AMLE}(\Box^-)$, $\mathcal{M}, w \models \varphi$ iff $\mathcal{M}', w' \models \varphi$.*

5. From AML to $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$

In [8, 9], we studied the relations between action models and the copy and remove primitives. In particular, we showed that AML (without effects) can be polynomially translated into $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$, when preconditions in action models are required to be Boolean formulas. In this section, we generalise the translation to preconditions in the full language. This translation is still polynomial for preconditions in ML, but becomes

exponential for arbitrary AML preconditions if we consider the traditional definition of formula size where formulas are represented as trees, because of some subformulas that have to be translated twice. However if we use the DAG size, for which identical subformulas are only counted once, the translation remains polynomial. Note that this translation produces formulas in the fragment $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$, i.e., the ability to remove paths of length one is still enough to simulate AML with arbitrary preconditions.

The translation is a direct adaptation of the one presented in [8, 9]. The main difference is the use of simultaneous deletion instead of sequential deletion, and the translation of preconditions. These modifications do not impact the complexity of the translation if we consider DAG size.

Definition 5.1. Let $\varphi \in \mathsf{AML}$ be such that the domains of action models in φ are disjoint. The translation $\mathsf{Tr}_1 : \mathsf{AML} \to \mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$, is defined as follows:

$$
\begin{array}{lcl}
\mathsf{Tr}_1(p) & = & p \\
\mathsf{Tr}_1(\neg\varphi) & = & \neg\mathsf{Tr}_1(\varphi) \\
\mathsf{Tr}_1(\Box_a\varphi) & = & \Box_a\mathsf{Tr}_1(\varphi) \\
\mathsf{Tr}_1(\varphi \wedge \psi) & = & \mathsf{Tr}_1(\varphi) \wedge \mathsf{Tr}_1(\psi) \\
\mathsf{Tr}_1([\mathcal{E}, F]\varphi) & = & \bigwedge\limits_{e \in F} \mathsf{Tr}_1([\mathcal{E}, e]\varphi) \\
\mathsf{Tr}_1([\mathcal{E}, e]\varphi) & = & \mathsf{Tr}_1(\mathsf{pre}(e)) \to \mathsf{cp}(\{p_{e_1}, \ldots, p_{e_n}\}, p_e)\mathsf{rm}(\mathrm{P})\mathsf{rm}(\Sigma)\mathsf{Tr}_1(\varphi),
\end{array}
$$

where $\mathcal{E} = \langle E, \to, \mathsf{pre}\rangle$ is an action model with $E = \{e_1, \ldots, e_n\}$, $p_{e_1}, \ldots, p_{e_n}$ are fresh distinct atomic propositions, and

$$
\begin{array}{ll}
\mathrm{P} & = \{\top; a; (p_{e_i} \wedge \neg\mathsf{Tr}_1(\mathsf{pre}(e_i))) \mid e_i \in E, a \in \mathsf{Mod}(\mathcal{E})\ \} \\
\Sigma & = \{p_{e_i}; a; p_{e_j} \mid e_i, e_j \in E, a \in \mathsf{Mod}(\mathcal{E}), e_i \not\to_a e_j\}.
\end{array}
$$

The first five cases are trivial, by definition of the semantics. Let us analyse the case of $[\mathcal{E}, e]$. The antecedent $\mathsf{pre}(e)$ reflects the first part of the semantics of $[\mathcal{E}, e]\varphi$. For the consequent, observe that by evaluating a formula $[\mathcal{E}, e]\varphi$ in a relational model $\langle W, R, V\rangle$, we need to simulate the product $\mathcal{M} \otimes \mathcal{E}$. For each action $e_i \in E$, we use a fresh propositional symbol p_{e_i} (which is possible as Prop is infinite). The operation $\mathsf{cp}(\{p_{e_1}, \ldots, p_{e_n}\})$ replicates the original model n times, where n is the number of actions in E. Each copy is 'marked' with some atomic proposition p_{e_i}. We call 'p_{e_i}-states' those states of the form (w, p_{e_i}), where w is a state from the original model. This operation simulates the cartesian product $W \times E$. However, there are pairs in $W \times E$ that are not in $\mathcal{M} \otimes \mathcal{E}$. It is not possible to remove them, but we can make them unreachable from the evaluation point. In order to do so, we use the operation $\mathsf{rm}(\mathrm{P})$. The set of path expressions P characterises all edges that point to p_{e_i}-states which do not satisfy $\mathsf{pre}(e_i)$; therefore, $\mathsf{rm}(\mathrm{P})$ removes all edges pointing to those states, making them unreachable. Still, by the use of the copy operator, we have too many edges that need to be removed in order to obtain the accessibility relation of $\mathcal{M} \otimes \mathcal{E}$. This is performed by the $\mathsf{rm}(\Sigma)$ operator. Remember that $((w, e_i), (v, e_j)) \in R_a^{\otimes}$ in $\mathcal{M} \otimes \mathcal{E}$ if, and only if, $(w, v) \in R_a$ and $e_i \to_a e_j$. By the semantics of the cp operator, two states (w, p_{e_i}) and (v, p_{e_j}) of the 'cartesian product' obtained after $\mathsf{cp}(\{p_{e_1}, \ldots, p_{e_n}\})$ are a-related if, and only if, w and v are related in the original model. Thus, it only remains to delete, for all $a \in \mathsf{Mod}$, all the a-edges of the form $((w, p_{e_i}), (v, p_{e_j}))$ such that there is no a-edge from e_i to e_j in $\mathcal{E}$. This is what $\mathsf{rm}(\Sigma)$ achieves. In this way we obtain a model which is bisimilar to $\mathcal{M} \otimes \mathcal{E}$.

Now, we proceed with the formal proof of correctness of the translation.

Theorem 5.2. *For every* AML *formula* φ *and every pointed model* $\mathcal{M}, w$

$$\mathcal{M}, w \models \varphi \text{ iff } \mathcal{M}, w \models \mathsf{Tr}_1(\varphi).$$

Moreover, $\mathsf{Tr}_1(\varphi)$ *is polynomial in the DAG size of* φ*.*

Proof. We will need the following claim.

Claim 1. Let $\mathcal{M} = \langle W, R, V\rangle$ be a model, $w \in W$, let $\mathcal{E} = \langle E, \to, \mathsf{pre}\rangle$ be a purely informative action model with $E = \{e_1, \ldots, e_n\}$, and where for all $e_i \in E$, $\mathsf{pre}(e_i) \in \mathsf{ML}$. Let $Q = \{p_{e_1}, \ldots, p_{e_n}\} \subset \mathsf{Prop}$ be a set of n distinct proposition symbols not occurring in the preconditions of any $e_i \in E$, and let $\mathcal{M}' = ((\mathcal{M}_{\mathsf{cp}(Q)})_{\mathsf{rm}(\mathrm{P})})_{\mathsf{rm}(\Sigma)}$, with P and Σ as in Definition 5.1. Finally, assume that $\mathcal{M}, w \models \mathsf{pre}(e_1)$. It holds that: $\mathcal{M} \otimes \mathcal{E}, (w, e_1) \leftrightarrows_P \mathcal{M}', (w, p_{e_1})$, where $P = \mathsf{Prop} \setminus Q$.

We prove the claim and Theorem 5.2 by mutual induction, starting with the claim.

Proof of the Claim. Let $\mathcal{M} \otimes \mathcal{E} = \langle W^{\otimes}, R^{\otimes}, V^{\otimes}\rangle$ be the update product of $\mathcal{M}$ and $\mathcal{E}$. Also, let:

- $\mathcal{M}_{\mathsf{cp}(Q)} = \langle W^1, R^1, V^1\rangle$,
- $(\mathcal{M}_{\mathsf{cp}(Q)})_{\mathsf{rm}(\mathrm{P})} = \langle W^2, R^2, V^2\rangle$ and
- $\mathcal{M}' = ((\mathcal{M}_{\mathsf{cp}(Q)})_{\mathsf{rm}(\mathrm{P})})_{\mathsf{rm}(\Sigma)} = \langle W', R', V'\rangle$.

Define the relation $Z = \{((w, e_i), (w, p_{e_i})) \mid (w, e_i) \in W^{\otimes} \text{ and } (w, p_{e_i}) \in W'\}$. We prove that Z is a P-bisimulation, for $P = \mathsf{Prop} \setminus Q$. Let $((w, e_i), (w, p_{e_i})) \in Z$.

- **(P-atom)**. Let $p \in P = \mathsf{Prop} \setminus Q$. By definition of $\otimes$, we have $(w, e_i) \in V^{\otimes}(p)$ if, and only if, $w \in V(p)$. Now, by Definition 3.4 and because $p \in \mathsf{Prop} \setminus Q$, $(w, p_{e_i}) \in V^1(p)$ if and only if $w \in V(p)$. Because the remove operator does not change valuations (see Definition 3.4 again), we obtain that $(w, p_{e_i}) \in V'(p)$ iff $(w, p_{e_i}) \in V^2(p)$ iff $(w, p_{e_i}) \in V^1(p)$ iff $w \in V(p)$, which we have seen holds if and only if $(w, e_i) \in V^{\otimes}(p)$.
- **(zig)**. Suppose there is (v, e_j) such that $(w, e_i) R_a^{\otimes} (v, e_j)$. By definition of $\otimes$, we have that $(w, v) \in R_a$ and $e_i \to_a e_j$, and because $(v, e_j) Z (v, p_{e_j})$ by definition of Z, it is enough to prove that $(w, p_{e_i}) R'_a (v, p_{e_j})$. By Definition 3.4, since $(w, v) \in R_a$ we have that $(w, p_{e_i}) R^1_a (v, p_{e_j})$. We need to check that this edge is not removed by $\mathsf{rm}(\mathrm{P})$ nor by $\mathsf{rm}(\Sigma)$. The set of path expressions P describes edges whose end state satisfy some p_{e_k} but not $\mathsf{Tr}_1(\mathsf{pre}(e_k))$. By assumption, $(w, e_i) R_a^{\otimes} (v, e_j)$, so that $(v, e_j) \in W^{\otimes}$, and thus $\mathcal{M}, v \models \mathsf{pre}(e_j)$ (by definition of the update product). By Lemma 3.9, $\mathcal{M}, v \leftrightarrows_P \mathcal{M}_{\mathsf{cp}(Q)}, (v, p_{e_j})$; because AML is invariant under bisimulation, and by assumption all atomic propositions occurring in $\mathsf{pre}(e_j)$ are in P, it follows that $\mathcal{M}_{\mathsf{cp}(Q)}, (v, p_{e_j}) \models \mathsf{pre}(e_j)$. By hypothesis of mutual induction, the translation is correct on $\mathsf{pre}(e_j)$, hence $\mathcal{M}_{\mathsf{cp}(Q)}, (v, p_{e_j}) \models \mathsf{Tr}_1(\mathsf{pre}(e_j))$. Therefore, $\mathsf{rm}(\mathrm{P})$ does not remove the a-edge between (w, p_{e_i}) and (v, p_{e_j}). On the other hand, we have that $e_i \to_a e_j$, while Σ describes a-edges of the form $((w, p_{e_i}), (v, p_{e_j}))$ such that $e_i \not\to_a e_j$. Thus, $\mathsf{rm}(\Sigma)$ does not remove the edge either. Then we conclude that $(w, p_{e_i}) R'_a (v, p_{e_j})$.
- **(zag)**. Suppose there is (v, p_{e_j}) such that $(w, p_{e_i}) R'_a (v, p_{e_j})$. Because rm only removes edges, we have that $R'_a \subseteq R^2_a \subseteq R^1_a$, and therefore $(w, p_{e_i}) R^1_a (v, p_{e_j})$. By definition of cp, this implies that $w R_a v$. Now, because $(w, p_{e_i}) R^2_a (v, p_{e_j})$, this edge is not removed by $\mathsf{rm}(\mathrm{P})$, which implies that $\mathcal{M}_{\mathsf{cp}(Q)}, (v, p_{e_j}) \models \mathsf{Tr}_1(\mathsf{pre}(e_j))$, and

thus $\mathcal{M}_{\mathsf{cp}(Q)}, (v, p_{e_j}) \models \mathsf{pre}(e_j)$ (by the assumption of the mutual induction that the translation is correct on $\mathsf{pre}(e_j)$). Again, by Lemma 3.9 we have that $\mathcal{M}, v \leftrightarrows_P \mathcal{M}_{\mathsf{cp}(Q)}, (v, p_{e_j})$, and because all proposition symbols occurring in $\mathsf{pre}(e_j)$ are in P, we have that $\mathcal{M}, v \models \mathsf{pre}(e_j)$, and therefore $(v, e_j) \in W^{\otimes}$. Similarly, because $(w, p_{e_i}) R'_a (v, p_{e_j})$ this edge is not removed by $\mathsf{rm}(\Sigma)$, implying that $e_i \to_a e_j$. We have that $(v, e_j) \in W^{\otimes}$, wR_av and $e_i \to_a e_j$. Therefore, $(w, e_i) R^{\otimes}_a (v, e_j)$. We conclude by noting that by definition of Z, $(v, e_j) Z (v, p_{e_j})$.

□

We can now prove Theorem 5.2, by induction on φ. All cases are trivial, except for $\varphi = [\mathcal{E}, e]\varphi'$, which we treat now.

Let $(\mathcal{M}, w)$ be a pointed model, let $\mathcal{E} = \langle E, \to, \mathsf{pre}\rangle$ be an action model, let $e \in E$ and let φ' be an AML formula with preconditions in ML. First, we have $\mathsf{Tr}_1([\mathcal{E}, e]\varphi') = \mathsf{pre}(e) \to \mathsf{cp}(\{p_{e_1}, \ldots, p_{e_n}\}, p_e)\mathsf{rm}(\mathrm{P})\mathsf{rm}(\Sigma)\mathsf{Tr}_1(\varphi')$. Then, let us set $\mathcal{M}' = ((\mathcal{M}_{\mathsf{cp}(p_{e_1}, \ldots p_{e_n})})_{\mathsf{rm}(\mathrm{P})})_{\mathsf{rm}(\Sigma)}$, and let $P = \mathsf{Prop} \setminus \{p_{e_1}, \ldots, p_{e_n}\}$. By the above claim we have that, if $\mathcal{M}, w \models \mathsf{pre}(e)$, $\mathcal{M} \otimes \mathcal{E}, (w, e) \leftrightarrows_P \mathcal{M}', (w, p_e)$. Because $p_{e_1}, \ldots, p_{e_n}$ do not occur in φ', we thus have that

$$\mathcal{M} \otimes \mathcal{E}, (w, e) \models \varphi' \text{ if, and only if, } \mathcal{M}', (w, p_e) \models \varphi'. \tag{1}$$

We need to prove that $\mathcal{M}, w \models [\mathcal{E}, e]\varphi'$ iff $\mathcal{M}, w \models \mathsf{Tr}_1([\mathcal{E}, e]\varphi')$. For the left-to-right implication, assume that $\mathcal{M}, w \models [\mathcal{E}, e]\varphi'$. We separate two cases. Either $\mathcal{M}, w \not\models \mathsf{pre}(e)$, or $\mathcal{M}, w \models \mathsf{pre}(e)$ and $(\mathcal{M} \otimes \mathcal{E}), (w, e) \models \varphi'$. In the first case, we directly obtain that $\mathcal{M}, w \models \mathsf{pre}(e) \to \mathsf{cp}(\{p_{e_1}, \ldots, p_{e_n}\}, p_e)\mathsf{rm}(\mathrm{P})\mathsf{rm}(\Sigma)\mathsf{Tr}_1(\varphi')$, i.e., $\mathcal{M}, w \models \mathsf{Tr}_1([\mathcal{E}, e]\varphi')$. In the second case, by (1) we get that $\mathcal{M}', (w, p_e) \models \varphi'$. Then, by induction hypothesis we obtain that $\mathcal{M}', (w, p_e) \models \mathsf{Tr}_1(\varphi')$, and by definition of Tr_1 it follows that $\mathcal{M}, w \models \mathsf{Tr}_1([\mathcal{E}, e]\varphi')$.

For the right-to-left implication, assume $\mathcal{M}, w \models \mathsf{Tr}_1([\mathcal{E}, e]\varphi')$. Again, we split between two cases: either $\mathcal{M}, w \not\models \mathsf{pre}(e)$, in which case it follows that $\mathcal{M}, w \models [\mathcal{E}, e]\varphi'$, or $\mathcal{M}, w \models \mathsf{pre}(e)$ and $\mathcal{M}, w \models \mathsf{cp}(\{p_{e_1}, \ldots, p_{e_n}\}, p_e)\mathsf{rm}(\mathrm{P})\mathsf{rm}(\Sigma)\mathsf{Tr}_1(\varphi')$. This implies that $\mathcal{M}', (w, p_e) \models \mathsf{Tr}_1(\varphi')$. By induction hypothesis $\mathcal{M}', (w, p_e) \models \varphi'$, and by (1) it follows that $\mathcal{M} \otimes \mathcal{E}, (w, e) \models \varphi'$ (recall that we have assumed that $\mathcal{M}, w \models \mathsf{pre}(e)$, and thus $(w, e) \in W^{\otimes}$). Therefore, $\mathcal{M}, w \models [\mathcal{E}, e]\varphi'$.

Concerning the size of $\mathsf{Tr}_1(\varphi)$, because the precondition of event e is translated twice when translating $\mathsf{Tr}_1([\mathcal{E}, e]\varphi)$ (once in the antecedent of the implication and once in P) this translation is exponential for the usual tree size of formulas. However it remains polynomial in DAG size. □

We see the encoding above applied to a concrete update in Figure 2, obtaining a model which is bisimilar to the updated model in Figure 1.

6. From ML(cp, rm) to AMLE($\Box^-$)

In this section we first provide a translation from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ to AMLE which, combined with the one from AML to $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ described in the previous section, will give rise to a normal form for AML formulas. Then we consider paths of arbitrary length. In this case AML seems not to be enough to simulate the remove operator. Instead, we provide a translation from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ to $\mathsf{AMLE}(\Box^-)$. This entails the decidability of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$.

6.1. Translating from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ *to* AMLE

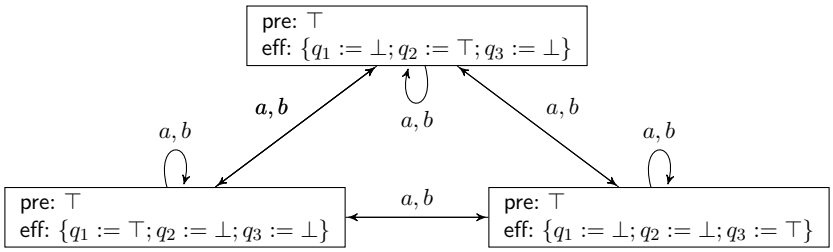
We first show that copy operations can be simulated with action models in AMLE.

Definition 6.1. Given a finite set $Q \subset \mathsf{Prop}$, we define the action model $\mathcal{E}_{\mathsf{cp}}(Q) = \langle Q, \to, \mathsf{pre}, \mathsf{eff}\rangle$, where $\to_a = Q \times Q$ (for all $a \in \mathsf{Mod}$), and for all $q \in Q$:

$$\mathsf{pre}(q) = \top \quad \text{and} \quad \mathsf{eff}(q)(p) = \begin{cases} \top & \text{if } p \in Q \text{ and } p = q \\ \bot & \text{if } p \in Q \text{ and } p \neq q \\ p & \text{if } p \in \mathsf{Prop} \setminus Q. \end{cases}$$

Observe that the only proposition in Q that holds after the occurrence of an event $q \in Q$, is q; the truth values of all atomic propositions not in Q are unaffected.

Example 6.2. The action model $\mathcal{E}_{\mathsf{cp}}(q_1, q_2, q_3)$ for two agents a and b is depicted below (we do not represent the postconditions for propositions not in $\{q_1, q_2, q_2\}$):



Lemma 6.3. *Let $Q \subset \mathsf{Prop}$ be a finite set of atomic propositions. For every pointed model $(\mathcal{M}, w)$, the models $\mathcal{M}_{\mathsf{cp}(Q)}, (w, q)$ and $\mathcal{M} \otimes \mathcal{E}_{\mathsf{cp}}(Q), (w, q)$ are isomorphic.*

Proof. The mapping $f : (u, p) \mapsto (u, p)$ is an isomorphism between $\mathcal{M}_{\mathsf{cp}(Q)}$ and $\mathcal{M} \otimes \mathcal{E}_{\mathsf{cp}}(Q)$. □

We now show that remove operations with paths of length one can also be simulated by application of action models in AMLE.

Definition 6.4. Let $\pi = \varphi; a; \psi$ be a path expression of length one, and assume that the translation Tr_2 from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ to AML is defined on φ and ψ. We define the action model $\mathcal{E}_{\mathsf{rm}}(\pi) = \langle E, \to, \mathsf{pre}\rangle$, where

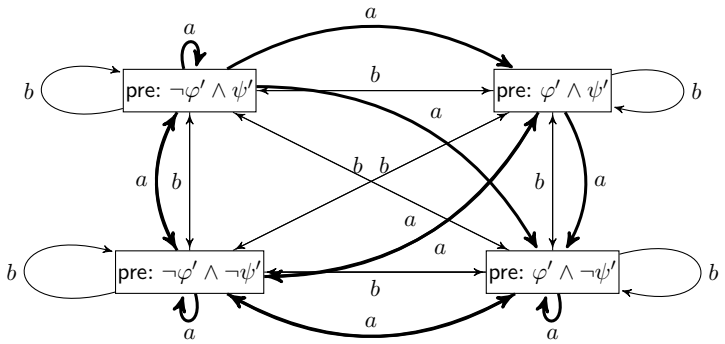
$$\begin{array}{lll} E & = \{00, 10, 01, 11\} & \mathsf{pre}(00) = \neg\mathsf{Tr}_2(\varphi) \wedge \neg\mathsf{Tr}_2(\psi), \\ \to_a & = (E \times E) \backslash \{(10, 01), (10, 11), (11, 01), (11, 11)\} & \mathsf{pre}(10) = \mathsf{Tr}_2(\varphi) \wedge \neg\mathsf{Tr}_2(\psi), \\ \to_b & = (E \times E) \quad \text{for all } b \neq a & \mathsf{pre}(01) = \neg\mathsf{Tr}_2(\varphi) \wedge \mathsf{Tr}_2(\psi), \\ & & \mathsf{pre}(11) = \mathsf{Tr}_2(\varphi) \wedge \mathsf{Tr}_2(\psi). \end{array}$$

If Π is a finite set of path expressions of length one, $\mathcal{E}_{\mathsf{rm}}(\Pi)$ is the parallel composition of all the $\mathcal{E}_{\mathsf{rm}}(\pi)$, for $\pi \in \Pi$, where the precondition of a conjunction of actions simply is the conjunction of the preconditions of its elements; this product represents the *simultaneous* occurrence of several action models. The idea of the definition of the parallel composition comes from [1], [15] and [38]. On the other hand, the composition of action models represents the *sequential* application of action models. Fittingly, the action model $\mathcal{E}_{\mathsf{rm}}(\pi)$ does not have postconditions.

Intuitively, when applied to some model, the action model $\mathcal{E}_{\mathsf{rm}}(\varphi; a; \psi)$ first marks

each state with the truth values of φ and ψ, and then removes all a-edges (w, v) such that φ holds in w and ψ holds in v.

Example 6.5. Suppose that we have two agents a and b. Then $\mathcal{E}_{\mathsf{rm}}(\varphi; a; \psi)$ is depicted below, where $\varphi' = \mathsf{Tr}_2(\varphi)$ and $\psi' = \mathsf{Tr}_2(\psi)$:



Observe that each world u of $\mathcal{M}$ satisfies the precondition of exactly one event e_u in $\mathcal{E}_{\mathsf{rm}}(\Pi)$, which is determined by the truth values in $\mathcal{M}, u$ of the different formulas involved in the path expressions in Π. It is then easy to check that the mapping $f : u \mapsto (u, e_u)$ is an isomorphism, hence:

Lemma 6.6. *If Π is a finite set of path expressions and $(\mathcal{M}, w)$ a pointed model, and if Tr_2 is correct on formulas in Π, then $\mathcal{M}_{\mathsf{rm}(\Pi)}, w$ and $\mathcal{M} \otimes \mathcal{E}_{\mathsf{rm}}(\Pi), (w, e_w)$ are isomorphic.*

Definition 6.7. The translation $\mathsf{Tr}_2 : \mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1) \to \mathsf{AMLE}$ is inductively defined as:

$$
\begin{array}{lcl}
\mathsf{Tr}_2(p) & = & p \\
\mathsf{Tr}_2(\neg\varphi) & = & \neg\mathsf{Tr}_2(\varphi) \\
\mathsf{Tr}_2(\varphi \wedge \psi) & = & \mathsf{Tr}_2(\varphi) \wedge \mathsf{Tr}_2(\psi) \\
\mathsf{Tr}_2(\Box_a\varphi) & = & \Box_a\mathsf{Tr}_2(\varphi) \\
\mathsf{Tr}_2(\mathsf{cp}(Q, q)\varphi) & = & [\mathcal{E}_{\mathsf{cp}}(Q), q]\mathsf{Tr}_2(\varphi) \\
\mathsf{Tr}_2(\mathsf{rm}(\Pi)\varphi) & = & \bigwedge\{[\mathcal{E}_{\mathsf{rm}}(\Pi), e]\mathsf{Tr}_2(\varphi) \mid e \in \mathcal{E}_{\mathsf{rm}}(\Pi)\},
\end{array}
$$

where $\mathcal{E}_{\mathsf{cp}}(Q)$ is given in Definition 6.1 and $\mathcal{E}_{\mathsf{rm}}(\Pi)$ in Definition 6.4.

Theorem 6.8. *For every $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ formula φ and every pointed model $\mathcal{M}, w$*

$$\mathcal{M}, w \models \varphi \text{ iff } \mathcal{M}, w \models \mathsf{Tr}_2(\varphi).$$

Moreover, $\mathsf{Tr}_2(\varphi)$ is polynomial in the DAG size of φ.

Proof. The Boolean and modal cases are trivial. For the copy operator, the result follows from Lemma 6.3, the fact that isomorphism implies bisimulation, and the invariance of AMLE by bisimulation.

Now for the case $\mathsf{rm}(\Pi)\varphi$, take a pointed model $\mathcal{M}, w$. First, observe that, by induction hypothesis, Tr_2 is correct on all formulas in Π. By Lemma 6.6, we have that $\mathcal{M} \otimes \mathcal{E}_{\mathsf{rm}}(\Pi), (w, e_w)$ is isomorphic (and thus bisimilar) to $\mathcal{M}_{\mathsf{rm}(\Pi)}, w$. Because $\mathsf{Tr}_2(\varphi)$

is an AMLE formula, it is invariant by bisimulation, and therefore,

$$\mathcal{M} \otimes \mathcal{E}_{\mathsf{rm}}(\Pi), (w, e_w) \models \mathsf{Tr}_2(\varphi) \text{ iff } \mathcal{M}_{\mathsf{rm}(\Pi)}, w \models \mathsf{Tr}_2(\varphi). \tag{2}$$

By definition, $\mathcal{M} \otimes \mathcal{E}_{\mathsf{rm}}(\Pi), (w, e_w) \models \mathsf{Tr}_2(\varphi)$ iff $\mathcal{M} \models [\mathcal{E}_{\mathsf{rm}}(\Pi), e_w]\mathsf{Tr}_2(\varphi)$. Now, for every $e \in \mathcal{E}_{\mathsf{rm}}(\Pi)$ such that $e \neq e_w$, it holds that $\mathcal{M}, w \not\models \mathsf{pre}(e)$, and therefore $\mathcal{M}, w \models [\mathcal{E}_{\mathsf{rm}}(\Pi), e]\mathsf{Tr}_2(\varphi)$. It follows that $\mathcal{M}, w \models \bigwedge_{e \in \mathcal{E}_{\mathsf{rm}}(\Pi)} [\mathcal{E}_{\mathsf{rm}}(\Pi), e]\mathsf{Tr}_2(\varphi)$ if and only if $\mathcal{M}, w \models [\mathcal{E}_{\mathsf{rm}}(\Pi), e_w]\mathsf{Tr}_2(\varphi)$, and thus

$$\mathcal{M}, w \models \mathsf{Tr}_2(\mathsf{rm}(\Pi)\varphi) \text{ iff } \mathcal{M} \otimes \mathcal{E}_{\mathsf{rm}}(\Pi), (w, e_w) \models \mathsf{Tr}_2(\varphi). \tag{3}$$

By induction hypothesis, $\mathcal{M}_{\mathsf{rm}(\Pi)}, w \models \mathsf{Tr}_2(\varphi)$ iff $\mathcal{M}_{\mathsf{rm}(\Pi)}, w \models \varphi$, i.e.,

$$\mathcal{M}_{\mathsf{rm}(\Pi)}, w \models \mathsf{Tr}_2(\varphi) \text{ iff } \mathcal{M}, w \models \mathsf{rm}(\Pi)\varphi. \tag{4}$$

Combining equations 2, 3 and 4 provides the desired conclusion.

It is easy to verify that the size of $\mathsf{Tr}_2(\varphi)$ is polynomial w.r.t. the DAG size of φ. □

6.2. Translating from ML(cp, rm) *to* AMLE($\Box^-$)

We now show that the full logic with copy and remove with paths of arbitrary length can be translated to AMLE with past modalities (AMLE($\Box^-$)). Together with the result of the previous section, this establishes that the satisfiability problem for the full logic ML(cp, rm) is decidable in ExpSpace. We will refine this bound in the next section.

The translation $\mathsf{Tr}_3 : \mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \to \mathsf{AMLE}(\Box^-)$ is the same as that for $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$, only the action models that simulate remove operations are more intricate.

First, observe that all path expressions are equivalent to an expression of the form $\varphi_0; a_1; \varphi_1; \ldots; a_n; \varphi_n$, by taking the conjunction of all formulas that are not separated by a modal symbol, and inserting a $\top$ between any two consecutive modal symbols. We say that path expressions of this form are in *normal form*, and in this section, we consider that all path expressions are in normal form.

Definition 6.9 (Positional formulas)**.** Let $\pi = \varphi_0; a_1; \varphi_1; \ldots; a_n; \varphi_n$ be a path expression. For every $0 \leq i \leq n$, we define $pos_i^\pi = \langle i \rangle^- \wedge \varphi_i \wedge \langle i \rangle$, where

$$\langle i \rangle^- = \begin{cases} \Diamond^-_{a_i}(\varphi_{i-1} \wedge \langle i-1 \rangle^-) & \text{if } i > 0, \\ \top & \text{otherwise.} \end{cases} \quad \Big| \quad \langle i \rangle = \begin{cases} \Diamond_{a_{i+1}}(\varphi_{i+1} \wedge \langle i+1 \rangle) & \text{if } i < n, \\ \top & \text{otherwise.} \end{cases}$$

Informally, pos_i^π means "the current state is at position i in some path that matches π". For example, for $\pi = \varphi_0; a_1; \varphi_1; \ldots; a_n; \varphi_n$, we have (omitting useless $\top$):

$$\begin{array}{lcl}
pos_0^\pi & = & \varphi_0 \wedge \Diamond_{a_1}(\varphi_1 \wedge \Diamond_{a_2}(\varphi_2 \ \ldots \ \wedge \Diamond_{a_{n-1}}(\varphi_{n-1} \wedge \Diamond_{a_n}\varphi_n)\ldots)) \\
pos_1^\pi & = & \Diamond^-_{a_1}\varphi_0 \wedge \varphi_1 \wedge \Diamond_{a_2}(\varphi_2 \ \ldots \ \wedge \Diamond_{a_{n-1}}(\varphi_{n-1} \wedge \Diamond_{a_n}\varphi_n)\ldots) \\
\ldots & & \\
pos_{n-1}^\pi & = & \Diamond^-_{a_{n-1}}(\ldots\Diamond^-_{a_1}(\varphi_0) \wedge \varphi_1 \ )\wedge \ \varphi_2 \ )\ldots) \wedge \varphi_{n-1} \wedge \Diamond_{a_n}\varphi_n \\
pos_n^\pi & = & \Diamond^-_{a_n}(\ldots\Diamond^-_{a_1}(\varphi_0) \wedge \varphi_1) \wedge \varphi_2) \wedge \ldots \ \varphi_{n-1}) \wedge \varphi_n.
\end{array}$$

The following lemma formalises the intuition about positional formulas.

Lemma 6.10. *Let $\pi{=}\varphi_0; a_1; \varphi_1; \ldots; a_n; \varphi_n$ be a path expression, and let $(\mathcal{M}, w)$ be a pointed model. For every i such that $0 \leq i \leq n$,*

$$\mathcal{M}, w \models pos_i^\pi \text{ iff there exists } P = w_0 a_1 w_1 \ldots a_n w_n \in \mathcal{P}_\pi^\mathcal{M} \text{ s.t. } w_i = w.$$

Proof. The proof is by induction on the length of π:

$\boldsymbol{\pi = \varphi_0}$: Simply observe that $pos_0^{\varphi_0} = \varphi_0$, and $\mathcal{P}_{\varphi_0}^\mathcal{M} = \{v \mid \mathcal{M}, v \models \varphi_0\}$.

$\boldsymbol{\pi = \varphi_0; a_1; \varphi_1; \ldots; a_n; \varphi_n}$: Suppose that $\mathcal{M}, w \models pos_i^\pi$, for some $0 \leq i \leq n$. By definition of pos_i^π, we have $\mathcal{M}, w \models \langle i \rangle^- \wedge \varphi_i \wedge \langle i \rangle$. Now, we know:

1. $\mathcal{M}, w \models \varphi_i$.
2. $\mathcal{M}, w \models \langle i \rangle^-$. By definition of $\langle i \rangle^-$, $\mathcal{M}, w \models \Diamond_{a_i}^-(\varphi_{i-1} \wedge \langle i-1 \rangle^-)$ and thus there is some $v \in W$ such that $(v, w) \in R_{a_i}$ and $\mathcal{M}, v \models \varphi_{i-1} \wedge \langle i-1 \rangle^-$. Let us define $\pi_1 = \varphi_0; a_1; \varphi_1; \ldots; a_{i-1}; \varphi_{i-1}$. By definition, $pos_{i-1}^{\pi_1} = \langle i-1 \rangle^- \wedge \varphi_{i-1}$, hence $\mathcal{M}, v \models pos_{i-1}^{\pi_1}$, and by induction hypothesis, there exists a path $\sigma_1 \in \mathcal{P}_{\pi_1}^\mathcal{M}$ such that $\sigma_1 = w_0 a_1 \ldots a_{i-1} w_{i-1}$ and $w_{i-1} = v$.
3. $\mathcal{M}, w \models \langle i \rangle$. By a symmetric argument, there exists a state $t \in W$ such that $(w, t) \in R_{a_{i+1}}$, and letting $\pi_2 = \varphi_{i+1}; a_{i+2}; \ldots; a_n; \varphi_n$, there exists a path $\sigma_2 \in \mathcal{P}_{\pi_2}^\mathcal{M}$ such that $\sigma_2 = w_{i+1} a_{i+2} \ldots a_n w_n$ and $w_{i+1} = t$.

Notice that $\pi = \pi_1; a_i; \varphi_i; a_{i+1}; \pi_2$. It is now easy to check that $\sigma{=}\sigma_1 a_i w_i a_{i+1} \sigma_2$ is in $\mathcal{P}_\pi^\mathcal{M}$, which concludes the proof.

□

We can now define action models associated to general remove operations. The following definition generalises Definition 6.4.

Definition 6.11. Let $\pi = \varphi_0; a_1; \varphi_1; \ldots; a_n; \varphi_n$ be a path expression. Assume that the translation Tr_3 is defined on φ_i, for $0 \leq i \leq n$. For each $0 \leq i \leq n$ we define the $\mathsf{AMLE}(\Box^-)$ formula $\psi_i^\pi = pos_i^\pi[\varphi_j \mapsto \mathsf{Tr}_3(\varphi_j)]$, i.e. the formula pos_i^π where for every j, φ_j is replaced with its translation $\mathsf{Tr}_3(\varphi_j)$. We define the action model $\mathcal{E}_{\mathsf{rm}}(\pi) = \langle E, \rightarrow, \mathsf{pre} \rangle$, where

- $E = \{0,1\}^{n+1}$;
- for $(b_0 \ldots b_n) \in E$, $\mathsf{pre}(b_0 \ldots b_n) = \bigwedge_{i \mid b_i = 0} \neg \psi_i^\pi \wedge \bigwedge_{i \mid b_i = 1} \psi_i^\pi$;
- for every $a \in \mathsf{Mod}$, $\rightarrow_a = (E \times E) \backslash \{(b_0 \ldots b_n, b'_0 \ldots b'_n) \mid \exists i, 0 \leq i < n, b_i = 1, b'_{i+1} = 1 \text{ and } a_{i+1} = a\}$.

If Π is a finite set of path expressions, $\mathcal{E}_{\mathsf{rm}}(\Pi)$ is the synchronous product of all the $\mathcal{E}_{\mathsf{rm}}(\pi)$, for $\pi \in \Pi$.

First observe that, like in the previous section, given a finite set of path expressions Π and a pointel model $\mathcal{M}, w$, there is a unique action point $e \in \mathcal{E}_{\mathsf{rm}}(\Pi)$ such that $\mathcal{M}, w \models \mathsf{pre}(e)$. This action point is defined by the truth value of each φ_i in each $\pi \in \Pi$ (assuming that Tr_3 is correct on these formula), and we write it e_w. We now establish a lemma which shows that the remove operator $\mathsf{rm}(\Pi)$ and the action model $\mathcal{E}_{\mathsf{rm}}(\Pi)$ essentially have the same effect.

Lemma 6.12. *If Π is a finite set of path expressions, $\mathcal{M}, w$ a pointed model, and if Tr_3 is correct on formulas in Π, then $\mathcal{M}_{\mathsf{rm}(\Pi)}, w$ and $\mathcal{M} \otimes \mathcal{E}_{\mathsf{rm}}(\Pi), (w, e_w)$ are isomorphic.*

Proof. We first consider the case where $\Pi = \{\pi\}$ and $\pi = \varphi_0; a_1; \varphi_1; \ldots; a_n; \varphi_n$.

Let $\mathcal{M} = \langle W, R, V\rangle$. First, recall that $\mathcal{M}_{\mathsf{rm}(\pi)} = \langle W, R_{\mathsf{rm}(\pi)}, V\rangle$, where $R_{\mathsf{rm}(\pi)} = R \setminus \bigcup_{\sigma\in\mathcal{P}^{\mathcal{M}}(\pi)} \mathsf{edges}(\sigma)$. Let us note $W \otimes \mathcal{E}_{\mathsf{rm}}(\pi) = \langle W', R', V'\rangle$.

As already observed, for every $w \in \mathcal{M}$, there is a unique $e_w \in \mathcal{E}_{\mathsf{rm}}(\pi)$ such that $\mathcal{M}, w \models \mathsf{pre}(e_w)$. We thus have $W' = \{(w, e_w) \mid w \in W\}$, and the mapping $f : w \mapsto (w, e_w)$ is therefore a bijection between W and W'. We now show that f is also an isomorphism between the structures $\mathcal{M}_{\mathsf{rm}(\pi)}$ and $\mathcal{M} \otimes \mathcal{E}_{\mathsf{rm}}(\pi)$.

First, concerning valuations, observe that because $\mathcal{E}_{\mathsf{rm}}(\pi)$ has trivial postconditions, we have that for all $(w, e_w) \in W'$ and $p \in \mathsf{Prop}$, $p \in V'(w, e_w)$ iff $p \in V(w)$. Now, because $(w, e_w) = f(w)$, we get that $p \in V(w)$ iff $p \in V'(f(w))$.

Concerning binary relations, we first observe that by Lemma 6.10, we have that for each $0 \leq i \leq n$ and each $w \in W$, $\mathcal{M}, w \models pos_i^\pi$ iff there is a path $\sigma = w_0a_1w_1 \dots a_nw_n \in \mathcal{P}^{\mathcal{M}}(\pi)$ such that $w_i = w$. Like in Definition 6.11, for $0 \leq i \leq n$, we let $\psi_i^\pi = pos_i^\pi[\varphi_j \mapsto \mathsf{Tr}_3(\varphi_j)]$. Because we have assumed that the translation is correct on formulas in π, we have that $\mathcal{M}, w \models pos_i^\pi$ iff $\mathcal{M}, w \models \psi_i^\pi$. It follows that for $e = b_0 \dots b_n \in \mathcal{E}_{\mathsf{rm}}(\pi)$, $\mathcal{M}, w \models \mathsf{pre}(e)$ iff for every $0 \leq i \leq n$ such that $b_i = 1$, there is a path in $\mathcal{P}^{\mathcal{M}}(\pi)$ with w at position i, and for every $0 \leq i \leq n$ such that $b_i = 0$, there is no path in $\mathcal{P}^{\mathcal{M}}(\pi)$ with w at position i. Let $a \in \mathsf{Mod}$, we prove that for every $w, w' \in W$, $(a, w, w') \in R_{\mathsf{rm}(\pi)}$ if, and only if, $(a, f(w), f(w')) \in R'$, which concludes the proof.

For the right-to-left implication, assume that $(a, w, w') \notin R_{\mathsf{rm}(\pi)}$. There are two possibilities. First, $(a, w, w') \notin R$, in which case $(a, f(w), f(w')) \notin R_{\mathsf{rm}(\pi)}$ by definition of the update product (recall that $f(w) = (w, e_w)$ and $f(w') = (w', e_{w'})$), and we are done. Second, $(a, w, w') \in R$ but this edge has been erased in $\mathcal{M}_{\mathsf{rm}(\pi)}$. This implies that there is a path $\sigma = w_0a_1w_1 \dots a_nw_n \in \mathcal{P}^{\mathcal{M}}(\pi)$ and $0 \leq i < n$ such that $w_i = w$, $w_{i+1} = w'$ and $a_{i+1} = a$. Therefore, $\mathcal{M}, w \models \psi_i^\pi$ and $\mathcal{M}, w' \models \psi_{i+1}^\pi$. Noting $e_w = b_0 \dots b_n$ and $e_{w'} = b'_0 \dots b'_n$, we have that $b'_i = 1$ and $b'_{i+1} = 1$, and by definition of $\mathcal{E}(\mathsf{rm}(\Pi))$, $e_w \not\to_a e_{w'}$, and thus $((w, e_w), (w', e_{w'})) \notin R'_a$, i.e. $(a, f(w), f(w')) \notin R'$.

For the left-to-right implication, assume that $(a, f(w), f(w')) \notin R'$, and recall that $f(w) = (w, e_w)$ and $f(w') = (w', e_{w'})$. Again, either $(a, w, w') \notin R$, in which case $(a, w, w') \notin R_{\mathsf{rm}(\pi)}$ and we are done, or $(a, w, w') \in R$ but $e_w \not\to_a e_{w'}$. By definition of $\to_a$, noting $e_w = b_0 \dots b_n$ and $e_{w'} = b'_0 \dots b'_n$, we have that there is $0 \leq i < n$ such that $b_i = 1$, $b'_{i+1} = 1$, and $a_{i+1} = a$. This implies that $\mathcal{M}, w \models \psi_i^\pi$ and $\mathcal{M}, w' \models \psi_{i+1}^\pi$, so there exist paths $\sigma = w_0a_1 \dots a_nw_n$ and $\sigma' = w'_0a_1 \dots a_nw'_n$ in $\mathcal{P}^{\mathcal{M}}(\pi)$ such that $w_i = w$ and $w'_{i+1} = w'$. Consider the path $\sigma'' = w_0a_1 \dots w_ia_{i+1}w'_{i+1} \dots a_{n+1}w'_{n+1}$ that combines σ and σ'. First, this indeed is a path in $\mathcal{M}$ as, by assumption, $(a, w, w') \in R$ and $a = a_{i+1}$. Also, σ'' is clearly a path in $\mathcal{P}^{\mathcal{M}}(\pi)$. Therefore $(a, w, w') \in \mathsf{edges}(\sigma)$ for some $\sigma \in \mathcal{P}^{\mathcal{M}}(\pi)$, and thus, by definition of $R_{\mathsf{rm}(\pi)}$, $(a, w, w') \notin R_{\mathsf{rm}(\pi)}$, which concludes the proof.

We now treat the general case informally, by considering $\Pi = \{\pi, \pi'\}$. Recall that $\mathcal{E}_{\mathsf{rm}}(\Pi)$ is the cartesian product $\mathcal{E}_{\mathsf{rm}}(\pi) \times \mathcal{E}_{\mathsf{rm}}(\pi')$. Given a model $\mathcal{M}$ and two worlds $w, u \in \mathcal{M}$ such that $(w, u) \in R_a$, this edge is removed in $\mathcal{M} \otimes \mathcal{E}_{\mathsf{rm}}(\Pi)$ if, and only if, there is no a-edge between (e_w, e'_w) and (e_u, e'_u). By definition of the cartesian product this is possible if, and only if, there is no a-edge between e_w and e_u in $\mathcal{E}_{\mathsf{rm}}(\pi)$, or there is no a-edge between e'_w and e'_u in $\mathcal{E}_{\mathsf{rm}}(\pi')$ which, according to the definition of $\mathcal{E}_{\mathsf{rm}}(\pi)$ and $\mathcal{E}_{\mathsf{rm}}(\pi')$, means that (a, w, u) is an edge either in a path that matches π or in a path that matches π'. □

Finally, define the translation $\mathsf{Tr}_3 : \mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \to \mathsf{AMLE}(\Box^-)$ as follows.

Definition 6.13. $\mathsf{Tr}_3 : \mathsf{ML}(\mathsf{cp}, \mathsf{rm}) \to \mathsf{AMLE}(\Box^-)$ is defined by induction as:

$$\begin{array}{lcl}
\mathsf{Tr}_3(p) & = & p \\
\mathsf{Tr}_3(\neg\varphi) & = & \neg\mathsf{Tr}_3(\varphi) \\
\mathsf{Tr}_3(\varphi \wedge \psi) & = & \mathsf{Tr}_3(\varphi) \wedge \mathsf{Tr}_3(\psi) \\
\mathsf{Tr}_3(\Box_a\varphi) & = & \Box_a\mathsf{Tr}_3(\varphi) \\
\mathsf{Tr}_3(\mathsf{cp}(Q,q)\varphi) & = & [\mathcal{E}_{\mathsf{cp}}(Q), q]\mathsf{Tr}_3(\varphi) \\
\mathsf{Tr}_3(\mathsf{rm}(\Pi)\varphi) & = & \bigwedge\{[\mathcal{E}_{\mathsf{rm}}(\Pi), e]\mathsf{Tr}_3(\varphi) \mid e \in \mathcal{E}_{\mathsf{rm}}(\Pi)\},
\end{array}$$

where $\mathcal{E}_{\mathsf{cp}}(Q)$ is given in Definition 6.1 and $\mathcal{E}_{\mathsf{rm}}(\Pi)$ in Definition 6.11.

Unlike the translation from $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ into AMLE in Definition 6.7, here the preconditions in events of $\mathcal{E}_{\mathsf{rm}}(\Pi)$ in $\mathsf{Tr}_3(\varphi)$ contain converse operators (see Definition 6.11); this is why the image of Tr_3 is in $\mathsf{AMLE}(\Box^-)$ and not just AMLE. Note that the translation $\mathsf{Tr}_3(\varphi)$ is exponential even in the DAG size of φ, because the event model $\mathcal{E}_{\mathsf{rm}}(\pi)$ contains exponentially many events.

Theorem 6.14. *For every* $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ *formula* φ *and every pointed model* $\mathcal{M}, w$

$$\mathcal{M}, w \models \varphi \text{ iff } \mathcal{M}, w \models \mathsf{Tr}_3(\varphi).$$

Proof. The proof is almost exactly the same as for Theorem 6.8. The only differences are that for the isomorphy between updated models we use Lemma 6.12 instead of Lemma 6.6, and we use the fact that isomorphy implies two-way bisimulation and that $\mathsf{AMLE}(\Box^-)$ is invariant by two-way bisimulation. □

On normal forms for AML. Starting from an arbitrary formula in AML it is possible to apply the composition of the translations Tr_1 and Tr_2 to arrive at a formula in AMLE ($\Box^-$ is not needed in this case) which is at most polynomial in the DAG size of the original formula. Notice that the actions used in the translation of the one-step remove operator have at most four action points (see Definition 6.4). Moreover, even though, for simplicity, we did not restrict the size of action models used for the copy operator in Definition 6.1, Figure 3 shows how a sequence of models of size at most 2 could be used instead (the intuitive, simple idea is that each $\mathcal{E}_i$ creates one of the required copies of the original model, with the appropriate labelling).

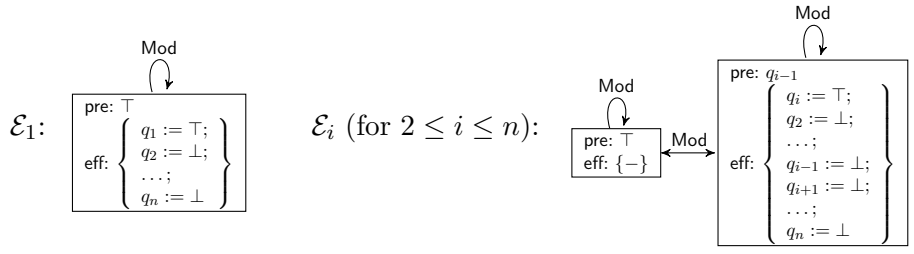


Figure 3. Sequence of action models $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ of size ≤ 2 needed to simulate $\mathcal{M}_{\mathsf{cp}(\{q_1,\ldots,q_n\})}$.

As a result, we can impose the following bound on the size of action models in AML:

Theorem 6.15. *For every* AML *formula* φ *there exists an* AMLE *formula* φ' *of DAG size at most polynomial in* φ *such that all actions models in* φ *have size at most 4, and for every pointed model* $\mathcal{M}, w$

$$\mathcal{M}, w \models \varphi \quad \textit{iff} \quad \mathcal{M}, w \models \varphi'.$$

The crucial point in Theorem 6.15 is that the reduction from AML formulas to AMLE formulas is *polynomial*. I.e., every AML formula has an equivalent (at most polynomially larger) formula in AMLE where actions models have size at most 4. We conjecture that this normal form actually holds for all formulas in AMLE.

7. Complexity of Deciding Satisfiability

It was proved in [8, 9] that the satisfiability problem for $\mathsf{ML}(\mathsf{cp})$ and for $\mathsf{ML}(\mathsf{rm}^1)$ with Boolean tests is PSPACE-complete. In this section we show that the latter result still holds for path expressions of arbitrary length, and with arbitrary test formulas (i.e., not only Boolean ones). We also establish the complexity of the full logic $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$, which we prove to be NEXPTIME-complete.

Notice that, the satisfiability for $\mathsf{ML}(\mathsf{rm}^1)$ with Boolean tests is shown PSPACE-complete in [9] by providing a polynomial translation from this language into $\mathsf{ML}(\Box^-)$. By using the same argument, PSPACE-completeness also holds for the logic $\mathsf{ML}(\mathsf{rm}^1)$ with arbitrary tests from this paper, if we use the DAG size of the formula.

Proposition 7.1. *The satisfiability problem for* $\mathsf{ML}(\mathsf{rm}^1)$ *is* PSPACE*-complete.*

Proof. The proof relies on a recursive application of the translation provided in [9]. □

We now proceed with the rest of the results. To obtain the above-mentioned results we first design a tableau method to solve the satisfiability problem for $\mathsf{AMLE}(\Box^-)$, showing that this problem is in NEXPTIME. We then extend this tableau method with two rules, one for copy and one for remove, which use the corresponding action models from Definitions 6.1 and 6.11. This provides a tableau method for the satisfiability problem for $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$, and allows us to show that this problem is in NEXPTIME, and actually NEXPTIME-complete. Finally, we show that in the particular case of $\mathsf{ML}(\mathsf{rm})$, the tableau method can be fine-tuned to run in polynomial space, thus establishing that the satisfiability problem for $\mathsf{ML}(\mathsf{rm})$ is PSPACE-complete, but using the usual tree size of a formula.

7.1. Complexity of $\mathsf{AMLE}(\Box^-)$

The satisfiability problem for AML is in NEXPTIME [11], and similarly for AMLE [42]. In this section, we adapt the tableau methods given in [11, 42] to handle both postconditions[1] and the converse modality (for this we draw inspiration from [31]). Notice that the upper bound result still holds when the input $\mathsf{AMLE}(\Box^-)$ formula is represented by a DAG.

Theorem 7.2. *The satisfiability problem for* $\mathsf{AMLE}(\Box^-)$ *is in* NEXPTIME.

[1] Prior to [42], a similar methodology was used in [47] for public announcements with effects.

Proof. We extend the tableau method in [11]. Let LAB be a countable set of labels used to represent states of the model $\mathcal{M}, w$ we are trying to construct. Our tableau method manipulates *tableau terms* of the following kind:

- $(\sigma \;\; \mathcal{E}_1, e_1; \ldots; \mathcal{E}_i, e_i \;\; \varphi)$ where $\sigma \in \mathsf{LAB}$ is a symbol (that represents a state in the initial model) and for all $j \in \{1, \ldots, i\}$, $\mathcal{E}_j, e_j$ is an action model. This term means that φ is true in the state denoted by σ after the execution of the sequence $\mathcal{E}_1, e_1, \ldots, \mathcal{E}_i, e_i$ and that the sequence is executable in the state denoted by σ.
- $(\sigma \;\; \mathcal{E}_1, e_1; \ldots; \mathcal{E}_i, e_i \;\; \checkmark)$ means that the sequence $\mathcal{E}_1, e_1, \ldots, \mathcal{E}_i, e_i$ is executable in the state denoted by σ. The symbol $\checkmark$ means that the state survives a sequence of pointed action models.
- $(\sigma \;\; \mathcal{E}_1, e_1; \ldots; \mathcal{E}_i, e_i \;\; \otimes)$ means that the sequence $\mathcal{E}_1, e_1, \ldots, \mathcal{E}_i, e_i$ is not executable in the state denoted by σ. The symbol $\otimes$ means that the state does not survive the sequence of pointed action models.
- $(\sigma R_a \sigma_1)$ means that the denotation of σ is R_a-related to the denotation of σ_1.
- $\bot$ denotes an inconsistency.

A *tableau rule* is represented by a *numerator* $\mathcal{N}$ above a line and a finite list of *denominators* $\mathcal{D}_1, \ldots, \mathcal{D}_k$ below this line, separated by vertical bars. In the following σ denotes a symbol for states. Σ, Σ', etc., denote sequences of pointed action models. λ denotes the empty sequence of pointed action models.

A *tableau tree* is a finite tree with a set of tableau terms at each node. A rule with numerator $\mathcal{N}$ is *applicable* to a node carrying a set Γ if Γ contains an instance of $\mathcal{N}$. If no rule is applicable, Γ is said to be *saturated*. We call a node σ an *end node* if the set of formulas Γ it carries is saturated, or if $\bot \in \Gamma$. The tableau tree is extended as follows:

(1) Choose a leaf node n carrying Γ where n is not an end node, and choose a rule ρ applicable to n.
(2) (a) If ρ has one denominator, add the appropriate instantiation to Γ.
    (b) If ρ has k denominators with $k > 1$, create k successor nodes for n, where each successor i carries the union of Γ with an appropriate instantiation of denominator $\mathcal{D}_i$.

A *branch* in a tableau tree is a path from the root to an end node. A branch is *closed* if its end node contains $\bot$, otherwise it is *open*. A tableau tree is *closed* if all its branches are closed, otherwise it is *open*.

The complete set of tableau rules is given in Figure 4. The tableau method contains the classical Boolean rules $(\wedge)$ and $(\neg\neg)$. It also contains the non-deterministic rule $(\neg\wedge)$ handling disjunction. The rule $(\bot)$ makes the current execution fail. Rule $\neg[\mathcal{E}, e]$ expresses that for $\neg[\mathcal{E}, e]\varphi$ to hold, the pointed event model $(\mathcal{E}, e)$ must be executable and the resulting state should not satisfy φ. For rule $[\mathcal{E}, e]$ instead, either $(\mathcal{E}, e)$ is not executable, or it is executable and φ holds in the resulting state. Rules $\neg[\mathcal{E}, \{e_1, \ldots, e_n\}]$ and $[\mathcal{E}, \{e_1, \ldots, e_n\}]$ treat the general case of multi-pointed event models by applying rule $\neg[\mathcal{E}, e]$ and $[\mathcal{E}, e]$, respectively, to each event. The rule for $(\Box_a)$ is applied for all $j \in \{1, \ldots i\}$ and all u'_j such that $w'_j R'_a u'_j$. Similarly, the rule for $(\neg\Box_a)$ is applied by choosing non-deterministically for all $j \in \{1, \ldots i\}$ some u'_j such that $w'_j R'_a u'_j$ and creating a new fresh label σ_{new}. Rules $(\checkmark)$, $(\otimes)$, $(\text{clash}_{\checkmark,\otimes})$ and $(\lambda_\otimes)$ handle preconditions: rule $(\checkmark)$ says that if a state survives, then the precondition should be true. Rule $(\otimes)$ involves non-determinism: either a state does not survive because the current precondition is false or because it did not survive because of a

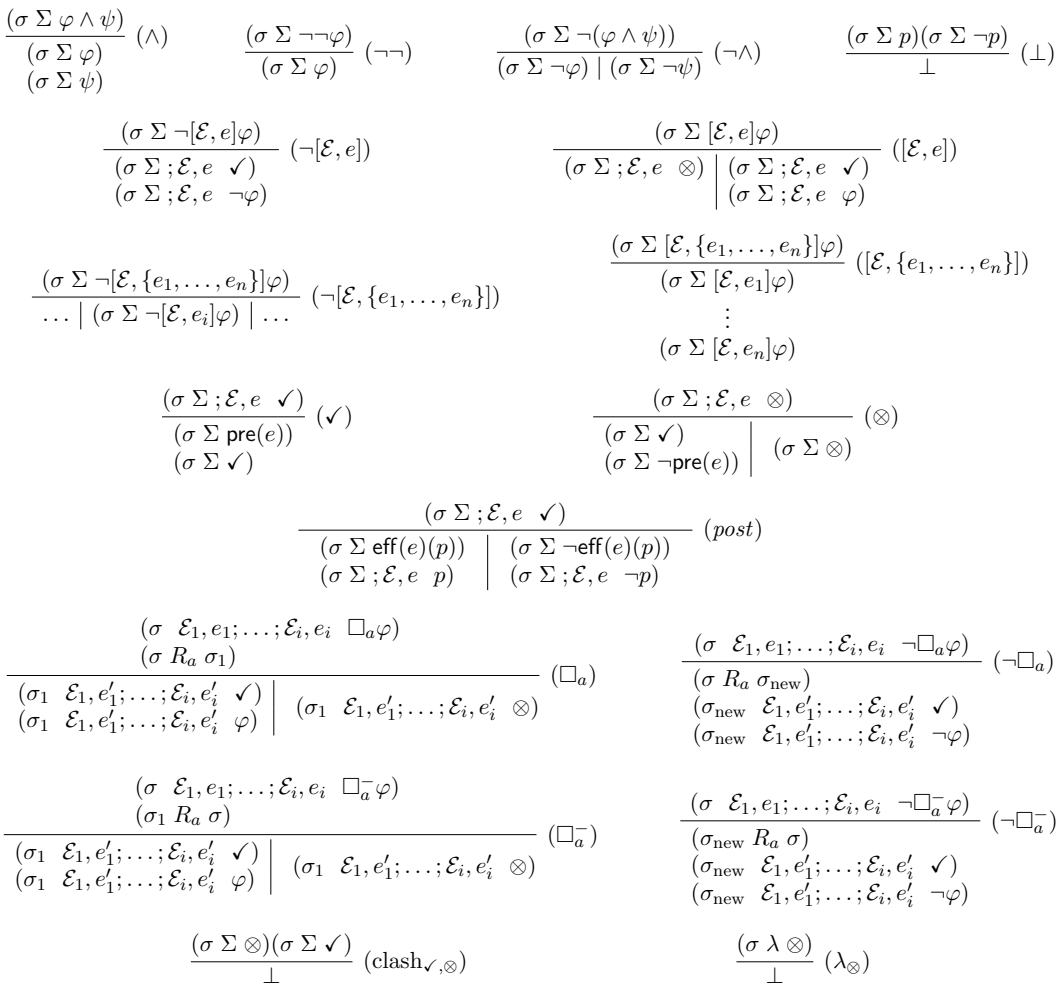


Figure 4. Tableau rules for AMLE($\Box^-$).

previous precondition. Rule ($\text{clash}_{\checkmark,\otimes}$) says that a state can not survive and not survive at the same time. Rule ($\lambda_\otimes$) says that a state always survives the empty sequence of pointed action models.

For action models without postconditions, Aucher et al. [11] proposed to bring back the values of atomic propositions with the following two rules: if Γ contains $(\sigma\ \Sigma\ p)$ (resp., $(\sigma\ \Sigma\ \neg p)$), then add $(\sigma\ \lambda\ p)$ (resp., $(\sigma\ \lambda\ \neg p)$) to Γ. These rules are no longer sound in the presence of postconditions, because the valuation may change after the application of an action model. We propose to replace them by the single rule (*post*). This rule is non-deterministic and says that either the postcondition was true and p is true now, or the postcondition was false and p is false now. This rule is applied for all atoms $p \in \text{Sub}(\varphi)$, where φ is the formula whose satisfiability is being tested.

The proof of soundness and completeness of the tableau is similar to the proof in [11]. To test whether φ is AMLE-satisfiable, start the tableau with $\Gamma := \{(\sigma\ \ \lambda\ \ \varphi)\}$. Given a branch Γ, consider the following tree $\mathcal{T}_\Gamma$:

- Nodes are labels $\sigma \in \mathsf{LAB}$ appearing in Γ;
- Two nodes σ, σ' are linked when a term of the form $(\sigma\ \ R_a\ \ \sigma')$ or $(\sigma'\ \ R_a\ \ \sigma)$ appears in Γ. Labels added later in Γ appear deeper in the tree. If σ is a node and a formula $\neg\Box_a\psi$ should hold at σ, we create a successor σ' with $(\sigma\ \ R_a\ \ \sigma')$

and we say that $\neg\psi$ holds in σ'. If σ is a node and a formula $\neg\Box_a^-\psi$ should hold at σ, we create a successor σ' with $(\sigma' \;\; R_a \;\; \sigma)$ and we say that $\neg\psi$ holds in σ'.

The idea is as follows. At any step of the algorithm, the depth of $\mathcal{T}_\Gamma$ is linear in the size of φ (number of nested diamonds, as in the case of $\mathsf{ML}(\Box^-)$ [29]). The arity of $\mathcal{T}_\Gamma$ is bounded by an exponential in φ since the set of terms of the form $(\sigma \;\; \Sigma \;\; \neg\Box_a\psi)$ and $(\sigma \;\; \Sigma \;\; \neg\Box_a^-\psi)$ is at most exponential for a given σ. Thus, the size of set Γ is bounded by an exponential in φ. Since at each step of the algorithm we add at least a new term in Γ, it takes an exponential amount of time to reach a situation in which Γ is saturated or a contradiction is found. The resulting tableau method can thus be turned into a non-deterministic algorithm running in exponential time. □

7.2. Complexity of Logics with Copy and Remove

We now establish the complexity of the satisfiability problem for the full logic $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ and for its fragments $\mathsf{ML}(\mathsf{cp},\mathsf{rm}^1)$ and $\mathsf{ML}(\mathsf{rm})$.

Theorem 7.3. *The satisfiability problem for* $\mathsf{ML}(\mathsf{cp},\mathsf{rm}^1)$ *and for* $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ *is* NExpTime*-complete.*

Proof. The lower bounds follow from the facts that there is a polynomial translation from AML with purely informative action models into $\mathsf{ML}(\mathsf{cp},\mathsf{rm}^1)$ that preserves satisfiability (see Definition 5.1 of Tr_1 and Theorem 5.2), and the satisfiability problem for AML is NExpTime-hard [11].

For the upper bounds we extend the tableau method for $\mathsf{AMLE}(\Box^-)$ presented in the previous section to obtain a tableau method for $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$. To do so we add two rules to those in Figure 4, one for the copy operator and one for the remove operator:

$$\frac{(\sigma\ \Sigma\ \mathsf{cp}(Q,q)\varphi)}{(\sigma\ \Sigma\ [\mathcal{E}_{\mathsf{cp}}(Q),q]\varphi)}\ (\mathsf{cp}) \qquad \frac{(\sigma\ \Sigma\ \mathsf{rm}(\Pi)\varphi)}{(\sigma\ \Sigma\ [\mathcal{E}_{\mathsf{rm}}(\Pi),E]\varphi)}\ (\mathsf{rm})$$

for E the set of all events in $\mathcal{E}_{\mathsf{rm}}(\Pi)$.

Observe that even if the language of $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ does not contain past modalities, the preconditions of models $\mathcal{E}_{\mathsf{rm}}(\Pi)$ that appear in rule rm introduce such modalities (see Definitions 6.9 and 6.11), so that we need the rules $\Box_a^-$ and $\neg\Box_a^-$.

The correctness of the obtained tableau method follows from Lemmas 6.3 and 6.12, which show that models $\mathcal{E}_{\mathsf{cp}}(Q)$ and $\mathcal{E}_{\mathsf{rm}}(\Pi)$ correctly simulate the application of $\mathsf{cp}(Q)$ and $\mathsf{rm}(\Pi)$, respectively.

As for the complexity, this tableau method can also be implemented by a non-deterministic procedure running in time exponential in the size of the input $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ formula φ: the depth of $\mathcal{T}_\Gamma$ at any step of the algorithm is still linear in the size of φ, and the arity of $\mathcal{T}_\Gamma$ is still bounded by an exponential in φ. Thus, the size of set Γ is bounded by an exponential in φ. The only difference is that now a single term can be of exponential size, because models $\mathcal{E}_{\mathsf{rm}}(\Pi)$ are of size exponential in the length of path expressions in Π. But an exponential number of terms of exponential size can still be enumerated in exponential time. Since at each step of the algorithm we add at least a new term in Γ, it takes an exponential amount of time to reach a situation in which Γ is saturated or a contradiction is found. □

For the following theorem, instead of the DAG size used in the rest of this work, we consider the traditional notion of formula size, that is, the size of the syntactic tree. For the DAG size the complexity may be higher, but this remains an open question.

Theorem 7.4. *The satisfiability problem for* $\mathsf{ML}(\mathsf{rm})$ *is* PSPACE*-complete (for the usual notion of formula size).*

Proof. Lower bound follows from the PSPACE-hardness of the satisfiability problem for ML [16]. For the upper bound, our starting point is the tableau method for $\mathsf{ML}(\mathsf{cp},\mathsf{rm})$ given by the rules we introduced above. Since the generated tree is linear in the modal depth (number of nested diamonds), we are *almost there* to provide a depth-first-search procedure that runs in polynomial space. To fill the gap, the rest of the proof consists in solving these three issues:

(1) in rule (rm), the formula creates the term $(\sigma\ \Sigma\ [\mathcal{E}_{\mathsf{rm}}(\Pi), E]\varphi)$ where $\mathcal{E}_{\mathsf{rm}}(\Pi)$ is of exponential size in $|\Pi|$;
(2) rule $([\mathcal{E}, \{e_1, \ldots, e_n\}])$ explicitly generates an exponential number of terms when $\mathcal{E}$ is $\mathcal{E}_{\mathsf{rm}}(\Pi)$ and $\{e_1, \ldots, e_n\}$ is E, as generated by rule (rm);
(3) at each node of the tree, there may be an exponential number of possible terms because there are an exponential number of sequences of events Σ.

In order to solve (1), we only write Π instead of $[\mathcal{E}_{\mathsf{rm}}(\Pi), E]$. In other words, we only write the relevant information in the term that is needed for the rules, instead of writing an explicit representation of $\mathcal{E}_{\mathsf{rm}}(\Pi)$ (the set of path expressions Π can be seen as a symbolic representation of $\mathcal{E}_{\mathsf{rm}}(\Pi)$).

In order to solve (2), let us first explain the solution in the case of a single path expression $\Pi = \{\pi\}$. Recall that preconditions of the different events in $\mathcal{E}_{\mathsf{rm}}(\pi)$ are mutually exclusive, and actually exactly one is satisfied in a given world (see Definition 6.11). Thus, we remark that exactly one event only of $\mathcal{E}_{\mathsf{rm}}(\pi)$ is applicable in a given world, depending on the truth values of the formulas pos_k^π in that world. As a result, instead of generating all terms $(\sigma\ \Sigma\ [\mathcal{E}, e_i]\varphi)$, we can guess which is the only event e_i that can be applied and only generate $(\sigma\ \Sigma\ [\mathcal{E}, e_i]\varphi)$. To ensure that we guess the right event e_i, we also add the term $(\sigma\ \Sigma\ \mathsf{pre}(e_i))$ which will check that the truth values of the different pos_k^π are those that correspond to event e_i. For the general case of a set Π of path expressions, by Definition 6.11 $\mathcal{E}_{\mathsf{rm}}(\Pi)$ is the synchronous product of all the $\mathcal{E}_{\mathsf{rm}}(\pi)$, for $\pi \in \Pi$. As before, for each $\pi \in \Pi$ there is a unique event $e_\pi \in \mathcal{E}_{\mathsf{rm}}(\pi)$ that is applicable in a given world. By definition of the synchronous product, there is therefore also a unique event e_Π in $\mathcal{E}_{\mathsf{rm}}(\Pi)$ that is applicable in a given world. This event is the tuple consisting of all the e_π for $\pi \in \Pi$, and it is characterised by the truth values of the different pos_k^π for $\pi \in \Pi$ and $0 \leq k \leq |\pi|$. We can thus again guess e_Π, add the term $(\sigma\ \Sigma\ [\mathcal{E}, e_\Pi]\varphi)$, and check this guess by adding the term $(\sigma\ \Sigma\ \mathsf{pre}(e_\Pi))$.

Issue (3) is actually solved by the trick used above to solve issue (2). Sequences $\mathcal{E}_1, e_1; \ldots; \mathcal{E}_i, e_i$ appearing in terms of the tableau method come from nested remove operators in the original formula. The number of sequences of nested remove operators is linear in the size of the formula, for the usual definition of formula size; and as we explained above, for each such model we can guess the only event that is applicable in the current world. Note that if the presentation is in DAG form there might be an exponential number of sequences of nested remove operators.

The tableau method that we obtain generates a tree of polynomial depth (linear in the modal depth of the formula). Rules $(\neg\Box_a)$ and $(\neg\Box_a^-)$ generate at most a polynomial number of new world symbols σ_{new}, at most one for each subformula and each sequence Σ of nested pointed event models in the original formula; so branching is also polynomial. By the considerations above, each node also contains a polynomial number of terms. As a result, a depth-first search implementation of the tableau method leads to a non-deterministic algorithm that runs in polynomial space. The complexity

result then follows from Savitch's theorem. □

8. Related work

In [2, 3, 23, 5] a family of logics called *relation-changing logics* are studied. These articles consider modalities with the ability of modifying the accessibility relation of a model while evaluating a formula. Three kinds of updates are considered, performing both local and global effects: adding, deleting and swapping around edges. All the obtained logics are very expressive (and all different), and they are computationally untractable (their satisfiability problem is undecidable). Among these modalities, the *sabotage* operator is particularly interesting, and it was also investigated in [37, 36, 45, 27]. In one of its versions, sabotage deletes an arbitrary arrow in the model, and afterwards, formulas are evaluated in the updated model. In this sense, it is similar to the edge-removal modality rm studied in this paper. But while sabotage deletes edges arbitrarily in the model, the effects of rm depend on the satisfiability of certain formulas that are explicit in the operation. Thus, rm performs more uniform updates, leading to a decidable logic as we show in this article.

Another example of very expressive relation-changing logics with untractable reasoning tasks is the *modal separation logic* family from [19, 20, 21, 22]. Over arbitrary models, modal separation logics become undecidable very quicky (or with very high complexity even on tree models, see [14]). This is a consequence of the second-order features hidden behind separation operations, not present in rm.

Operators performing more controlled updates have also been introduced in the literature. In [10] global and local graph modifiers are proposed, where the modifications can concern both the valuations of propositional variables (also known as ontic change) and the accessibility relations, as is the case with $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$. A global graph modification $a-(\varphi,\psi)$ [10, page 295] corresponds to our rm^1 action $(\varphi; a; \psi)$, whereas a local graph modification of that kind would amount to a sabotage operator as the one in [35]. However, the only local graph modifiers considered there are of the ontic change kind [10, page 300]. Complexity and expressivity are studied for the logic combining different modifiers, but not for each of them separately.

The family of arrow update logics [32, 33] shares several features with $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$. Arrow update logic [32] is a dynamic epistemic logic with model changing modalities called 'arrow updates'. An arrow for agent a is a pair (w, v) in the respective accessibility relation for a, such that w satisfies a *source condition* φ and v satisfies a *target condition* ψ. Thus, an *arrow update* $(\varphi; a; \psi)$ preserves all such pairs. Using our notation, an arrow update preserving all arrows $(\varphi; a; \psi)$ corresponds to three simultaneous rm^1 actions $(\neg\varphi; a; \neg\psi)$, $(\neg\varphi; a; \psi)$, and $(\varphi; a; \neg\psi)$. Dually, each $\mathsf{rm}(\varphi; a; \psi)$ corresponds to three simultaneous arrow updates $(\neg\varphi; a; \neg\psi)$, $(\neg\varphi; a; \psi)$, and $(\varphi; a; \neg\psi)$). In [32] the authors do not give complexity results but focus on expressivity and succinctness. It is well-known that arrow updates cannot 'simulate' action models in our sense of a compositional translation Tr for which $\mathsf{Tr}([\alpha]\varphi)$ is equivalent to some expression with components $\mathsf{Tr}(\alpha)$ and $\mathsf{Tr}(\varphi)$. In [33] the results of [32] are generalised to so-called *arrow update models*. Instead of the aforementioned arrow updates, these structures consist of a domain of outcomes, thus the arrows $(\varphi; a; \psi)$ are specified between different outcomes. In that sense, the arrow update from [33] is a singleton arrow update model. It is then shown that for this arrow update model logic there is a compositional translation (in the above sense) between arrow update models and action models and vice versa. Such a translation is also explicitly given in [51].

Clearly we achieve a similar effect to that of arrow update models by the interaction of cp and rm, as each point in an arrow update model generates a copy of the initial relational model in which it is executed. An elegant correspondence between both approaches is still missing. First, the arrow preservation and copying process in the arrow update model is simultaneous, instead of being divided into two primitives such as cp and rm. Second, edges removal and edge preservation are dual notions. Recall that expressivity of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ increases since it allows simultaneous path removal.

Finally, the connections between the modal logic of copy and remove and the various action model logics that we presented are interesting examples of what is *called update equivalence* in [33]. Two dynamic modalities interpreted as model transformers are update equivalent if, whenever executed in a given epistemic model, they always result in bisimilar epistemic models. Update equivalence is also related to *action emulation* [52].

9. Conclusion

We proposed the dynamic modal logic $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ which contains copy and remove operators. We investigated model theoretic properties of $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ such as its relative expressive power, introducing the notion of path bisimulations. With this notion at hand, we have been able to show that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ lies strictly between the basic modal logic ML and its extension with the inverse modality $\mathsf{ML}(\Box^-)$. As a relevant line for future research one would try to prove whether this notion exactly captures the expressive power of the language over finite or image-finite models [28], and whether a van Benthem Characterisation Theorem holds [44]. Moreover, it would be interesting to reveal whether the cp modality increases the expressivity of $\mathsf{ML}(\mathsf{rm})$ or not. We conjecture that $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ is strictly more expressive than $\mathsf{ML}(\mathsf{rm})$.

We showed that the action model logic AML, one of the best-known dynamic epistemic logics, can be polynomially embedded in $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ (the fragment with length-one path removals). This is in line with the previously known result that, when AML is evaluated on the class of finite models, action model execution corresponds to model restriction ('remove') on a bisimilar copy ('copy') of the initial model [46]. The embedding simulates every finite action model with a combination of copy and remove operators. The embedding can be done within $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ as it only requires single-step removals. We showed that the copy and one-step removal themselves correspond to particular action models. As a result we obtain a decomposition method for action models. By decomposing product updates in sequences of copy and remove operators, it would be possible to characterise syntactic fragments of AML which might lead to interesting complexity results for the satisfiability problem.

Finally, we show that the complexity of the satisfiability problems of the full language $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ and its fragment $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ are NExpTime-complete; while satisfiability for the fragments $\mathsf{ML}(\mathsf{cp})$, $\mathsf{ML}(\mathsf{rm})$ and $\mathsf{ML}(\mathsf{rm}^1)$ is PSpace-complete. To achieve that, we designed a tableau method which handles cp and rm modalites. Notice that all results are stated using the DAG size of a formula, except by $\mathsf{ML}(\mathsf{rm})$ in which the usual tree size has been used. These complexity results improve claims made in [9][2], and they are summarised in Figure 5.

[2] In [9], the stated NExpTime lower bounds for $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ and $\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$ (with Boolean tests) are claimed to follow from the NExpTime hardness of the satisfiability problem for AML with *single-pointed* action models. However the known result of NExpTime hardness holds for AML with *multi-pointed* action models.

$\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$	NExpTime-complete
$\mathsf{ML}(\mathsf{cp}, \mathsf{rm}^1)$	NExpTime-complete
$\mathsf{ML}(\mathsf{cp})$	PSpace-complete
$\mathsf{ML}(\mathsf{rm}^1)$	PSpace-complete
$\mathsf{ML}(\mathsf{rm})$	PSpace-complete

Figure 5. Complexity of the satisfiability problem for $\mathsf{ML}(\mathsf{cp}, \mathsf{rm})$ and some of its syntactic fragments. The last result is for the usual formula size (aka the "tree size"), the others use the DAG-size of formulas.

Acknowledgments We would like to thank the two anonymous reviewers for their helpful comments and suggestions. C. Areces and R. Fervari are partially supported by projects ANPCyT-PICTs-2017-1130, Stic-AmSud 20-STIC-03 'DyLo-MPC', Secyt-UNC, GRFT Mincyt-Cba, and by the Laboratoire International Associé SINFIN.

References

[1] T. Ågotnes and H. van Ditmarsch. What will they say? - Public Announcement Games. *Synthese*, 179(1):57–85, 2011.

[2] C. Areces, R. Fervari, and G. Hoffmann. Moving arrows and four model checking results. In L. Ong and R. Queiroz, editors, *Logic, Language, Information and Computation*, volume 7456 of *LNCS*, pages 142–153. Springer, 2012.

[3] C. Areces, R. Fervari, and G. Hoffmann. Swap logic. *Logic Journal of the IGPL*, 22(2):309–332, 2014.

[4] C. Areces, R. Fervari, and G. Hoffmann. Relation-changing modal operators. *Logic Journal of the IGPL*, 23(4):601–627, 2015.

[5] C. Areces, R. Fervari, G. Hoffmann, and M. Martel. Satisfiability for relation-changing logics. *Journal of Logic and Computation*, 28(7):1443–1470, 2018.

[6] C. Areces, R. Fervari, G. Hoffmann, and M. Martel. Undecidability of relation-changing modal logics. In *Dynamic Logic. New Trends and Applications. DALI 2017*, volume 10669 of *LNCS*, pages 1–16. Springer, 2018.

[7] C. Areces and B. ten Cate. Hybrid logics. In Blackburn et al. [18], pages 821–868.

[8] C. Areces, H. van Ditmarsch, R. Fervari, and F. Schwarzentruber. Logics with copy and remove. In *Proceedings of WoLLIC 2014*. Springer, 2014.

[9] C. Areces, H. van Ditmarsch, R. Fervari, and F. Schwarzentruber. The modal logic of copy and remove. *Information and Computation*, 255:243–261, 2017.

[10] G. Aucher, P. Balbiani, L. Fariñas Del Cerro, and A. Herzig. Global and local graph modifiers. *Electronic Notes in Theoretical Computer Science (ENTCS), Special issue Proceedings of the 5th Workshop on Methods for Modalities (M4M5 2007)*, 231:293–307, 2009.

[11] G. Aucher and F. Schwarzentruber. On the complexity of dynamic epistemic logic. In *Proceedings of TARK 2013*, Chennai, India, January 2013. Electronic Notes in Theoretical Computer Science (ENTCS).

[12] G. Aucher, J. van Benthem, and D. Grossi. Modal logics of sabotage revisited. *Journal of Logic and Computation*, 28(2):269–303, 2018.

[13] A. Baltag, L. Moss, and S. Solecki. The logic of public announcements, common knowledge and private suspicions. In I. Gilboa, editor, *TARK 1998*, pages 43–56, Evanstin, IL, USA, July 1998. Morgan Kaufmann.

[14] B. Bednarczyk, S. Demri, R. Fervari, and A. Mansutti. Modal logics with composition on finite forests: Expressivity and complexity. In *35th Annual ACM/IEEE*

Symposium on Logic In Computer Science (LICS'20), pages 167–180. IEEE Press, 2020.

[15] M. Benevides and I. Lima. Dynamic epistemic logic with communication actions. *Electronic Notes in Theoretical Computer Science*, 344:67–82, 2019.

[16] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2001.

[17] P. Blackburn and J. van Benthem. Modal logic: A semantic perspective. In Blackburn et al. [18].

[18] P. Blackburn, F. Wolter, and J. van Benthem, editors. *Handbook of Modal Logics*, volume 3 of *Studies in logic and practical reasoning*. Elsevier, 2006.

[19] S. Demri and M. Deters. Separation logics and modalities: a survey. *Journal of Applied Non-Classical Logics*, 25(1):50–99, 2015.

[20] S. Demri and R. Fervari. On the complexity of modal separation logics. In *AiML'18*, pages 179–198. College Publications, 2018.

[21] S. Demri and R. Fervari. The power of modal separation logics. *Journal of Logic and Computation*, 29(8):1139–1184, 2019.

[22] S. Demri, R. Fervari, and A. Mansutti. Axiomatising logics with separating conjunctions and modalities. In *JELIA'19*, volume 11468 of *LNAI*, pages 692–708. Springer, 2019.

[23] R. Fervari. *Relation-Changing Modal Logics*. PhD thesis, Facultad de Matemática Astronomía y Física, Universidad Nacional de Córdoba, Argentina, 2014.

[24] R. Fervari and F. Velázquez-Quesada. Dynamic epistemic logics of introspection. In *DaLí 2017*, volume 10669 of *LNCS*, pages 82–97. Springer, 2017.

[25] R. Fervari and F. Velázquez-Quesada. Introspection as an action in relational models. *Journal of Logical and Algebraic Methods in Programming*, 108:1–23, 2019.

[26] T. French, J. Hales, and E. Tay. A composable language for action models. In *AiML'14*, pages 197–216. College Publications, 2014.

[27] N. Gierasimczuk, L. Kurzen, and F. Velázquez-Quesada. Learning and teaching as a game: A sabotage approach. In X. He, J. Horty, and E. Pacuit, editors, *Logic, Rationality, and Interaction, Second International Workshop, LORI 2009*, volume 5834 of *LNCS*, pages 119–132. Springer, 2009.

[28] R. Goldblatt. Saturation and the Hennessy-Milner property. In A. Ponse, M. de Rijke, and Y. Venema, editors, *Modal Logic and Process Algebra*, number 53 in CSLI Lecture Notes, pages 107–129. CSLI Publications, Stanford, 1995.

[29] R. Goré. Tableau methods for modal and temporal logics. In M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga, editors, *Handbook of Tableau Methods*, pages 297–396. Springer, 1999.

[30] D. Harel. Dynamic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic. Vol. II*, volume 165 of *Synthese Library*, pages 497–604. D. Reidel Publishing Co., Dordrecht, 1984. Extensions of classical logic.

[31] I. Horrocks, U. Hustadt, U. Sattler, and R. Schmidt. Computational modal logic. In Blackburn et al. [18], pages 181–245.

[32] B. Kooi and B. Renne. Arrow update logic. *Review of Symbolic Logic*, 4(4):536–559, 2011.

[33] B. Kooi and B. Renne. Generalized arrow update logic. In K. Apt, editor, *TARK*, pages 205–211. ACM, 2011.

[34] S. Kripke. Semantical analysis of modal logic I. Normal propositional calculi. *Zeitschrift fur mathematische Logik und Grundlagen der Mathematik*, 9:67–96, 1963.

[35] D. Li. Losing connection: the modal logic of definable link deletion. *Journal of Logic and Computation*, 30(3):715–743, 2020.

[36] C. Löding and P. Rohde. Model checking and satisfiability for sabotage modal logic. In P. Pandya and J. Radhakrishnan, editors, *Proceedings of Foundations of Software Technology and Theoretical Computer Science, 23rd Conference*, volume 2914 of *LNCS*, pages 302–313. Springer, 2003.

[37] C. Löding and P. Rohde. Solving the sabotage game is PSPACE-hard. In *Mathematical Foundations of Computer Science 2003*, volume 2747 of *LNCS*, pages 531–540. Springer, Berlin, 2003.

[38] B. Maubert, S. Pinchinat, F. Schwarzentruber, and S. Stranieri. Concurrent games in dynamic epistemic logic. In C. Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 1877–1883, 2020.

[39] S. Mera. *Modal Memory Logics*. PhD thesis, Universidad de Buenos Aires, and UFR STMIA - Ecole Doctorale IAEM, 2009.

[40] M. Otto. Modal and guarded characterisation theorems over finite transition systems. *Annals of Pure and Applied Logic*, 130(1-3):173–205, 2004.

[41] D. Sangiorgi. On the origins of bisimulation and coinduction. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 31(4), 2009.

[42] F. Schwarzentruber. Epistemic reasoning in AI, 2019. Habilitation à diriger des recherches.

[43] B. ten Cate and M. Marx. Axiomatizing the logical core of XPath 2.0. *Theory of Computing Systems*, 44(4):561–589, 2009.

[44] J. van Benthem. *Modal Correspondence Theory*. PhD thesis, University of Amsterdam, 1976.

[45] J. van Benthem. An essay on sabotage and obstruction. In D. Hutter and W. Stephan, editors, *Mechanizing Mathematical Reasoning*, volume 2605 of *LNCS*, pages 268–276. Springer, 2005.

[46] H. van Ditmarsch and T. French. Simulation and information. In J. Broersen and J. Meyer, editors, *Knowledge Representation for Agents and Multi-Agent Systems*, LNAI 5605, pages 51–65. Springer, 2009.

[47] H. van Ditmarsch, A. Herzig, and T. de Lima. Public announcements, public assignments and the complexity of their logic. *Journal of Applied Non-Classical Logics*, 22(3):249–273, 2012.

[48] H. van Ditmarsch and B. Kooi. Semantic results for ontic and epistemic change. In *Proceedings of 7th LOFT*, Texts in Logic and Games 3, pages 87–117. Amsterdam University Press, 2008.

[49] H. van Ditmarsch, W. van der Hoek, and B. Kooi. Concurrent dynamic epistemic logic. In V. Hendricks, K. Jørgensen, and S. Pedersen, editors, *Knowledge Contributors*, pages 45–82, Dordrecht, 2003. Kluwer Academic Publishers. Synthese Library Volume 322.

[50] H. van Ditmarsch, W. van der Hoek, and B. Kooi. *Dynamic Epistemic Logic*. Kluwer, 2007.

[51] H. van Ditmarsch, W. van der Hoek, B. Kooi, and L. Kuijer. Arrow update synthesis. *Information and Computation*, 2020. Online first at https://doi.org/10.1016/j.ic.2020.104544.

[52] J. van Eijck, J. Ruan, and T. Sadzik. Action emulation. *Synthese*, 185(1):131–151, 2012.

[53] J. van Eijck, F. Sietsma, and Y. Wang. Composing models. *Journal of Applied Non-Classical Logics*, 21(3-4):397–425, 2011.