

Satisfiability for Relation-Changing Logics

Carlos Areces^{1,2} Raul Fervari^{1,2}
Guillaume Hoffmann^{1,2} Mauricio Martel³

¹FaMAF, Universidad Nacional de Córdoba, Argentina

²Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
{areces, fervari, hoffmann}@famaf.unc.edu.ar

³Fachbereich Mathematik und Informatik, Universität Bremen, Germany
martel@informatik.uni-bremen.de

Abstract

Relation-changing modal logics are extensions of the basic modal logic with dynamic operators that modify the accessibility relation of a model during the evaluation of a formula. These languages are equipped with dynamic modalities that are able, for example, to delete, add, and swap edges in the model, both locally and globally. We study the satisfiability problem for some of these logics. We first show that they can be translated into hybrid logic. As a result, we can transfer some results from hybrid logics to relation-changing modal logics. We discuss in particular, decidability for some fragments. We then show that satisfiability is, in general, undecidable for all the languages introduced, via translations from memory logics.

Keywords: modal logic, dynamic logics, satisfiability, undecidability.

1 Introduction

Modal logics [18, 16] were originally conceived as logics of necessary and possible truths. They are now viewed, more broadly, as logics that explore a wide range of modalities, or modes of truth: epistemic (“it is known that”), doxastic (“it is believed that”), deontic (“it ought to be the case that”), or temporal (“it has been the case that”), among others. From a model-theoretic perspective, the field evolved into a discipline that deals with languages interpreted on various kinds of relational structures or graphs. Nowadays, modal logics are actively used in areas as diverse as software verification, artificial intelligence, semantics and pragmatics of natural language, law, philosophy, etc.

As we just mentioned, from an abstract point of view, modal logics can be seen as formal languages to navigate and explore properties of a given relational structure. If we are interested, on the other hand, in describing how a given relational structure evolves (through time or through the application of certain operations) then classical modal languages seem, a priori, to fall short of the mark. Of course, it is possible to statically model the whole space of possible transformations as a graph, and use modal languages at that level, but this soon becomes unwieldy (see [5, 19] for some results using this approach). It is

also possible to represent model update conditions as *parts* of the model itself, and interact with them by means of the classical modal language. This is the approach taken by Gabbay’s in his study of reactive Kripke frames [23, 24]. Alternatively, it is possible to use standard relational models, and use modal languages with *dynamic modalities* encoding the desired changes.

There exist several dynamic modal logics that fit in this last approach. A clear example are the dynamic operators introduced in dynamic epistemic logics (see, e.g., [39]). These operators are used to model changes in the epistemic state of an agent by removing edges from the graph that represents the information states the agent considers possible. A less obvious example is given by hybrid logics [11, 17] equipped with the down arrow operator \downarrow which is used to ‘rebind’ names to the current point of evaluation. Finally, a classical example is sabotage logic introduced by van Benthem in [38]. The sabotage operator deletes individual edges in a graph and was introduced to model the *sabotage game*. This game is played on a graph by two players, *Runner* and *Blocker*. Runner can move on the graph from node to accessible node, starting from a designated point, and with the goal of reaching a given final point. Blocker, on the other hand, can delete one edge from the graph every time it is his turn. Runner wins if he manages to move from the origin to the final point, while Blocker wins otherwise. Van Benthem turns the sabotage game into a modal logic, where the (global) sabotage operator $\langle \text{gsb} \rangle$ models the moves of Blocker, and is interpreted on a graph \mathcal{M} at a point w as:

$$\mathcal{M}, w \models \langle \text{gsb} \rangle \varphi \text{ iff there is an edge } (u, v) \text{ of } \mathcal{M} \text{ such that } \mathcal{M}_{(u,v)}^-, w \models \varphi$$

where $\mathcal{M}_{(u,v)}^-$ is identical to \mathcal{M} except that the edge (u, v) has been removed. The moves of Runner, on the other hand, can be modeled using the standard \diamond operator of classical modal logics.

More recently, sabotage logic was proposed as a formalism for reasoning about formal learning theory [25]. Learning can be seen as a game with two players, *Teacher* and *Learner*, where Learner changes his information state through a step-by-step process. The process is successful if he eventually reaches an information state describing the real state of affairs. The information that Teacher provides can be interpreted as feedback about Learner’s conjectures about the current state of affairs, allowing him to discard inconsistent hypotheses. From this game-theoretical perspective, the interaction between Teacher and Learner can be modeled using operators similar to those of sabotage logic.

The dynamic approach seems appealing and flexible: it is easy to come up with situations that nicely fit and extend the examples we mentioned. Discovering alternative routes for Runner in van Benthem’s sabotage game, or possible shortcuts that Learner can take in learning theory can be modeled by adding new edges to the graph. Swapping an edge can be used to represent other scenarios such as changing the direction of a route, or allowing Learner to return to a previous information state. All these primitives can be turned into a modal logic in order to obtain a formal language for reasoning about these scenarios.

Motivated by applications like the ones we just described, in this article we investigate three dynamic primitives that can change the accessibility relation of a model: *sabotage* (deletes edges from the model), *bridge* (adds edges to a model), and *swap* (turns around edges), both in a global version (performing changes anywhere in the model) and local (changing adjacent edges from the

evaluation point). The particular operators we will investigate should be seen as just examples of the possibilities offered by the framework, with no intention of being complete or comprehensive. Intuitively, they were chosen because they represented simple, different ways in which a relation could be updated.

The six primitive operators we will discuss in this article were introduced in [3] where we investigated their expressive power and the complexity of their model checking problem. We presented tableaux methods for relation-changing modal logics in [4]. In [5] we studied local swap logic, in particular its decidability problem and its relation with first-order and hybrid logics. Global sabotage logic, together with some variants of relation-changing logics, have been investigated in [14, 15]. In [6] a general framework for representing model updates is defined. Connections with dynamic epistemic logic were introduced in [12, 13].

Contributions. In this article, we focus on the decidability problem of relation-changing logics. We first show that these logics can be seen as fragments of hybrid logics. We consider hybrid logics because they can naturally simulate the semantics of relation-changing operators. We introduce translations to $\mathcal{HL}(\mathbf{E}, \downarrow)$, the basic modal logic extended with nominals, the down arrow binder \downarrow , and the universal modality \mathbf{E} . The translations were implemented in the hybrid logic prover HTab [27], which can now be used as a tool for satisfiability checking and model building for relation-changing logics. We then show that all the relation-changing logics presented are undecidable, by means of reductions from memory logics – a weaker version of hybrid logics [1, 32].

This article presents and extends, in a uniform way, results that have been incrementally obtained in recent years. Translations from relation-changing logics to hybrid logics were investigated in [7]. The undecidability results have been established in [29, 19, 5, 30, 8].

Outline. The article is organized as follows. In Section 2 we introduce the syntax and semantics of relation-changing modal logics. In Section 3 we introduce hybrid extensions of the basic modal logic, and encode relation-changing operators into them. In Section 4 we present undecidability results. An implementation of the translations from Section 3 is described in Section 5, together with some examples. Finally, we conclude with some remarks and future work in Section 6.

2 Relation-Changing Modal Logics

In this section, we formally introduce extensions of the basic modal logic with relation-changing operators. We call these extensions Relation-Changing modal logics (RC for short). For more details and motivations, we direct the reader to [19].

Definition 1 (Syntax). Let PROP be a countable, infinite set of propositional symbols. The set FORM of formulas over PROP is defined as:

$$\text{FORM} ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \Diamond\varphi \mid \blacklozenge\varphi,$$

where $p \in \text{PROP}$, $\blacklozenge \in \{\langle \text{sb} \rangle, \langle \text{br} \rangle, \langle \text{sw} \rangle, \langle \text{gsb} \rangle, \langle \text{gbr} \rangle, \langle \text{gsw} \rangle\}$, and $\varphi, \psi \in \text{FORM}$. Other operators are defined as usual.

Let \mathcal{ML} (the basic modal logic) be the logic without the $\{\langle \text{sb} \rangle, \langle \text{br} \rangle, \langle \text{sw} \rangle, \langle \text{gsb} \rangle, \langle \text{gbr} \rangle, \langle \text{gsw} \rangle\}$ operators, and $\mathcal{ML}(\diamond)$ the extension of \mathcal{ML} allowing also \diamond , for $\diamond \in \{\langle \text{sb} \rangle, \langle \text{br} \rangle, \langle \text{sw} \rangle, \langle \text{gsb} \rangle, \langle \text{gbr} \rangle, \langle \text{gsw} \rangle\}$.

In particular, $\mathcal{ML}(\langle \text{sb} \rangle, \langle \text{gsb} \rangle)$, $\mathcal{ML}(\langle \text{br} \rangle, \langle \text{gbr} \rangle)$, and $\mathcal{ML}(\langle \text{sw} \rangle, \langle \text{gsw} \rangle)$ will be called Sabotage Logic, Bridge Logic, and Swap Logic, respectively.

Semantically, formulas are evaluated in standard relational models, and the meaning of the operators of the basic modal logic remains unchanged (see [16] for details). When we evaluate formulas containing relation-changing operators, we will need to keep track of the edges that have been modified. To that end, let us define precisely the models that we will use.

Definition 2 (Models and model updates). A model \mathcal{M} is a triple $\mathcal{M} = \langle W, R, V \rangle$, where W is a non-empty set whose elements are called points or states; $R \subseteq W \times W$ is the accessibility relation and the elements in R are called edges or arrows; finally $V : \text{PROP} \rightarrow \mathcal{P}(W)$ is called the valuation. We define the following notation:

$$\begin{aligned} \text{(sabotaging)} \quad \mathcal{M}_S^- &= \langle W, R_S^-, V \rangle, \text{ with } R_S^- = R \setminus S, S \subseteq R. \\ \text{(bridging)} \quad \mathcal{M}_S^+ &= \langle W, R_S^+, V \rangle, \text{ with } R_S^+ = R \cup S, S \subseteq (W \times W) \setminus R. \\ \text{(swapping)} \quad \mathcal{M}_S^* &= \langle W, R_S^*, V \rangle, \text{ with } R_S^* = (R \setminus S^{-1}) \cup S, S \subseteq R^{-1}. \end{aligned}$$

Intuitively, \mathcal{M}_S^- is obtained from \mathcal{M} by deleting the edges in S ; similarly, \mathcal{M}_S^+ adds the edges in S to the accessibility relation, and \mathcal{M}_S^* adds the edges in S as inverses of edges previously in the accessibility relation. These operators can be seen as particular cases of the *jump functions* introduced in [22], or the model update functions from [6].

Let w be a state in \mathcal{M} , the pair (\mathcal{M}, w) is called a pointed model; we will usually drop parenthesis and write \mathcal{M}, w instead of (\mathcal{M}, w) . In the rest of this article, we will use wv as a shorthand for $\{(w, v)\}$ or (w, v) ; context will always disambiguate the intended use.

Definition 3 (Semantics). Given a pointed model \mathcal{M}, w and a formula φ , we say that \mathcal{M}, w satisfies φ , and write $\mathcal{M}, w \models \varphi$, when

$$\begin{aligned} \mathcal{M}, w \models p & \quad \text{iff} \quad w \in V(p) \\ \mathcal{M}, w \models \neg \varphi & \quad \text{iff} \quad \mathcal{M}, w \not\models \varphi \\ \mathcal{M}, w \models \varphi \wedge \psi & \quad \text{iff} \quad \mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, w \models \psi \\ \mathcal{M}, w \models \diamond \varphi & \quad \text{iff} \quad \text{for some } v \in W \text{ s.t. } (w, v) \in R, \mathcal{M}, v \models \varphi \\ \mathcal{M}, w \models \langle \text{sb} \rangle \varphi & \quad \text{iff} \quad \text{for some } v \in W \text{ s.t. } (w, v) \in R, \mathcal{M}_{wv}^-, v \models \varphi \\ \mathcal{M}, w \models \langle \text{br} \rangle \varphi & \quad \text{iff} \quad \text{for some } v \in W \text{ s.t. } (w, v) \notin R, \mathcal{M}_{wv}^+, v \models \varphi \\ \mathcal{M}, w \models \langle \text{sw} \rangle \varphi & \quad \text{iff} \quad \text{for some } v \in W \text{ s.t. } (w, v) \in R, \mathcal{M}_{vw}^*, v \models \varphi \\ \mathcal{M}, w \models \langle \text{gsb} \rangle \varphi & \quad \text{iff} \quad \text{for some } v, u \in W, \text{ s.t. } (v, u) \in R, \mathcal{M}_{vu}^-, w \models \varphi \\ \mathcal{M}, w \models \langle \text{gbr} \rangle \varphi & \quad \text{iff} \quad \text{for some } v, u \in W, \text{ s.t. } (v, u) \notin R, \mathcal{M}_{vu}^+, w \models \varphi \\ \mathcal{M}, w \models \langle \text{gsw} \rangle \varphi & \quad \text{iff} \quad \text{for some } v, u \in W, \text{ s.t. } (v, u) \in R, \mathcal{M}_{uv}^*, w \models \varphi. \end{aligned}$$

We say that φ is satisfiable if for some pointed model \mathcal{M}, w , $\mathcal{M}, w \models \varphi$.

The meaning of the relation-changing operators $\langle \text{sb} \rangle$ (local sabotage), $\langle \text{br} \rangle$ (local bridge), $\langle \text{sw} \rangle$ (local swap), $\langle \text{gsb} \rangle$ (global sabotage), $\langle \text{gbr} \rangle$ (global bridge) and $\langle \text{gsw} \rangle$ (global swap) should be clear from the semantic conditions above. The local operators alter one arrow which is adjacent to the point of evaluation (deleting, adding and swapping it, respectively) while the global versions can change an arrow anywhere in the model.

3 Translations to Hybrid Logics

3.1 Hybrid Logic

We introduce several extensions of the basic modal logic \mathcal{ML} . The existential modality [26], written $E\varphi$, extends \mathcal{ML} in the following way:

$$\mathcal{M}, w \models E\varphi \quad \text{iff} \quad \text{for some } v \in W, \mathcal{M}, v \models \varphi.$$

In words, $E\varphi$ is true at a state w if φ is true somewhere in the model. The E operator, with its dual A , has been extensively investigated in classical modal logic [35].

Now we consider several traditional ‘hybrid’ operators (see [11] for details): nominals, the satisfaction operator, and the down-arrow binder. The basic hybrid logic \mathcal{HL} is obtained by adding *nominals* to \mathcal{ML} . A nominal is a propositional symbol that is true at exactly one state in a model. Fix the signature $\langle \text{PROP}, \text{NOM} \rangle$, with $\text{NOM} \subseteq \text{PROP}$. For $n \in \text{NOM}$, we require that its valuation is a singleton set, i.e., there is a single state w such that $V(n) = \{w\}$. In addition to nominals, hybrid logic typically involves the *satisfaction operator*. Given a nominal n and a formula φ , the satisfaction operator is written $n:\varphi$. The intended meaning is “ φ is true at the state named by n ”. Its semantics is given by the following clause:

$$\mathcal{M}, w \models n:\varphi \quad \text{iff} \quad \mathcal{M}, v \models \varphi \text{ where } V(n) = \{v\}.$$

Observe that if the language has the E operator and nominals, then $n:\varphi$ is definable as $E(n \wedge \varphi)$.

Finally, consider the *down-arrow binder*, written \downarrow . Let the valuation V_n^w be defined by $V_n^w(n) = \{w\}$ and $V_n^w(m) = V(m)$, when $n \neq m$. The semantic condition for \downarrow is the following:

$$\langle W, R, V \rangle, w \models \downarrow n.\varphi \quad \text{iff} \quad \langle W, R, V_n^w \rangle, w \models \varphi.$$

The language $\mathcal{HL}(\cdot, \downarrow)$ is a reduction class of first-order logic, and it is, hence, undecidable [17, 36]. It remains undecidable even with a single accessibility relation, no satisfaction operators, and only nominal propositional symbols [2]. $\mathcal{HL}(E, \downarrow)$ is equivalent to first-order logic, since \downarrow can define the operators \exists and \forall when combined with E and A .

HTab [27] is a theorem prover and model builder for hybrid logic. It handles the hybrid logic $\mathcal{HL}(E, \downarrow)$ and guarantees termination of any input formula that lacks the \downarrow binder. It is based on a tableaux calculus and it can output a model when the input formula is satisfiable.

The logic $\mathcal{HL}(E, \downarrow)$ is not able to modify the accessibility relation of a model. However, it can use the binder to name states and, hence, it can refer to *specific edges* in the model. This will be exploited by the translations introduced in the next section.

3.2 Simulating Relation-Changing Operators with Hybrid Logics

Relation-changing logics and hybrid logics with the \downarrow binder are two families of logics that are dynamic in their own way. The dynamicity of RC logics is

quite obvious: they are able to modify the accessibility relation in a model in an explicit way. On the other hand, hybrid logics carefully move nominals around, avoiding to touch anything else in the model. If we consider both formalisms, it would seem that hybrid logics are the gentler and weaker of both. But this is not true. Hybrid logics have the advantage of surgical precision over RC logics. Being able to name states of the model turns out to be a crucial advantage. As we will see, it is possible to manipulate *edges* by naming pairs of states using the pattern $\downarrow x.\diamond\downarrow y.\varphi$. We use this naming technique to simulate edge deletion, addition, and swapping.

Our translations are parametrized over a set of pairs of nominals $S \subseteq \text{NOM} \times \text{NOM}$. For a given RC formula φ , we write its translation as a hybrid formula $(\varphi)'_S$. When translating a formula, S will originally be empty and, during the translation, it will store pairs of nominals that we will use to simulate edges affected by the relation-changing operators we encounter.

Intuitively, we simulate updates by recording possible affected edges using nominals and \downarrow . As a result, in all the relation-changing logics we will consider, the RC formula $\diamond\psi$ cannot be simply translated into a hybrid formula $\diamond(\psi)'_S$, even though we have \diamond at our disposition in the hybrid language, because in the source language \diamond is interpreted over the updated accessibility relation. Instead, diamond-formulas need to be translated in a way that takes into account the edges that should be considered as deleted, added, or swapped. This is why the translation of diamond-formulas involve the \diamond operator mixed with specific considerations about the set of altered edges S .

Consider Sabotage Logic with either the local or global operator. We use the set $S \subseteq \text{NOM} \times \text{NOM}$ to represent sabotaged edges, i.e., edges that have been deleted in a given updated model.

Definition 4 (Sabotage to Hybrid Logic). Let $S \subseteq \text{NOM} \times \text{NOM}$. Define the translation $(\)'_S$ from formulas of $\mathcal{ML}(\langle \text{sb} \rangle, \langle \text{gsb} \rangle)$ to formulas of $\mathcal{HL}(\text{E}, \downarrow)$ as:

$$\begin{aligned} (p)'_S &= p \\ (\neg\varphi)'_S &= \neg(\varphi)'_S \\ (\varphi \wedge \psi)'_S &= (\varphi)'_S \wedge (\psi)'_S \\ (\diamond\varphi)'_S &= \downarrow n.\diamond(\neg\text{belongs}(n, S) \wedge (\varphi)'_S) \\ (\langle \text{sb} \rangle\varphi)'_S &= \downarrow n.\diamond(\neg\text{belongs}(n, S) \wedge \downarrow m.(\varphi)'_{S \cup nm}) \\ (\langle \text{gsb} \rangle\varphi)'_S &= \downarrow k.\text{E}\downarrow n.\diamond(\neg\text{belongs}(n, S) \wedge \downarrow m.k:(\varphi)'_{S \cup nm}) \end{aligned}$$

where n , m and k are nominals that do not appear in S , and:

$$\text{belongs}(n, S) = \bigvee_{xy \in S} (y \wedge n:x).$$

Some explanations are in order to understand the translation. First, given some model $\mathcal{M} = \langle W, R, V \rangle$ and some set $S \subseteq \text{NOM} \times \text{NOM}$, the formula $\downarrow n.\diamond(\neg\text{belongs}(n, S))$ is true at some state $w \in W$ if there exists some state v such that $(w, v) \in R$ and there is no pair of nominals $(x, y) \in S$ such that $(V(x), V(y)) = (w, v)$. Observe that the cases for $\langle \text{sb} \rangle$ and $\langle \text{gsb} \rangle$ modify the set of deleted pairs in the recursive call to the translation by adding an edge named nm . In the $\langle \text{sb} \rangle$ case, n names the evaluation state of the formula, while in the $\langle \text{gsb} \rangle$ case, n names some state anywhere in the model.

We will now prove that the translation preserves equivalence. We start by introducing some preliminary notions and definitions.

First, notice that all nominals used in the translation are bound exactly once. We can, then, define the following unequivocal notation: let $S \subseteq \text{NOM} \times \text{NOM}$, we define $\bar{S} = \{(\bar{x}, \bar{y}) \mid (x, y) \in S\}$, where \bar{n} is the state named by the nominal $n \in \text{NOM}$ under the current valuation of a model.

Second, when considering the truth of a translated formula $(\varphi)'_S$ in some model $\mathcal{M} = \langle W, R, V \rangle$, one question that may arise is what should be the initial valuation of the nominals that appear in $(\varphi)'_S$. Because all nominals in $(\varphi)'_S$ are bound by \downarrow , the truth value of $(\varphi)'_S$ does not depend on their initial valuation. Even if these symbols are not treated as nominals in the original model \mathcal{M} they will be interpreted correctly when evaluating $(\varphi)'_S$. This enables us to talk about equivalence preservation of the translation over the same model \mathcal{M} .

Theorem 5. For $\mathcal{M} = \langle W, R, V \rangle$ a model, $w \in W$, and $\varphi \in \mathcal{ML}(\langle \text{sb} \rangle, \langle \text{gsb} \rangle)$ we have:

$$\mathcal{M}, w \models \varphi \text{ iff } \mathcal{M}, w \models (\varphi)'_{\emptyset}.$$

Proof. We use structural induction on the relation-changing formula, the inductive hypothesis being:

$$\mathcal{M}_{\bar{S}}^-, w \models \varphi \text{ iff } \langle W, R, V' \rangle, w \models (\varphi)'_S$$

with $S \subseteq \text{NOM} \times \text{NOM}$, and V' is exactly as V except that for all $(x, y) \in S$, there are $v, u \in W$ such that $V'(x) = v$ and $V'(y) = u$. Boolean cases are straightforward, so we only prove the non-trivial inductive cases.

$\varphi = \diamond\psi$: For the left to right direction, suppose $\mathcal{M}_{\bar{S}}^-, w \models \diamond\psi$. Then there is some $v \in W$ such that $(w, v) \in R_{\bar{S}}^-$ and $\mathcal{M}_{\bar{S}}^-, v \models \psi$. Because $(w, v) \notin \bar{S}$, then there is no $(x, y) \in S$ such that $(\bar{x}, \bar{y}) = (w, v)$. By inductive hypothesis, we have $\mathcal{M}, v \models (\psi)'_S$, and because we can name w with a fresh nominal n , we obtain $\langle W, R, V_n^w \rangle, v \models \neg\text{belongs}(n, S) \wedge (\psi)'_S$. Therefore, we have $\mathcal{M}, w \models \downarrow n. \diamond(\neg\text{belongs}(n, S) \wedge (\psi)'_S)$, and as a consequence we get $\mathcal{M}, w \models (\psi)'_S$.

For the other direction, suppose $\mathcal{M}, w \models (\psi)'_S$. Then we have $\mathcal{M}, w \models \downarrow n. \diamond(\neg\text{belongs}(n, S) \wedge (\psi)'_S)$. Then, $\langle W, R, V_n^w \rangle, w \models \diamond(\neg\text{belongs}(n, S) \wedge (\psi)'_S)$, and, by definition, there is some $v \in W$ such that $(w, v) \in R$, $\langle W, R, V_n^w \rangle, v \models \neg\text{belongs}(n, S)$ and $\langle W, R, V_n^w \rangle, v \models (\psi)'_S$. Because we have $\neg\text{belongs}(n, S)$, there is no $(x, y) \in S$ such that $(\bar{x}, \bar{y}) = (w, v)$, which implies $(w, v) \in R$ if and only if $(w, v) \in R_{\bar{S}}^-$. On the other hand, by inductive hypothesis we have $\mathcal{M}_{\bar{S}}^-, v \models \psi$, then we have $\mathcal{M}_{\bar{S}}^-, w \models \diamond\psi$.

$\varphi = \langle \text{sb} \rangle \psi$: For the left to right direction, suppose $\mathcal{M}_{\bar{S}}^-, w \models \langle \text{sb} \rangle \psi$. Then there is some $v \in W$ such that $(w, v) \in R_{\bar{S}}^-$ and $(\mathcal{M}_{\bar{S}}^-)_{wv}^-, v \models \psi$. This is equivalent to say $\mathcal{M}_{\bar{S} \cup wv}^-, v \models \psi$. Because $(w, v) \notin \bar{S}$, then there is no $(x, y) \in S$ such that $(\bar{x}, \bar{y}) = (w, v)$ (\otimes). By inductive hypothesis we have $\langle W, R, ((V')_n^w)_m^v \rangle, v \models (\psi)'_{S \cup nm}$, where V' is exactly as V but it binds all the nominals which appear in S . By definition, we get $\langle W, R, (V')_n^w \rangle, v \models \downarrow m. (\psi)'_{S \cup nm}$, and by (\otimes) we have $\langle W, R, (V')_n^w \rangle, v \models \neg\text{belongs}(n, S) \wedge \downarrow m. (\psi)'_{S \cup nm}$. Then (by definition) $\langle W, R, V' \rangle, v \models \downarrow n. \diamond(\neg\text{belongs}(n, S) \wedge \downarrow m. (\psi)'_{S \cup nm})$, and, as a consequence, we have $\langle W, R, V' \rangle, v \models (\varphi)'_S$.

For the other direction, suppose $\langle W, R, V' \rangle, w \models (\psi)'_S$, i.e., $\langle W, R, V' \rangle, w \models \downarrow n. \diamond(\neg\text{belongs}(n, S) \wedge \downarrow m. (\psi)'_{S \cup nm})$, where V' is exactly as V but it binds all the nominals which appear in S . Then, we have $\langle W, R, (V')_n^w \rangle, w \models \diamond(\neg\text{belongs}(n, S) \wedge \downarrow m. (\psi)'_{S \cup nm})$, and, by definition, there is some $v \in W$ such that $(w, v) \in$

$R, \langle W, R, V_n^w \rangle, v \models \neg \text{belongs}(n, S)$ and $\langle W, R, V_n^w \rangle, v \models \downarrow m.(\psi)'_{S \cup nm}$. Then, $\langle W, R, ((V')_n^w)_m^v \rangle, v \models (\psi)'_{S \cup nm}$. Because we have $\neg \text{belongs}(n, S)$, there is no $(x, y) \in S$ such that $(\bar{x}, \bar{y}) = (w, v)$, which implies $(w, v) \in R$ if and only if $(w, v) \in R_S^-$. On the other hand, by inductive hypothesis we have $\mathcal{M}_{S \cup vv}^-, v \models \psi$, and thus we have $\mathcal{M}_{S^-}, w \models \langle \text{sb} \rangle \psi$.

$\varphi = \langle \text{gsb} \rangle \psi$: this case is similar to the previous one. \square

For Bridge Logic, we use the set $B \subseteq \text{NOM} \times \text{NOM}$ to represent new edges added by the dynamic operator. The translation of \diamond should be able to consider edges in B . This explains why the translation of \diamond is a disjunction: either we traverse an edge that is in the original model or an edge in B .

Definition 6 (Bridge to Hybrid Logic). Let $B \subseteq \text{NOM} \times \text{NOM}$. We define the translation $(\)'_B$ from formulas of $\mathcal{ML}(\langle \text{br} \rangle, \langle \text{gbr} \rangle)$ to formulas of $\mathcal{HL}(\text{E}, \downarrow)$ as:

$$\begin{aligned} (p)'_B &= p \\ (\neg \varphi)'_B &= \neg(\varphi)'_B \\ (\varphi \wedge \psi)'_B &= (\varphi)'_B \wedge (\psi)'_B \\ (\diamond \varphi)'_B &= \downarrow n. \text{E} \downarrow m. (n: \diamond m \vee \text{belongs}(n, B)) \wedge (\varphi)'_B \\ (\langle \text{br} \rangle \varphi)'_B &= \downarrow n. \text{E} \downarrow m. (\neg n: \diamond m \wedge \neg \text{belongs}(n, B) \wedge (\varphi)'_{B \cup nm}) \\ (\langle \text{gbr} \rangle \varphi)'_B &= \downarrow k. \text{E} \downarrow n. \text{E} \downarrow m. (\neg n: \diamond m \wedge \neg \text{belongs}(n, B) \wedge k: (\varphi)'_{B \cup nm}) \end{aligned}$$

where n, m and k are nominals that do not appear in B , and belongs is defined as in Definition 4.

Theorem 7. For $\mathcal{M} = \langle W, R, V \rangle$ a model, $w \in W$, and $\varphi \in \mathcal{ML}(\langle \text{br} \rangle, \langle \text{gbr} \rangle)$, we have:

$$\mathcal{M}, w \models \varphi \text{ iff } \mathcal{M}, w \models (\varphi)'_{\emptyset}.$$

Proof. A similar reasoning to the proof of Sabotage Logic can be done with the following inductive hypothesis:

$$\mathcal{M}_B^+, w \models \varphi \text{ iff } \langle W, R, V' \rangle, w \models (\varphi)'_B$$

with $B \subseteq \text{NOM} \times \text{NOM}$, and V' is exactly as V except that for all $(x, y) \in B$, there are $v, u \in W$ such that $V'(x) = v$ and $V'(y) = u$. \square

We finish with the case of Swap Logic. A different translation, for the local case only, is given in [5]. As we did for Sabotage Logic, we use $S \subseteq \text{NOM} \times \text{NOM}$ to represent the set of deleted edges, i.e., the edges that should not be possible to traverse in a given updated model. Indeed, swapping a non-reflexive edge of a model has the effect of deleting it, along with adding its inverse. This implies that S^{-1} is a set of edges that we can currently traverse.

To ensure this, the translation treats $\langle \text{sw} \rangle$ and $\langle \text{gsw} \rangle$ carefully. Three cases should be taken into account. The first one is when a reflexive edge is swapped either locally or globally. In this case, the translation continues with the set S unchanged, but it ensures the presence of a reflexive edge: at the current state for $\langle \text{sw} \rangle$ with $\downarrow n. \diamond n$; or anywhere in the model for $\langle \text{gsw} \rangle$ with $\text{E} \downarrow n. \diamond n$.

The second case is when an irreflexive edge that has never been swapped before is swapped. Here we need to ensure that the edge is present in the model, that it is irreflexive, and that neither this edge nor its inverse is in S . We then add the nominals that name it to S before moving on with the translation.

The last case is when we traverse an already swapped edge. That is, for some $xy \in S$, we traverse the edge referred to by the nominals yx . In this case, we do not need to require the presence of any new edge in the model. We assume to be standing at the state named by y and that the rest of the formula is satisfied at x , but we remove xy from S and add yx . Why not just remove xy from the set S since swapping some edge twice just makes it return to its configuration in the original model? This is not always the case: if some edge *and* its symmetric are both present in the initial model, the action of swapping one of them twice is not supposed to restore its symmetric. yx is added to S to ensure that the symmetric edge is no longer present.

Definition 8 (Swap to Hybrid Logic). Let $S \subseteq \text{NOM} \times \text{NOM}$. We define the translation $(\)'_S$ from formulas of $\mathcal{ML}(\langle \text{sw} \rangle, \langle \text{gsw} \rangle)$ to formulas of $\mathcal{HL}(\text{E}, \downarrow)$ as:

$$\begin{aligned}
(p)'_S &= p \\
(\neg\varphi)'_S &= \neg(\varphi)'_S \\
(\varphi \wedge \psi)'_S &= (\varphi)'_S \wedge (\psi)'_S \\
(\diamond\varphi)'_S &= (\downarrow n. \diamond(\neg\text{belongs}(n, S) \wedge (\varphi)'_S)) \vee \text{isSat}(S^{-1}, (\varphi)'_S) \\
(\langle \text{sw} \rangle \varphi)'_S &= (\downarrow n. \diamond n \wedge (\varphi)'_S) \\
&\quad \vee \downarrow n. \diamond(\neg n \wedge \neg\text{belongs}(n, S \cup S^{-1}) \wedge \downarrow m. (\varphi)'_{S \cup nm}) \\
&\quad \vee \bigvee_{xy \in S} (y \wedge x: (\varphi)'_{(S \setminus xy) \cup yx}) \\
(\langle \text{gsw} \rangle \varphi)'_S &= (\text{E} \downarrow n. \diamond n \wedge (\varphi)'_S) \\
&\quad \vee \downarrow k. \text{E} \downarrow n. \diamond(\neg n \wedge \neg\text{belongs}(n, S \cup S^{-1}) \wedge \downarrow m. k: (\varphi)'_{S \cup nm}) \\
&\quad \vee \bigvee_{xy \in S} (\varphi)'_{(S \setminus xy) \cup yx}
\end{aligned}$$

where n, m and k are nominals that do not appear in S , belongs is defined as in Definition 4, and

$$\text{isSat}(S, \varphi) = \bigvee_{xy \in S} (x \wedge y: \varphi).$$

The formula $\text{isSat}(S, (\varphi)'_S)$ says that the translation of φ is satisfiable at the end of some of the edges belonging to S . Note that the translation maps formulas of $\mathcal{ML}(\langle \text{sw} \rangle)$ to the less expressive $\mathcal{HL}(\cdot, \downarrow)$, i.e., the E operator is not required.

Theorem 9. For $\mathcal{M} = \langle W, R, V \rangle$ a model, $w \in W$ and $\varphi \in \mathcal{ML}(\langle \text{sw} \rangle, \langle \text{gsw} \rangle)$ we have:

$$\mathcal{M}, w \models \varphi \text{ iff } \mathcal{M}, w \models (\varphi)'_{\emptyset}.$$

Proof. Again, a similar reasoning to the proof of Sabotage Logic can be done with the following inductive hypothesis:

$$\mathcal{M}_{S^{-1}}^*, w \models \varphi \text{ iff } \langle W, R, V' \rangle, w \models (\varphi)'_S$$

with $S \subseteq \text{NOM} \times \text{NOM}$, and V' is exactly as V except that for all $(x, y) \in S$, there are $v, u \in W$ such that $V'(x) = v$ and $V'(y) = u$. \square

3.3 Decidable Fragments

Interesting decidable fragments of hybrid logics with binders have been found over time. Such decidable fragments are convenient for our relation-changing

logics in the light of the (computable) translations presented in Section 3. First, let us consider restricting the satisfiability problem over certain classes of models. The following logics are known to be decidable over the indicated classes:

- $\mathcal{H}\mathcal{L}(\mathbb{E}, \downarrow)$ over linear frames (i.e., irreflexive, transitive, and trichotomous frames [21, 34], this includes $(\mathbb{N}, <)$),
- $\mathcal{H}\mathcal{L}(\mathbb{E}, \downarrow)$ over models with a single, transitive tree relation [34],
- $\mathcal{H}\mathcal{L}(\mathbb{E}, \downarrow)$ over models with a single, $S5$, or complete relation [34],
- $\mathcal{H}\mathcal{L}(:, \downarrow)$ over models with a single relation of bounded finite width [37]; as a corollary, also over finite models.

Since the translations preserve equivalence, we get:

Corollary 10. *The satisfiability problem for all relation-changing modal logics over linear, transitive trees, $S5$, and complete frames is decidable.*

Corollary 11. *The satisfiability problem for local sabotage and local swap logics over models of bounded width is decidable.*

Curiously, these results mean that relation-changing modal logics are decidable over certain classes of models, even if the modifications implied by evaluating RC formulas yield models that *do not* belong to such class. For instance, these two facts are simultaneously true: sabotage logic is decidable on the class of $S5$ models, and deleting edges in an $S5$ model can yield a non- $S5$ model.

Now, let us turn to syntactical definitions of decidable fragments. We recall that local sabotage and local swap can be translated to $\mathcal{H}\mathcal{L}(:, \downarrow)$. Consider formulas of $\mathcal{H}\mathcal{L}(:, \downarrow)$ in negation normal form. $\mathcal{H}\mathcal{L}(:, \downarrow) \setminus \square \downarrow \square$ is the fragment obtained by removing formulas that contain a nesting of \square , \downarrow and again \square . This fragment is decidable [37].

Our translations use the \downarrow binder in many places, but we can make them a little more economical in that sense, at the expense of losing succinctness.

Take the following case for $\mathcal{M}\mathcal{L}(\langle \text{sb} \rangle)$:

$$(\diamond\varphi)'_S = \downarrow n. \diamond(\neg \text{belongs}(n, S) \wedge (\varphi)'_S).$$

Instead of using the down-arrow binder and later ensuring that we did not take a deleted edge by using $\neg \text{belongs}(n, S)$, we can do the following. For all pairs of nominals $(x, y) \in S$, the current state w satisfies one combination of the truth values of the nominals x . Let X be the set of true nominals x at w . Then, $(\varphi)'_S$ should be true at some accessible state v that should not satisfy any of the corresponding y nominals for all $x \in X$.

Then, the translation becomes:

$$(\diamond\varphi)'_S = \bigvee_{X \subseteq \text{fst}(S)} \left(\bigwedge_{x \in X} x \wedge \bigwedge_{x \notin X} \neg x \wedge \diamond \left(\bigwedge_{y \in \text{snd}(S, X)} \neg y \wedge (\varphi)'_S \right) \right)$$

where $\text{fst}(S) = \{x \mid (x, y) \in S\}$ and $\text{snd}(S, X) = \{y \mid (x, y) \in S, x \in X\}$.

In the case of $\mathcal{M}\mathcal{L}(\langle \text{sw} \rangle)$ we can do the same. For $\langle \text{sw} \rangle$, \diamond was translated as

$$(\diamond\varphi)'_S = (\downarrow n. \diamond(\neg \text{belongs}(n, S) \wedge (\varphi)'_S)) \vee \text{isSat}(S^{-1}, (\varphi)'_S).$$

Here the $\text{isSat}(S^{-1}, (\varphi)'_S)$ disjunct does not use the \downarrow binder, while the first disjunct is similar to the case of local sabotage, and can be replaced by:

$$(\diamond\varphi)'_S = \bigvee_{X \subseteq \text{fst}(S)} \left(\bigwedge_{x \in X} x \wedge \bigwedge_{x \notin X} \neg x \wedge \diamond \left(\bigwedge_{y \in \text{snd}(S, X)} \neg y \wedge (\varphi)'_S \right) \right) \vee \text{isSat}(S^{-1}, (\varphi)'_S).$$

Let \blacklozenge be either $\langle \text{sb} \rangle$ or $\langle \text{sw} \rangle$ and \blacksquare be either $[\text{sb}]$ or $[\text{sw}]$. The following patterns in RC formulas result in the shown patterns in the hybrid formula obtained by the translations:

RC pattern	Produced pattern
\square	\square
\blacklozenge	\downarrow
\blacksquare	$\downarrow \square \downarrow$

By considering these new versions of the translations, and by taking into account the syntactic decidable fragment of $\mathcal{HL}(\cdot, \downarrow)$ mentioned above, we can establish the following result:

Corollary 12. *The following fragments are decidable on the class of all relational models:*

- $\mathcal{ML}(\langle \text{sb} \rangle) \setminus \{\blacksquare\blacksquare, \blacksquare\square, \square\blacksquare, \blacksquare\blacklozenge\blacksquare\}$
- $\mathcal{ML}(\langle \text{sw} \rangle) \setminus \{\blacksquare\blacksquare, \blacksquare\square, \square\blacksquare, \blacksquare\blacklozenge\blacksquare\}$

where \blacksquare is either \square or \blacksquare .

3.4 Comparing Expressive Power

We have introduced translations for the six relation-changing modal logics from Section 2 into hybrid logic. In some cases (for the local version of swap and sabotage), the obtained formulas fall into the fragment $\mathcal{HL}(\cdot, \downarrow)$. On the other hand, for encoding the rest of the logics we need also to use the universal modality E. An interesting question is whether we can obtain translations from hybrid to relation-changing logics, i.e., if some of the relation-changing logics considered in this article are as expressive as some hybrid logic. Let us define first, the expressive power comparisons we will use.

Definition 13 ($\mathcal{L} \leq \mathcal{L}'$). We say that \mathcal{L}' is *at least as expressive as* \mathcal{L} (notation $\mathcal{L} \leq \mathcal{L}'$) if there is a function Tr between formulas of \mathcal{L} and \mathcal{L}' such that for every model \mathcal{M} and every formula φ of \mathcal{L} we have that

$$\mathcal{M} \models_{\mathcal{L}} \varphi \text{ iff } \mathcal{M} \models_{\mathcal{L}'} \text{Tr}(\varphi).$$

\mathcal{M} is seen as a model of \mathcal{L} on the left and as a model of \mathcal{L}' on the right, and we use in each case the appropriate semantic relation $\models_{\mathcal{L}}$ or $\models_{\mathcal{L}'}$ as required.

\mathcal{L}' is strictly more expressive than \mathcal{L} ($\mathcal{L} < \mathcal{L}'$) if $\mathcal{L} \leq \mathcal{L}'$ but not $\mathcal{L}' \leq \mathcal{L}$. Finally, we say that \mathcal{L} and \mathcal{L}' are *incomparable* if $\mathcal{L} \not\leq \mathcal{L}'$ and $\mathcal{L}' \not\leq \mathcal{L}$.

In [3, 5, 19, 6] we discussed the expressive power of relation-changing modal logics by introducing their corresponding notions of bisimulations and using them to compare the logics among each other. We concluded that they are all

incomparable in expressive power.¹ As a consequence, we conclude that it is not possible that two of them capture the same fragment of hybrid logic. In fact, we will prove that all the relation-changing logics considered here are strictly less expressive than the corresponding hybrid logic in which they are translated.

Theorem 14. *Let $\blacklozenge_1 \in \{\langle \text{sb} \rangle, \langle \text{sw} \rangle\}$, we have $\mathcal{ML}(\blacklozenge_1) < \mathcal{HL}(\cdot, \downarrow)$. For $\blacklozenge_2 \in \{\langle \text{gsb} \rangle, \langle \text{gsw} \rangle, \langle \text{br} \rangle, \langle \text{gbr} \rangle\}$, we have $\mathcal{ML}(\blacklozenge_2) < \mathcal{HL}(\text{E}, \downarrow)$.*

Proof. For any of the logics mentioned above, we have translations into the corresponding hybrid logic. Now we need to prove that these translations do not cover their entire target language (modulo equivalence). In order to do that, we provide bisimilar models for relation-changing modal logics which can be distinguished by some hybrid formula. In Figure 1, we show two pairs of models already introduced in [6] that cover all possibilities of bisimilarity.



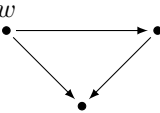

\mathcal{M}, w	\mathcal{M}', w'	Bisimilar for
		$\mathcal{ML}(\langle \text{sw} \rangle)$ $\mathcal{ML}(\langle \text{br} \rangle)$ $\mathcal{ML}(\langle \text{gsw} \rangle)$ $\mathcal{ML}(\langle \text{gbr} \rangle)$
		$\mathcal{ML}(\langle \text{sb} \rangle)$ $\mathcal{ML}(\langle \text{gsb} \rangle)$

Figure 1: Bisimilar models

The two models in the first row can be distinguished by the formula $\downarrow n. \Box n$, which establishes that the only successor of the evaluation point is itself. This formula is true at \mathcal{M}, w and false at \mathcal{M}', w' . Models in the second row can be distinguished by the formula $\downarrow n. \Diamond \downarrow m. n: \Diamond \Diamond m$, which says that from the evaluation point it is possible to arrive to the same state in one or two steps. This is true at \mathcal{M}, w but false at \mathcal{M}', w' . \square

Notice that both hybrid formulas we introduced above belong to the fragment $\mathcal{HL}(\cdot, \downarrow)$, i.e., it was not necessary to use the E operator. This means that even though we use E in some of the translations (and we strongly believe that it is essential for some encodings) there are fragments of $\mathcal{HL}(\cdot, \downarrow)$ that cannot be captured by relation-changing modal logics.

4 Undecidability of Relation-Changing Logics

As we mentioned, we will prove that the satisfiability problem of the relation-changing logics introduced are undecidable by reductions from memory logics.

¹Except for the local and global swap operators, which is still open in one direction.

4.1 Memory Logic

Memory logics [1, 32] are modal logics that can *store* the current state of evaluation into a memory and *check* whether the current state belongs to this memory. Its syntax and semantics are extensions of the syntax and semantics of the basic modal logic \mathcal{ML} . The memory is a subset of the domain of the model. $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ is the memory logic that extends \mathcal{ML} with the operators \mathbb{R} and \mathbb{K} , which stand for “remember” and “known”, respectively.

Definition 15 (Syntax). Let PROP be a countable, infinite set of propositional symbols. The set FORM of formulas over PROP is defined as:

$$\text{FORM} ::= p \mid \mathbb{K} \mid \neg\varphi \mid \varphi \wedge \psi \mid \diamond\varphi \mid \mathbb{R}\varphi,$$

where $p \in \text{PROP}$ and $\varphi, \psi \in \text{FORM}$. Other Boolean and modal operators are defined as usual.

Definition 16 (Semantics). A model $\mathcal{M} = \langle W, R, V, S \rangle$ is a relational model equipped with a set $S \subseteq W$ called the *memory*. Let w be a state in W . The inductive definition of satisfiability for the cases specific to memory logic is:

$$\begin{aligned} \langle W, R, V, S \rangle, w \models \mathbb{R}\varphi & \text{ iff } \langle W, R, V, S \cup \{w\} \rangle, w \models \varphi \\ \langle W, R, V, S \rangle, w \models \mathbb{K} & \text{ iff } w \in S. \end{aligned}$$

The remaining cases coincide with the semantics of \mathcal{ML} , and do not involve the memory.

An $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ -formula φ is *satisfiable* if there are a model $\mathcal{M} = \langle W, R, V, \emptyset \rangle$ and $w \in W$ such that $\mathcal{M}, w \models \varphi$. The empty initial memory ensures that no state of the model satisfies the unary predicate \mathbb{K} unless a formula $\mathbb{R}\psi$ has previously been evaluated there.

We will show that relation-changing operators simulate the remember and known operators. However, there is one subtle difference between the \mathbb{R} operator and relation-changing operators like $\langle \text{sb} \rangle$. While evaluating $\langle \text{sb} \rangle\varphi$ always results in a change in the model, $\mathbb{R}\varphi$ can leave the memory unchanged if the current state of evaluation is already memorized. We can ignore this difference by observing that any $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ -formula can be rewritten into an equivalent formula where every occurrence of \mathbb{R} is “proper,” in the sense that it actually modifies the memory.

Definition 17 (PNF). An $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ -formula φ is in *proper normal form* (PNF) if every occurrence of the \mathbb{R} operator in φ occurs as $(\neg\mathbb{K} \wedge \mathbb{R}\psi) \vee (\mathbb{K} \wedge \psi)$.

In the next section we assume that memory logic formulas are always in PNF. This is important for structural inductive proofs.

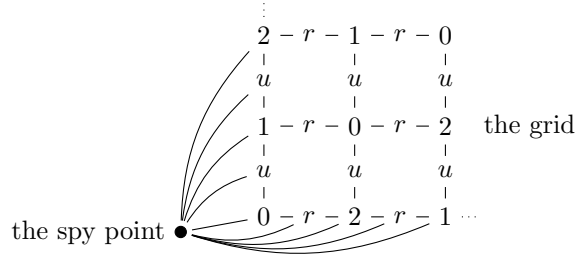
Finally, we define the notion of *modal depth* of an $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ -formula.

Definition 18. Given φ in $\mathcal{ML}(\mathbb{R}, \mathbb{K})$, we define the *modal depth* of φ (notation $\text{md}(\varphi)$) as

$$\begin{aligned} \text{md}(\mathbb{K}) &= 0 \\ \text{md}(p) &= 0 \text{ for } p \in \text{PROP} \\ \text{md}(\mathbb{R}\varphi) &= \text{md}(\varphi) \\ \text{md}(\neg\varphi) &= \text{md}(\varphi) \\ \text{md}(\varphi \wedge \psi) &= \max\{\text{md}(\varphi), \text{md}(\psi)\} \\ \text{md}(\diamond\varphi) &= 1 + \text{md}(\varphi). \end{aligned}$$

Multimodal memory logic is shown to be undecidable in [9]. We strengthen this result, showing that undecidability holds also in the monomodal case. We adapt the encoding of [31]: the $\mathbb{N} \times \mathbb{N}$ -tiling problem (i.e., the problem of tiling the plane given a set of tile types [40]) is encoded into $\mathcal{ML}(\mathbb{R}, \mathbb{K})$.

We define an $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ -formula that is satisfiable only in models with a *spy state* s (i.e., a state that has direct access to all other states in the model), where the formula is evaluated, and *grid states* that will be used for the tiling. We show below the intended shape of the model, where the following notation is used: the symbols $0, 1, 2, r$ and u all represent grid states, all shown edges are symmetric, and the spy state is connected to all grid states (i.e., not all edges from spy to grid are drawn).



Let us enumerate the conjuncts of the formula that encodes a given instance of the tiling problem. The first conjunct specifies that the spy state and all its successors in one steps are irreflexive. The second enforces symmetry of outgoing edges from all the successors of the spy state:

1. $\mathbb{R}\Box\neg\mathbb{K} \wedge \Box\mathbb{R}\Box\neg\mathbb{K}$
2. $\Box\mathbb{R}\Box\Diamond\mathbb{K}$

The following conjunct ensures the spy state see every grid state:

3. $\Box\Box\mathbb{R}\Box\Diamond\mathbb{K} \wedge \mathbb{R}\Box\Box(\neg\mathbb{K} \rightarrow \Diamond\mathbb{K})$

More formally, we have the following lemma:

Lemma 19. *Let $\mathcal{M} = \langle W, R, V, \emptyset \rangle$ be a model and $w \in W$, such that $\mathcal{M}, w \models \Box\Box\mathbb{R}\Box\Diamond\mathbb{K} \wedge \mathbb{R}\Box\Box(\neg\mathbb{K} \rightarrow \Diamond\mathbb{K})$. Then every state accessible in two steps from w is also accessible in a single step.*

Proof. Since $\mathcal{M}, w \models \Box\Box\mathbb{R}\Box\Diamond\mathbb{K}$ holds, for every state v accessible in two steps from w , for all $u \in W$, if $(v, u) \in R$ then $(u, v) \in R$. Moreover, since $\mathcal{M}, w \models \mathbb{R}\Box\Box(\neg\mathbb{K} \rightarrow \Diamond\mathbb{K})$, for every state v accessible in two steps from w , we have $(v, w) \in R$. This implies that for every state v accessible in two steps from w , $(w, v) \in R$. \square

We now combine a single symmetric relation with propositional symbols to encode the relations R_{up} and R_{right} in the grid. We use the propositional symbols $0, 1, 2, u$, and r , and the notations $i, s(i)$ and $p(i)$, with $i \in \{0, 1, 2\}$; $s(i) = (i + 1) \bmod 3$; and $p(i) = (i + 2) \bmod 3$. More precisely, given a model $\mathcal{M} = \langle W, R, V, S \rangle$, we define the relations R_{up} and R_{right} as follows:

- $R_{up} = \{(x, y) \mid \exists i \in \{0, 1, 2\}, \mathcal{M}, x \models i, \mathcal{M}, y \models s(i), \exists z \in W, \mathcal{M}, z \models u, \{(x, z), (z, y)\} \subseteq R\}$
- $R_{right} = \{(x, y) \mid \exists i \in \{0, 1, 2\}, \mathcal{M}, x \models i, \mathcal{M}, y \models p(i), \exists z \in W, \mathcal{M}, z \models r, \{(x, z), (z, y)\} \subseteq R\}$

We ensure that exactly one of the propositional symbols used to define these relations holds at every state of the grid. Given some set $P \subset \text{PROP}$, we use the notation $\text{one}(P) = \bigvee_{p \in P} (p \wedge \bigwedge_{q \in P \setminus \{p\}} \neg q)$ to define:

$$4. \quad \Box(\text{one}(\{0, 1, 2, u, r\}))$$

We now specify that every grid state has a successor through the relations R_{up} and R_{right} :

$$5. \quad \bigwedge_{i \in \{0, 1, 2\}} \Box(i \rightarrow \Diamond(u \wedge \Diamond s(i)))$$

$$6. \quad \bigwedge_{i \in \{0, 1, 2\}} \Box(i \rightarrow \Diamond(r \wedge \Diamond p(i)))$$

The relation R_{up} and its inverse are functional:

$$7. \quad \bigwedge_{i \in \{0, 1, 2\}} \Box(s(i) \rightarrow \textcircled{F}\Box(u \rightarrow \Box(i \rightarrow \Box(u \rightarrow \Box(s(i) \rightarrow \textcircled{K}))))))$$

$$8. \quad \bigwedge_{i \in \{0, 1, 2\}} \Box(i \rightarrow \textcircled{F}\Box(u \rightarrow \Box(s(i) \rightarrow \Box(u \rightarrow \Box(i \rightarrow \textcircled{K}))))))$$

The relation R_{right} and its inverse are functional:

$$9. \quad \bigwedge_{i \in \{0, 1, 2\}} \Box(p(i) \rightarrow \textcircled{F}\Box(r \rightarrow \Box(i \rightarrow \Box(r \rightarrow \Box(p(i) \rightarrow \textcircled{K}))))))$$

$$10. \quad \bigwedge_{i \in \{0, 1, 2\}} \Box(i \rightarrow \textcircled{F}\Box(r \rightarrow \Box(p(i) \rightarrow \Box(r \rightarrow \Box(i \rightarrow \textcircled{K}))))))$$

They are also confluent:

$$11. \quad \bigwedge_{i \in \{0, 1, 2\}} \Box(s(i) \rightarrow \textcircled{F}\Box(u \rightarrow \Box(i \rightarrow \Box(r \rightarrow \Box(p(i) \rightarrow \Diamond(u \wedge \Diamond(i \wedge \Diamond(r \wedge \Diamond s(i) \wedge \textcircled{K}))))))))))$$

Let $T = \{t_1, \dots, t_k\} \subset \text{PROP}$ a set of propositional symbols representing tile types. We write $\text{right}(t_n)$ for the set of tiles types that can be placed to the right of some tile type t_n , and $\text{top}(t_n)$ the set of tile types that can be placed above it.

At every state of the grid, one and only one tile holds, and tiles must match:

$$12. \quad \bigwedge_{i \in \{0, 1, 2\}} \Box(i \rightarrow \text{one}(T))$$

$$13. \quad \bigwedge_{i \in \{0, 1, 2\}, 1 \leq n \leq k} \Box(i \wedge t_n \rightarrow \Box(u \rightarrow \Box(s(i) \rightarrow \bigvee \text{top}(t_n))))$$

$$14. \quad \bigwedge_{i \in \{0, 1, 2\}, 1 \leq n \leq k} \Box(i \wedge t_n \rightarrow \Box(r \rightarrow \Box(p(i) \rightarrow \bigvee \text{right}(t_n))))$$

We finish by setting the spy state apart and initializing the grid:

$$15. \neg 0 \wedge \neg 1 \wedge \neg 2 \wedge \neg u \wedge \neg r \wedge \Diamond 0$$

Lemma 20. *Given a tiling problem $T = t_1 \dots t_k$, let $\text{Grid}(T)$ be the conjunction of formulas (1)–(15) above. T tiles the $\mathbb{N} \times \mathbb{N}$ -grid if and only if $\text{Grid}(T)$ is satisfiable.*

Theorem 21. *The satisfiability problem of $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ is undecidable.*

4.2 Mapping Memory Logic into Relation-Changing Logics

We now present satisfiability-preserving translations from $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ to relation-changing modal logics. Combining these translations with the undecidability result of Theorem 21, we can claim:

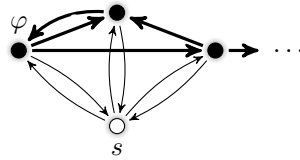
Theorem 22. *The satisfiability problem of $\mathcal{ML}(\Diamond)$ is undecidable, for $\Diamond \in \{\langle \text{sb} \rangle, \langle \text{br} \rangle, \langle \text{sw} \rangle, \langle \text{gsb} \rangle, \langle \text{gbr} \rangle, \langle \text{gsw} \rangle\}$.*

The main idea of these translations is to simulate the behavior of $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ without having an external memory in the model. We simulate the ability to store states in a memory by changing the accessibility relation of a model. Checking for membership in the memory is simulated by checking for changes in the accessibility relation.

Every translation τ_\Diamond from $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ -formulas to $\mathcal{ML}(\Diamond)$ -formulas proceeds in two steps. For a given target logic, the translation includes a fixed part called Struct_\Diamond , that enforces constraints on the structure of the model. The second part, called Tr_\Diamond , is defined inductively on $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ -formulas, and uses the structure provided by Struct_\Diamond to simulate the \mathbb{R} and \mathbb{K} operators.

4.2.1 Sabotage Logic

Local Sabotage. In the translation to local sabotage logic, the $\text{Struct}_{\langle \text{sb} \rangle}$ subformula should ensure that every state of the model can be memorized using the expressivity of $\langle \text{sb} \rangle$. This operator changes the point of evaluation after deleting an edge. To compensate for this, the $\text{Struct}_{\langle \text{sb} \rangle}$ formula guarantees that every state has an edge that is deleted when the state is memorized, and a second edge back to the original state to ensure that evaluation can continue at the correct state. We use a spy point s to ensure this structure. The idea is illustrated in the following image.



We need to ensure that every satisfiable formula of $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ is translated into a satisfiable formula (and vice-versa, if the translated formula is satisfiable, then the original formula is satisfiable, too). The image above shows an intended model for the translated formula $\tau_{\langle \text{sb} \rangle}(\varphi)$. Intuitively, bold edges and arrows correspond to the model of φ . The complete translation is given in Definition 23. Here we discuss in detail how it works.

$Struct_{\langle sb \rangle}$ adds a spy state with symmetric edges between itself and all other states. In particular, (1) in Definition 23 ensures that the evaluation state satisfies s and that it is irreflexive, and (2) guarantees that its immediate successors reach a state where s holds. Formulas (3) and (4) ensure that this state is the original s state. They work together as follows: (3) makes $\Box\Diamond s$ true in any s -state reachable in two steps, and by deleting the traversed edges we avoid a cycle of size two between this s -state and an immediate successor of the evaluation state, distinguishing the original s -state from any other s -state reachable in two steps. (4) then traverses one edge, deletes the next one, and reaches a state where s implies $\Diamond\Box\neg s$. This contradicts (3), unless we have arrived in the original s state. Formulas (5), (6) and (7) mimic (2), (3) and (4), but for edges which are removed twice. Observe that (6) now avoids a cycle of size three between any other s -state reachable in two steps and an immediate successor of the evaluation state. Finally, (8) and (9) ensure that the evaluation state is indeed a spy state, i.e., that it is linked to every other state of the input model.

$Tr_{\langle sb \rangle}$ starts by placing the translation $(\)'$ of φ in a successor of the evaluation state. Boolean cases are obvious. For the diamond case, $\Diamond\psi$ is satisfied if there is a successor v where ψ holds, but we must ensure that v is not the spy state. For $(\textcircled{R})\psi'$, we do a round-trip of sabotaging from the current state to the spy state. Note that after reaching the spy state an edge does come back to the same state where it came from, since the only accessible state where $\neg\Diamond s$ holds is the one we are memorizing. For $(\textcircled{K})'$, we check whether there is an edge pointing to some s -state.

Definition 23. Define $\tau_{\langle sb \rangle}(\varphi) = Struct_{\langle sb \rangle} \wedge Tr_{\langle sb \rangle}(\varphi)$, where:

$$\begin{aligned}
Struct_{\langle sb \rangle} = & s \wedge \Box\neg s & (1) \\
& \wedge \Box\Diamond s & (2) \\
& \wedge [\mathbf{sb}][\mathbf{sb}](s \rightarrow \Box\Diamond s) & (3) \\
& \wedge \Box[\mathbf{sb}](s \rightarrow \Diamond\Box\neg s) & (4) \\
& \wedge \Box\Box(\neg s \rightarrow \Diamond s) & (5) \\
& \wedge \Box[\mathbf{sb}](s \rightarrow [\mathbf{sb}](\Box\neg s \rightarrow \Box\Box(s \rightarrow \Box\Diamond s))) & (6) \\
& \wedge \Box[\mathbf{sb}](s \rightarrow \Box(\Box\neg s \rightarrow \Box\Box(s \rightarrow \Diamond\Box\neg s))) & (7) \\
& \wedge \Box\Box\Box(s \rightarrow \Box\Diamond s) & (8) \\
& \wedge \Box\Box[\mathbf{sb}](s \rightarrow \Diamond\Box\neg s) & (9)
\end{aligned}$$

$Tr_{\langle sb \rangle}(\varphi) = \Diamond(\varphi)'$, with:

$$\begin{aligned}
(p)' & = p \quad \text{for } p \in \text{PROP appearing in } \varphi \\
(\textcircled{K})' & = \neg\Diamond s \\
(\neg\psi)' & = \neg(\psi)' \\
(\psi \wedge \chi)' & = (\psi)' \wedge (\chi)' \\
(\Diamond\psi)' & = \Diamond(\neg s \wedge (\psi)') \\
(\textcircled{R})\psi' & = [\mathbf{sb}](s \wedge [\mathbf{sb}](\neg\Diamond s \wedge (\psi)'))
\end{aligned}$$

Proposition 24. *If $\langle W, R, V \rangle, w \models Struct_{\langle sb \rangle}$. Let W^w the connected component of W from state w . Then for every state $v \in W^w \setminus \{w\}$, there exists exactly one state v' such that $(v, v'), (v', v) \in R$ and $v' \in V(s)$.*

Lemma 25. *Let φ be an $M\mathcal{L}(\textcircled{R}, \textcircled{K})$ -formula in PNF that does not contain the propositional symbol s . Then, φ is satisfiable iff $\tau_{\langle sb \rangle}(\varphi)$ is satisfiable.*

Proof. (\Leftarrow) Suppose $\langle W, R, V \rangle, s \models \tau_{(\text{sb})}(\varphi)$. Let $W' = W \setminus V(s)$, $R' = R \cap (W' \times W')$ and $V'(p) = V(p) \cap W'$ for all $p \in \text{PROP}$. By definition of $\text{Tr}_{(\text{sb})}$ there is $w' \in W'$ such that $(s, w') \in R$ and $\langle W, R, V \rangle, w' \models (\varphi)'$.

Now, let ψ be a sub-formula of φ , $v \in W'$, $S \subseteq W'$ and $R_S = R \setminus \{(v, s), (s, v) \mid v \in S\}$. We prove by structural induction on ψ that $\langle W', R', V', S \rangle, v \models \psi$ if, and only if, $\langle W, R_S, V \rangle, v \models (\psi)'$. In particular, this will prove that $\langle W', R', V', \emptyset \rangle, w' \models \varphi$ if, and only if, $\langle W, R, V \rangle, w' \models (\varphi)'$.

We prove the case for \mathbb{K} . Suppose $\langle W', R', V', S \rangle, v \models \mathbb{K}$. By definition of \models , $v \in S$, but then $(v, s), (s, v) \notin R_S$ by definition of R_S , so by construction $\langle W, R_S, V \rangle, v \models \neg \diamond s$. Hence, $\langle W, R_S, V \rangle, v \models (\mathbb{K})'$.

The propositional, Boolean and modal cases are trivial. For $\psi = \mathbb{K}$, we should prove that $\langle W', R', V', S \rangle, v \models \mathbb{K}$ if, and only if, $\langle W, R_S, V \rangle, v \models \neg \diamond s$. However this is immediate by definition of S and R_S and Proposition 24.

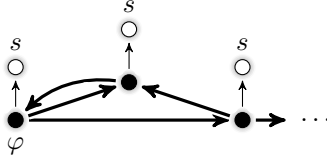
For the last case, consider $\psi = \neg \mathbb{K} \wedge \mathbb{F}\chi$ (remember that formulas are in PNP), so we should prove that $\langle W', R', V', S \rangle, v \models \neg \mathbb{K} \wedge \mathbb{F}\chi$ if, and only if, $\langle W, R_S, V \rangle, v \models \diamond s \wedge \langle \text{sb} \rangle (s \wedge \langle \text{sb} \rangle (\neg \diamond s \wedge \chi))'$. Again, the equivalence is immediate by Proposition 24.

(\Rightarrow) Suppose $\langle W, R, V, \emptyset \rangle, w \models \varphi$. We build a model for $\tau_{(\text{sb})}(\varphi)$ by adding the necessary parts to this model, that are, the spy state and the round-trip paths. Define $\langle W', R', V' \rangle$ as follows. Let $s \notin W$ some state, $W' = W \cup \{s\}$, $R' = R \cup \{(x, s), (s, x) \mid x \in W\}$, $V'(s) = \{s\}$ and $V'(p) = V(p)$ for $p \in \text{PROP} \setminus \{s\}$. By construction, $\langle W', R', V' \rangle, s \models \text{Struct}_{(\text{sb})}$, so Proposition 24 holds. We prove that for all ψ sub-formula of φ , $v \in W$, $S \subseteq W$ and $R'_S = R' \setminus \{(x, s), (s, x) \mid x \in S\}$, $\langle W, R, V, S \rangle, v \models \psi$ iff $\langle W', R'_S, V' \rangle, v \models (\psi)'$. This can be done by structural induction on ψ using Proposition 24. This proves that $\langle W, R, V, \emptyset \rangle, w \models \varphi$ iff $\langle W', R', V' \rangle, s \models \tau_{(\text{sb})}(\varphi)$, so $\tau_{(\text{sb})}(\varphi)$ is satisfiable. \square

Global Sabotage. In [29] it is shown that multimodal sabotage logic is undecidable via a reduction of the Post Correspondence Problem. The present proof extends this result to the monomodal case via a reduction of the satisfiability problem of the memory logic $\mathcal{ML}(\mathbb{F}, \mathbb{K})^2$. The notation $\square^i \varphi$ is defined as $\square^0 \varphi = \varphi$ and $\square^{n+1} \varphi = \square \square^n \varphi$.

One piece of data needed to build $\tau_{(\text{gsb})}(\varphi)$ is the modal depth of the input formula ($\text{md}(\varphi)$). Up to the depth indicated by this value, $\text{Struct}_{(\text{gsb})}(\varphi)$ adds to every state a transition to some state where s holds (In fact, this latter state can be shared among several states of the input model.) It is as if each state of the input model had a flag that could be turned on to identify the state. Thus, remembering some state is simulated with $\text{Tr}_{(\text{gsb})}(\mathbb{F})$ by deleting the edge between the state and its s -successor. For $\text{Tr}_{(\text{gsb})}(\mathbb{K})$, we check whether the current state has an s -successor. The idea is illustrated in the following image.

²The same translation and proof can be adapted to show that *locSML* [15] is undecidable. One only needs to change the $\langle \text{gsb} \rangle$ operator into the operator of *locSML* in Definition 26. This dynamic operator is similar to the global sabotage one, except that it can only delete edges that start at the evaluation state.



Definition 26. Define $\tau_{\langle \text{gsb} \rangle}(\varphi) = \text{Struct}_{\langle \text{gsb} \rangle}(\varphi) \wedge \text{Tr}_{\langle \text{gsb} \rangle}(\varphi)$, where:

$$\text{Struct}_{\langle \text{gsb} \rangle}(\varphi) = \neg s \wedge \bigwedge_{0 \leq i \leq \text{md}(\varphi)} \Box^i (\neg s \rightarrow (\Diamond s \wedge \langle \text{gsb} \rangle \neg \Diamond s))$$

$$\begin{aligned} \text{Tr}_{\langle \text{gsb} \rangle}(p) &= p \quad \text{for } p \in \text{PROP appearing in } \varphi \\ \text{Tr}_{\langle \text{gsb} \rangle}(\mathbb{K}) &= \neg \Diamond s \\ \text{Tr}_{\langle \text{gsb} \rangle}(\neg \psi) &= \neg \text{Tr}_{\langle \text{gsb} \rangle}(\psi) \\ \text{Tr}_{\langle \text{gsb} \rangle}(\psi \wedge \chi) &= \text{Tr}_{\langle \text{gsb} \rangle}(\psi) \wedge \text{Tr}_{\langle \text{gsb} \rangle}(\chi) \\ \text{Tr}_{\langle \text{gsb} \rangle}(\Diamond \psi) &= \Diamond (\neg s \wedge \text{Tr}_{\langle \text{gsb} \rangle}(\psi)) \\ \text{Tr}_{\langle \text{gsb} \rangle}(\mathbb{R} \psi) &= \langle \text{gsb} \rangle (\neg \Diamond s \wedge \text{Tr}_{\langle \text{gsb} \rangle}(\psi)) \end{aligned}$$

Proposition 27. Let $\text{dist}(a, b)$ the minimal number of R -steps to reach some state b from some state a . Let φ some memory logic formula. If $\langle W, R, V \rangle, w \models \text{Struct}_{\langle \text{gsb} \rangle}(\varphi)$, then for all $x \in W$ such that $\text{dist}(w, x) \leq \text{md}(\varphi)$, x has a successor where s holds.

Proposition 28. If $\langle W, R, V \rangle, w \models \Diamond s \wedge \langle \text{gsb} \rangle \neg \Diamond s$, then w has one and only one successor where s holds.

Lemma 29. Let φ be an $\mathcal{ML}(\mathbb{R}, \mathbb{K})$ -formula in PNF that does not contain the propositional symbol s . Then, φ is satisfiable iff $\tau_{\langle \text{gsb} \rangle}(\varphi)$ is satisfiable.

Proof. (\Leftarrow) Suppose $\langle W, R, V \rangle, w \models \tau_{\langle \text{gsb} \rangle}(\varphi)$. Let $W' = W \setminus V(s)$, $R' = R \cap (W' \times W')$, and $V'(p) = V(p) \cap W'$ for $p \in \text{PROP} \setminus \{s\}$. We should prove that for all ψ sub-formula of φ of modal depth $\text{md}(\psi) \leq \text{md}(\varphi) - \text{dist}(w, v)$, $v \in W'$ accessible from w within $\text{md}(\varphi)$ steps, $S \subseteq W'$, and $R_S = R \setminus \{(x, y) \mid x \in S, y \in V(s)\}$, then $\langle W', R', V', S \rangle, v \models \psi$ iff $\langle W, R_S, V \rangle, v \models \text{Tr}_{\langle \text{gsb} \rangle}(\psi)$.

The proof is by structural induction on ψ . The non-memory cases are easy. For the \mathbb{K} case, we should show that $\langle W', R', V', S \rangle, v \models \mathbb{K}$ iff $\langle W, R_S, V \rangle, v \models \neg \Diamond s$, this is immediate by Proposition 27 and the definitions of S and R_S .

Then for the remaining case, we have to show that $\langle W', R', V', S \rangle, v \models \neg \mathbb{K} \wedge \mathbb{R} \chi$ iff $\langle W, R_S, V \rangle, v \models \Diamond s \wedge \langle \text{gsb} \rangle (\neg \Diamond s \wedge \text{Tr}_{\langle \text{gsb} \rangle}(\chi))$.

This is done by the following series of equivalences:

$$\begin{aligned} & \langle W', R', V', S \rangle, v \models \neg \mathbb{K} \wedge \mathbb{R} \chi \\ \stackrel{\text{iff}}{\Leftrightarrow} & v \notin S \text{ and } \langle W', R', V', S \rangle, v \models \mathbb{R} \chi \\ \stackrel{\text{iff}}{\Leftrightarrow} & v \notin S \text{ and } \langle W', R', V', S \cup \{v\} \rangle, v \models \chi \\ \stackrel{\text{iff}}{\Leftrightarrow} & v \notin S \text{ and } \langle W, R_S \setminus \{(v, y) \mid y \in V(s)\}, V \rangle, v \models \text{Tr}_{\langle \text{gsb} \rangle}(\chi) \\ \stackrel{\text{iff}}{\Leftrightarrow} & v \notin S \text{ and } \langle W, R_S \setminus \{(v, y) \mid y \in V(s)\}, V \rangle, v \models \neg \Diamond s \wedge \text{Tr}_{\langle \text{gsb} \rangle}(\chi) \\ \stackrel{\text{Proposition 28}}{\Leftrightarrow} & \langle W, R_S, V \rangle, v \models \Diamond s \wedge \langle \text{gsb} \rangle (\neg \Diamond s \wedge \text{Tr}_{\langle \text{gsb} \rangle}(\chi)) \end{aligned}$$

(\Rightarrow) Suppose $\langle W, R, V, \emptyset \rangle, w \models \varphi$. Let $s \notin W$. Define $\langle W', R', V' \rangle$, where $W' = W \cup \{s\}$, $R' = R \cup \{(v, s) \mid v \in W\}$, $V'(s) = \{s\}$, and $V'(p) = V(p)$, for $p \in$

PROP appearing in φ . It is easy to check that $\langle W', R', V' \rangle, w \models \text{Struct}_{\langle \text{gsb} \rangle}(\varphi)$, hence Proposition 27 holds. Then, let us prove that for all ψ sub-formula of φ of modal depth $\text{md}(\psi) \leq \text{md}(\varphi) - \text{dist}(w, v)$, $v \in W$ accessible from w within $\text{md}(\varphi)$ steps, $S \subseteq W$ and $R'_S = R' \setminus \{(x, s) \mid x \in S\}$, we have the equivalence $\langle W, R, V, S \rangle, v \models \psi$ iff $\langle W', R'_S, V' \rangle, v \models \text{Tr}_{\langle \text{gsb} \rangle}(\psi)$. This is done by structural induction on ψ . For the case $\langle \mathbb{K} \rangle$ the equivalence is immediate, and for the case $\neg \langle \mathbb{K} \rangle \wedge \langle \mathbb{R} \rangle \chi$, Proposition 28 provides the equivalence needed. \square

4.2.2 Bridge Logic

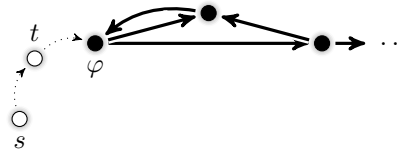
Local Bridge. For local bridge logic, we use a spy state that is initially disconnected from the input model. When some state should be memorized, the spy state gets connected (in both directions) to it. This construction is quite special since we do not have pre-built gadgets in the input model, as they get built on demand.

Let us first show the following result, that enables us to force the evaluation state to be the only one in the model to satisfy s :

Lemma 30. *Let $\varphi = s \wedge \Box \perp \wedge [\text{br}](s \rightarrow [\text{br}] \neg s)$. If $\mathcal{M}, w \models \varphi$, then w is the only state in the model \mathcal{M} where s holds.*

Proof. First, w obviously satisfies s and does not have any successor. Now, we have $\mathcal{M}, w \models [\text{br}](s \rightarrow [\text{br}] \neg s)$. In particular this means that $\mathcal{M}_{ww}^+, w \models s \rightarrow [\text{br}] \neg s$, hence $\mathcal{M}_{ww}^+, w \models [\text{br}] \neg s$. Since in \mathcal{M}_{ww}^+ , the state w is only connected to itself, this means that for all $v \neq w$, we have $\mathcal{M}_{ww, wv}^+, v \models \neg s$, this also means that $\mathcal{M}, v \not\models s$ for all $v \neq w$. \square

For Bridge Logics, $\text{Struct}_{\langle \text{br} \rangle}$ adds to the input model a spy state in which s holds. By Lemma 30, (1) in Definition 31 ensures that the evaluation state has no successor and that it is the only state in the model where s holds. And (2) ensures that there are no edges from $\neg s$ -states (anywhere in the model) to the spy state. The idea is illustrated in the following image, where t is a propositional symbol used in $\text{Tr}_{\langle \text{br} \rangle}(\varphi)$ and dotted lines represent edges created with the $\langle \text{br} \rangle$ operator.



Definition 31. Define $\tau_{\langle \text{br} \rangle}(\varphi) = \text{Struct}_{\langle \text{br} \rangle} \wedge \text{Tr}_{\langle \text{br} \rangle}(\varphi)$, where:

$$\begin{aligned} \text{Struct}_{\langle \text{br} \rangle} = & s \wedge \Box \perp \wedge [\text{br}](s \rightarrow [\text{br}] \neg s) & (1) \\ & \wedge [\text{br}](\neg s \rightarrow \Box \neg s) & (2) \end{aligned}$$

$\text{Tr}_{\langle \text{br} \rangle}(\varphi) = \langle \text{br} \rangle(\neg s \wedge t \wedge \langle \text{br} \rangle(\neg s \wedge \neg t \wedge (\varphi)'))$, with:

$$\begin{aligned} (p)' &= p \quad \text{for } p \in \text{PROP appearing in } \varphi \\ (\langle \mathbb{K} \rangle)' &= \langle \mathbb{K} \rangle s \\ (\neg \psi)' &= \neg(\psi)' \\ (\psi \wedge \chi)' &= (\psi)' \wedge (\chi)' \\ (\langle \mathbb{D} \rangle \psi)' &= \langle \mathbb{D} \rangle(\neg s \wedge \neg t \wedge (\psi)') \\ (\langle \mathbb{R} \rangle \psi)' &= \langle \text{br} \rangle(s \wedge \langle \text{br} \rangle(\neg s \wedge \langle \mathbb{D} \rangle s \wedge (\psi)')) \end{aligned}$$

$\text{Tr}_{\langle \text{br} \rangle}(\varphi)$ first creates two edges until a $\neg s$ -state, where the translation of φ holds. For $\text{Tr}_{\langle \text{br} \rangle}(\textcircled{\text{r}})$ we do a round-trip of bridging from the current state to the spy state. Note that the second part of this round-trip has to be from the spy state to the remembered state, since it is the only way to satisfy $\langle \text{br} \rangle(\diamond s)$. Also note that this would not work if the s state was directly connected to the input model; this is why we use the intermediate t -state. For $\text{Tr}_{\langle \text{br} \rangle}(\textcircled{\text{k}})$ we check whether there is an edge to a state where s holds.

Proposition 32. *Let $\langle W, R, V \rangle$ a model such that there is a unique state s where s holds, there is no state $x \in W$ such that $(x, s) \in R$, and there is a component $C \subseteq W$ such that $s \notin C$ and for all $y \in C$, $(s, y) \notin C$. Let $S \subseteq C$ and $R_S = R \cup \{(x, s), (s, x) \mid x \in S\}$.*

Then in the model $\langle W, R_S, V \rangle$, evaluating the formula $\langle \text{br} \rangle(s \wedge \langle \text{br} \rangle \diamond s)$ at some state $y \in C \setminus S$, changes the evaluation state to s , then again to the same state y , adding the edges (y, s) and (s, y) to the relation.

Lemma 33. *Let φ be an $\mathcal{ML}(\textcircled{\text{r}}, \textcircled{\text{k}})$ -formula in PNF that does not contain the propositional symbols s and t . Then, φ is satisfiable iff $\tau_{\langle \text{br} \rangle}(\varphi)$ is satisfiable.*

Proof. (\Leftarrow) Suppose $\langle W, R, V \rangle, s \models \tau_{\langle \text{br} \rangle}(\varphi)$. Define $\mathcal{M}' = \langle W', R', V', \emptyset \rangle$ with $W' = W \setminus V(s) \setminus V(t)$, $R' = R \cap (W' \times W')$, and $V'(p) = V(p) \cap W'$ for all $p \in \text{PROP}$. By definition of $\text{Tr}_{\langle \text{br} \rangle}$ there is $w' \in W'$ such that $s \neq w'$ and $\langle W, R, V \rangle, w' \models (\varphi)'$.

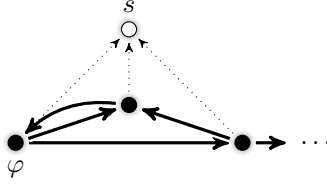
Let ψ a sub-formula of φ , $v \in W'$, $S \subseteq W'$, and $R_S = R \cup \{(x, s), (x, v) \mid x \in S\}$, then we will prove that $\langle W', R', V', S \rangle, v \models \psi$ if, and only if, $\langle W, R_S, V \rangle, v \models (\psi)'$.

We prove it by structural induction on ψ . For the $\neg(\textcircled{\text{k}}) \wedge (\textcircled{\text{r}})\chi$ case, suppose $\langle W', R', V', S \rangle, v \models \neg(\textcircled{\text{k}}) \wedge (\textcircled{\text{r}})\chi$. By definition, this is equivalent to $\langle W', R', V', S \cup \{v\} \rangle, v \models \chi$ with $v \notin S$. Then, by definition of R_S and inductive hypothesis we get $\langle W, (R_S)_{\{(v,s), (s,v)\}}^+, V \rangle, s \models (\chi)'$, with $(v, s) \notin R_S$ and $(s, v) \notin R_S$. By Proposition 32, this is equivalent to $\langle W, R_S, V \rangle, v \models \neg \diamond s \wedge \langle \text{br} \rangle(s \wedge \langle \text{br} \rangle(\diamond s \wedge (\chi)'))$. thus we have $\langle W, R_S, V \rangle, v \models (\neg(\textcircled{\text{k}}) \wedge (\textcircled{\text{r}})\chi)'$.

(\Rightarrow) Suppose $\langle W, R, V, \emptyset \rangle, w \models \varphi$. Let $s, t \notin W$. Define $\mathcal{M}' = \langle W', R, V' \rangle$ such that $W' = W \cup \{s, t\}$, $V'(s) = \{s\}$, $V'(t) = \{t\}$ and $V'(p) = V(p)$ for $p \in \text{PROP}$ appearing in φ . We can easily check that $\langle W', R, V' \rangle, s \models \text{Struct}_{\langle \text{br} \rangle}$, and we can also check by structural induction on φ that $\langle W, R, V, S \rangle, w \models \varphi$ iff $\langle W', R_S, V' \rangle, s \models \text{Tr}_{\langle \text{br} \rangle}(\varphi)$, where $R_S = R \cup \{(v, s), (s, v) \mid v \in S\}$. \square

Global Bridge. The global bridge operator is able to add edges in the model. This is why, to mark some state, we use this operator to add an edge to some s -state. Then, we enforce that the initial model does not have any reachable state where s holds.

In this case $\text{Struct}_{\langle \text{gbr} \rangle}(\varphi)$ ensures that no state of the input model has s -successors. Storing a state in the memory is simulated by creating an edge to an s -state, and checking whether the current state of evaluation is in the memory is simulated by checking the presence of an s -successor. Observe that we could have either one state where s holds or (possibly) different s -states for each state of the input model.



Definition 34. Define $\tau_{\langle \text{gbr} \rangle}(\varphi) = \text{Struct}_{\langle \text{gbr} \rangle}(\varphi) \wedge \text{Tr}_{\langle \text{gbr} \rangle}(\varphi)$, where:

$$\text{Struct}_{\langle \text{gbr} \rangle}(\varphi) = \bigwedge_{0 \leq i \leq \text{md}(\varphi)+1} \Box^i \neg s$$

$$\begin{aligned} \text{Tr}_{\langle \text{gbr} \rangle}(p) &= p \quad \text{for } p \in \text{PROP} \text{ appearing in } \varphi \\ \text{Tr}_{\langle \text{gbr} \rangle}(\mathbb{K}) &= \Diamond s \\ \text{Tr}_{\langle \text{gbr} \rangle}(\neg\psi) &= \neg \text{Tr}_{\langle \text{gbr} \rangle}(\psi) \\ \text{Tr}_{\langle \text{gbr} \rangle}(\psi \wedge \chi) &= \text{Tr}_{\langle \text{gbr} \rangle}(\psi) \wedge \text{Tr}_{\langle \text{gbr} \rangle}(\chi) \\ \text{Tr}_{\langle \text{gbr} \rangle}(\Diamond\psi) &= \Diamond(\neg s \wedge \text{Tr}_{\langle \text{gbr} \rangle}(\psi)) \\ \text{Tr}_{\langle \text{gbr} \rangle}(\mathbb{T}\psi) &= \langle \text{gbr} \rangle(\Diamond s \wedge \text{Tr}_{\langle \text{gbr} \rangle}(\psi)) \end{aligned}$$

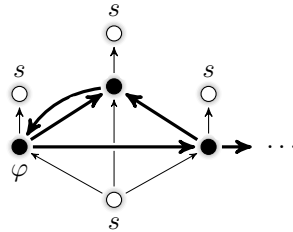
Lemma 35. Let φ be an $\mathcal{ML}(\mathbb{T}, \mathbb{K})$ -formula in PNF that does not contain the propositional symbol s . Then, φ is satisfiable iff $\tau_{\langle \text{gbr} \rangle}(\varphi)$ is satisfiable.

Proof. It is an easier version of the proof of Lemma 29, by observing that we are creating edges to a (possibly more than one) s -state instead of deleting them. \square

4.2.3 Swap Logic

Local Swap. We introduce a new version of the translation given in [5] that uses only one propositional symbol. The idea is that we have each state pointing to some states called *switch states*, and memorizing a state is represented by swapping such edges. Then, no edge pointing to a switch means that the state has been memorized. We use the notation $\Box^{(n)}\varphi$ for $\bigwedge_{1 \leq i \leq n} \Box^i \varphi$.

In this case $\text{Struct}_{\langle \text{sw} \rangle}$ adds “switch states”, which are in one-to-one correspondence with the states of the input model, together with a spy state. By (2) in Definition 36, each $\neg s$ -state at one, two and three steps from the evaluation state, has a unique dead-end successor where s holds (switch state). By (3) and (4), switch states (corresponding to states at distance 1, 2 and 3) can be reached from the evaluation state by a unique path. (5) makes the evaluation state a spy state. All these conjuncts together ensure that switch states are independent one from another. The idea is illustrated in the following image.



Definition 36. Define $\tau_{\langle \text{sw} \rangle} = \text{Struct}_{\langle \text{sw} \rangle} \wedge \text{Tr}_{\langle \text{sw} \rangle}(\varphi)$, where:

$$\begin{aligned} \text{Struct}_{\langle \text{sw} \rangle} = & \\ & s \wedge \Box \neg s \tag{1} \\ & \wedge \Box^{(3)}(\neg s \rightarrow \text{Uniq}) \tag{2} \\ & \wedge \Box[\text{sw}](s \rightarrow \Box\Box\Box(s \rightarrow \Box\perp)) \tag{3} \\ & \wedge \Box\Box[\text{sw}](s \rightarrow \Box\Box\Box(s \rightarrow \Box\perp)) \tag{4} \\ & \wedge [\text{sw}][\text{sw}](\neg s \rightarrow \langle \text{sw} \rangle(s \wedge \Diamond((\Box\neg s) \rightarrow \Diamond\Diamond(s \wedge \Diamond\neg\Diamond s)))) \tag{5} \end{aligned}$$

$$\text{Uniq} = \Diamond(s \wedge \Box\perp) \wedge [\text{sw}](s \rightarrow \Box\neg\Diamond s)$$

$\text{Tr}_{\langle \text{sw} \rangle}(\varphi) = \Diamond(\varphi)'$, with:

$$\begin{aligned} (p)' &= p \quad \text{for } p \in \text{PROP} \text{ appearing in } \varphi \\ (\mathbb{K})' &= \neg\Diamond s \\ (\neg\psi)' &= \neg(\psi)' \\ (\psi \wedge \chi)' &= (\psi)' \wedge (\chi)' \\ (\Diamond\psi)' &= \Diamond(\neg s \wedge (\psi)') \\ (\mathbb{T}\psi)' &= \langle \text{sw} \rangle(s \wedge \Diamond(\psi)') \end{aligned}$$

For $\text{Tr}_{\langle \text{sw} \rangle}(\mathbb{T}\varphi)$ we traverse and swap the edge between the current state and its switch state, and come back to the same state. For $\text{Tr}_{\langle \text{sw} \rangle}(\mathbb{K})$, we check whether the current state has not an edge to its switch state.

Proposition 37. Let $\langle W, R, V \rangle, s \models \text{Struct}_{\langle \text{sw} \rangle}$. Let W^w the connected component of W from state w . Let $W' = W^w \setminus V(s)$ and $S \subseteq W'$. Then $T = \{(v', v) \mid v \in S \wedge (v, v') \in R \wedge v' \in V(s)\}$ is a bijection.

Lemma 38. Let φ be an $\mathcal{ML}(\mathbb{T}, \mathbb{K})$ -formula in PNF that does not contain the propositional symbol s . Then, φ is satisfiable iff $\tau_{\langle \text{sw} \rangle}(\varphi)$ is satisfiable.

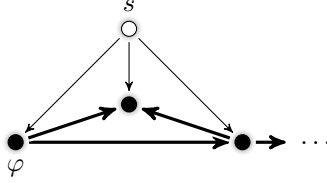
Proof. (\Leftarrow) From a pointed model $\langle W, R, V \rangle, w$ of $\tau_{\langle \text{sw} \rangle}(\varphi)$ we can extract a pointed model $\langle W', R', V', \emptyset \rangle, w'$ satisfying φ following the same definition as in the proof of Lemma 25.

For all ψ sub-formula of φ , $v \in W'$, $S \subseteq W'$, $T = \{(v', v) \mid v \in S \wedge (v, v') \in R \wedge v' \in V(s)\}$ and $R_S = (R \setminus T^{-1}) \cup T$, we will prove that $\langle W', R', V', S \rangle, v \models \psi$ if, and only if, $\langle W, R_S, V \rangle, v \models (\psi)'$.

We do it by structural induction on ψ . We prove the $\neg\mathbb{K} \wedge \mathbb{T}\chi$ case. Suppose $\langle W', R', V', S \rangle, v \models \neg\mathbb{K} \wedge \mathbb{T}\chi$. Then by definition, $v \notin S$ and $\langle W', R', V', S \cup \{v\} \rangle, v \models \chi$, and by Proposition 37, we have $(v, v') \in R_S$ for a unique $v' \in V(s)$. Then, by definition of R_S and inductive hypothesis we get $\langle W, (R_S)_{v', v}^*, V \rangle, v \models (\chi)'$. By definition of \models and by Proposition 37, $\langle W, (R_S)_{v', v}^*, V \rangle, v' \models s \wedge \Diamond(\chi)'$, and again, $\langle W, R_S, V \rangle, v \models \Diamond s \wedge \langle \text{sw} \rangle(s \wedge \Diamond(\chi)')$, thus we have, equivalently, $\langle W, R_S, V \rangle, v \models (\neg\mathbb{K} \wedge \mathbb{T}\chi)'$.

(\Rightarrow) Suppose $\langle W, R, V, \emptyset \rangle, w \models \varphi$. Let sw be a bijective function between W and a set U such that $U \cap W = \emptyset$, and $s \notin U \cup W$. Define $\mathcal{M}' = \langle W', R', V' \rangle$ such that $W' = W \cup \{s\} \cup U$, $R' = R \cup \{(s, w) \mid w \in W\} \cup \{(w, sw(w)) \mid w \in W\}$, $V'(s) = \{s\} \cup U$, and $V'(p) = V(p)$ for $p \in \text{PROP}$ appearing in φ . It is easy to check that $\langle W', R', V' \rangle, s \models \text{Struct}_{\langle \text{sw} \rangle}$, in particular, Proposition 37 is relevant. Then, we can easily prove that for all ψ sub-formula of φ , $v \in W$, $S \subseteq W$, $T = \{(sw(v), v) \mid v \in S\}$ and $R'_S = (R' \setminus T^{-1}) \cup T$, we have the equivalence $\langle W, R, V, S \rangle, v \models \psi$ iff $\langle W', R'_S, V' \rangle, v \models (\psi)'$. This is done by structural induction on ψ . \square

Global Swap. The global swap operator is able to change the direction of some edge in the model. In particular, we are interested in the ability to swap, for some state, an incoming edge (undetectable for the basic modal logic) into an outgoing edge. This is why this translation is similar to the one of global bridge logic. Initially, the model does not have any reachable state where s holds. As for global sabotage and global bridge, there may be many states where s holds in the model with edges to states of the input model. The idea is illustrated in the following image, where only one s state is shown.



Definition 39. Define $\tau_{\langle \text{gsw} \rangle}(\varphi) = \text{Struct}_{\langle \text{gsw} \rangle}(\varphi) \wedge \text{Tr}_{\langle \text{gsw} \rangle}(\varphi)$, where:

$$\text{Struct}_{\langle \text{gsw} \rangle}(\varphi) = \bigwedge_{0 \leq i \leq \text{md}(\varphi)+1} \Box^i \neg s$$

$$\begin{aligned} \text{Tr}_{\langle \text{gsw} \rangle}(p) &= p \quad \text{for } p \in \text{PROP} \text{ appearing in } \varphi \\ \text{Tr}_{\langle \text{gsw} \rangle}(\mathbb{K}) &= \Diamond s \\ \text{Tr}_{\langle \text{gsw} \rangle}(\neg\psi) &= \neg \text{Tr}_{\langle \text{gsw} \rangle}(\psi) \\ \text{Tr}_{\langle \text{gsw} \rangle}(\psi \wedge \chi) &= \text{Tr}_{\langle \text{gsw} \rangle}(\psi) \wedge \text{Tr}_{\langle \text{gsw} \rangle}(\chi) \\ \text{Tr}_{\langle \text{gsw} \rangle}(\Diamond\psi) &= \Diamond(\neg s \wedge \text{Tr}_{\langle \text{gsw} \rangle}(\psi)) \\ \text{Tr}_{\langle \text{gsw} \rangle}(\textcircled{\text{R}}\psi) &= \langle \text{gsw} \rangle(\Diamond s \wedge \text{Tr}_{\langle \text{gsw} \rangle}(\psi)) \end{aligned}$$

Proposition 40. Let $\langle W, R, V \rangle, w \models \neg \Diamond s \wedge \langle \text{gsw} \rangle \Diamond s$. Then, by the semantics of the global swap operator, there exists a state $v \in W \setminus \{w\}$ such that $(v, w) \in R$ and $v \in V(s)$.

Lemma 41. Let φ be an $\text{ML}(\textcircled{\text{R}}, \mathbb{K})$ -formula in PNF that does not contain the propositional symbol s . Then, φ is satisfiable iff $\tau_{\langle \text{gsw} \rangle}(\varphi)$ is satisfiable.

Proof. It is an adaptation of the proof of Lemma 35, using Proposition 40. \square

5 Implementation and Examples

In Section 3 we have shown translations from relation-changing logics into hybrid logics. We mentioned that this provided us a way to reuse existing hybrid logic theorem provers to check for satisfiability of relation-changing logics.

We implemented these translations into the tableaux-based theorem prover HTab [27]. Its version 1.7.1 can be downloaded from <http://hub.darcs.net/gh/htab> along with example formulas.

When HTab gets a relation-changing formula as input, it first translates it to the corresponding $\mathcal{HL}(\text{E}, \downarrow)$ formula (or more precisely, $\mathcal{HL}(\cdot, \downarrow)$ formula in the case of Local Sabotage and Local Swap), and then runs its internal hybrid tableaux calculus on the translation.

For all three translations, the implementation has the following particular case to avoid introducing unnecessary nominals:

$$(\Diamond\varphi)'_{\emptyset} = \Diamond(\varphi)'_{\emptyset}$$

Since the translations are equivalence-preserving, the models built by HTab satisfy the input relation-changing formula. Two illustrative examples are shown in Figure 2.

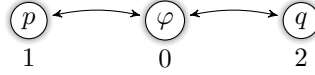
Input $\mathcal{ML}(\langle \text{sb} \rangle)$ -formula:

$$\begin{aligned} & \diamond(p \wedge \neg q \wedge \diamond\diamond p) \\ & \wedge \diamond(q \wedge \neg p \wedge \diamond\diamond q) \\ & \wedge [\text{sb}](p \rightarrow \square\square\neg p) \\ & \wedge [\text{sb}](q \rightarrow \square\square\neg q) \end{aligned}$$

Translated hybrid formula:

$$\begin{aligned} & \diamond(p \wedge \neg q \wedge \diamond\diamond p) \\ & \wedge \diamond(q \wedge \neg p \wedge \diamond\diamond q) \\ & \wedge \downarrow n_0.\square(\downarrow n_1.(\neg p \vee \downarrow n_2.\square((n_1 \wedge n_2:n_0) \vee \downarrow n_3.\square((n_1 \wedge n_3:n_0) \vee \neg p)))) \\ & \wedge \downarrow n_0.\square(\downarrow n_1.(\neg q \vee \downarrow n_2.\square((n_1 \wedge n_2:n_0) \vee \downarrow n_3.\square((n_1 \wedge n_3:n_0) \vee \neg q)))) \end{aligned}$$

Model found by HTab:



Input $\mathcal{ML}(\langle \text{gsb} \rangle)$ -formula:

$$\begin{aligned} & \diamond(p \wedge \neg q \wedge \diamond\diamond p) \\ & \wedge \diamond(q \wedge \neg p \wedge \diamond\diamond q) \\ & \wedge \square\square(r \wedge \square\neg r) \\ & \wedge \langle \text{gsb} \rangle \square\square\square\perp \end{aligned}$$

Translated hybrid formula:

$$\begin{aligned} & \diamond(p \wedge \neg q \wedge \diamond\diamond p) \\ & \wedge \diamond(q \wedge \neg p \wedge \diamond\diamond q) \\ & \wedge \square\square(r \wedge \square\neg r) \\ & \wedge \downarrow n_0.\mathbf{E}\downarrow n_1.\diamond(\downarrow n_2.n_0:(\downarrow n_3.\square((n_2 \wedge n_3:n_1) \vee \downarrow n_4.\square((n_2 \wedge n_4:n_1) \\ & \vee \downarrow n_5.\square((n_2 \wedge n_5:n_1)))))) \end{aligned}$$

Model found by HTab:

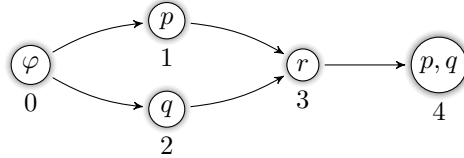


Figure 2: Examples of models built by HTab from relation-changing formulas

This implementation is useful to test the correctness of the translations in particular cases, just by checking the satisfiable/unsatisfiable output of the prover for known formulas. It is also useful for checking that models are built in the expected way, such as non-tree or diamond-shaped models.

Since the translation uses the \downarrow binder, HTab may never terminate on some specific relation-changing formulas. Even in terminating cases, the size of the translated formula (in particular for Swap Logic) leads to long running times. In many cases modifying the default heuristics used by HTab result in time improvements. A detailed empirical evaluation is left as future research.

6 Conclusions

In this article, we considered the satisfiability problem of six logics we named relation-changing logics. We considered three kinds of dynamic updates to the accessibility relation in a model: deleting, adding, and swapping edges, that can be performed either globally (anywhere in the model) or locally (modifying adjacent edges from the evaluation point).

We first introduced equivalence-preserving translations from each of these logics to a very expressive hybrid logic. Indeed, hybrid logic has operators to rename states in a model. We use the down-arrow operator \downarrow to name pairs of states that represent modified states. In this way, we keep track of the evolution of a model. It is known that the hybrid logic $\mathcal{HL}(\mathbf{E}, \downarrow)$ has the same expressive power as first-order logic, and we introduced standard translations from relation-changing logics to first-order logic in [6]. The advantage of translating to $\mathcal{HL}(\mathbf{E}, \downarrow)$ is that we easily obtain an implementation for relation-changing modal logics, by extending the hybrid logic theorem prover HTab [27]. Satisfiability checking and model building can thus be automated and were useful to empirically verify our translations on concrete cases. Of course, this could have also been achieved using automated provers for first-order logic, but at least in the case of classical logics, it has been noted in the literature that non-optimized translations of modal formulas into first-order logic lead to very poor performance (see, e.g., [28, 10] for details).

In the second half of this article, we showed that all six relation-changing logics are undecidable. We first showed that monomodal memory logic was undecidable, by reduction of a grid tiling problem. We then reduced satisfiability of memory logic to satisfiability of each one of the relation-changing logics we introduced.

We studied six relation-changing modal logics with the goal of covering a sufficiently varied sample of alternatives. Clearly, other operators could have been included in this exploration, and actually some alternative choices have been investigated in the literature, e.g., the adjacent sabotage operator discussed in [33]. The corresponding logic is denoted *locSML* in [15]. The undecidability proof of global sabotage in the present work can be easily adapted for *locSML*, since the dynamic operator of *locSML* does not change the evaluation state either. This shows that *locSML* is undecidable, answering an open question in [15]. In fact, we can also define *locBr* and *locSw*, i.e. adjacent bridge and swap logic respectively, and adapt the other global proofs to prove their undecidability. While we still need *ad hoc* proofs for each operator, unlike the results introduced in [6], these proofs are similar enough to easily get more evidence about the complex behaviour of this family of logics.

There are still many interesting questions to be answered. A natural direction to explore is the decidability status of the *finite satisfiability problem*, and more generally, finding interesting decidable fragments. Also, the *hybrid perspective* we present in this article provides a new way to think of the relation-changing framework. In particular, it would be interesting to use *hybridization techniques* (a standard technique in modal logic [16]) to find complete axiomatizations for these logics, or investigate the status of their interpolation theorem. On the other hand, it is possible to obtain decidable languages by defining operators which are specific for certain purposes. For instance, the relation-changing operators from [20], designed to model introspection steps in dynamic epistemic

logic, are decidable by reduction into Propositional Dynamic Logic, and can be seen as restricted versions of bridge and sabotage operators.

Acknowledgements. This work was partially supported by grant ANPCyT-PICTs-2016-0215, SeCyT-UNC, DFG grant LU 1417/2, and the Laboratoire International Associé “INFINIS”.

References

- [1] C. Areces. Hybrid Logics: The Old and the New. In *Proceedings of LogKCA-07*, pages 15–29, 2007.
- [2] C. Areces, P. Blackburn, and M. Marx. A road-map on complexity for hybrid logics. In J. Flum and M. Rodríguez-Artalejo, editors, *Computer Science Logic*, number 1683 in Lecture Notes in Computer Science, pages 307–321, Madrid, Spain, 1999. Springer.
- [3] C. Areces, R. Fervari, and G. Hoffmann. Moving Arrows and Four Model Checking Results. In *Logic, Language, Information and Computation*, volume 7456 of *Lecture Notes in Computer Science*, pages 142–153. Springer, 2012.
- [4] C. Areces, R. Fervari, and G. Hoffmann. Tableaux for Relation-Changing Modal Logics. In *Frontiers of Combining Systems*, volume 8152 of *Lecture Notes in Computer Science*, pages 263–278, 2013.
- [5] C. Areces, R. Fervari, and G. Hoffmann. Swap Logic. *Logic Journal of the IGPL*, 22(2):309–332, 2014.
- [6] C. Areces, R. Fervari, and G. Hoffmann. Relation-Changing Modal Operators. *Logic Journal of the IGPL*, 23(4):601–627, 2015.
- [7] C. Areces, R. Fervari, G. Hoffmann, and M. Martel. Relation-Changing Logics as Fragments of Hybrid Logics. In D. Cantone and G. Delzanno, editors, *Proceedings of the Seventh International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2016, Italy*, volume 226 of *EPTCS*, pages 16–29, 2016.
- [8] C. Areces, R. Fervari, G. Hoffmann, and M. Martel. Undecidability of Relation-Changing Modal Logics. In *Dynamic Logic. New Trends and Applications - First International Workshop, DALI 2017, Brasilia, Brazil, September 23-24, 2017, Proceedings*, pages 1–16, 2017.
- [9] C. Areces, D. Figueira, S. Figueira, and S. Mera. The Expressive Power of Memory Logics. *The Review of Symbolic Logic*, 4(2):290–318, 2011.
- [10] C. Areces and D Gorín. Unsorted functional translations. *Electronic Notes in Theoretical Computer Science*, 278(0):3–16, 2011. Proceedings of the 7th Workshop on Methods for Modalities (M4M 2011).

- [11] C. Areces and B. ten Cate. Hybrid Logics. In P. Blackburn, F. Wolter, and J. van Benthem, editors, *Handbook of Modal Logic*, pages 821–868. Elsevier, 2007.
- [12] C. Areces, H. van Ditmarsch, R. Fervari, and F. Schwarzentruber. Logics with Copy and Remove. In *Logic, Language, Information, and Computation*, volume 8652 of *Lecture Notes in Computer Science*, pages 51–65. Springer, 2014.
- [13] C. Areces, H. van Ditmarsch, R. Fervari, and F. Schwarzentruber. The modal logic of copy and remove. *Information and Computation*, 255:243–261, 2017.
- [14] G. Aucher, J. van Benthem, and D. Grossi. Sabotage modal logic: Some model and proof theoretic aspects. In Wiebe van der Hoek, Wesley H. Holliday, and Wen-Fang Wang, editors, *Logic, Rationality, and Interaction - 5th International Workshop, LORI 2015 Taipei, Taiwan, October 28-31, 2015, Proceedings*, volume 9394 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2015.
- [15] G. Aucher, J. van Benthem, and D. Grossi. Modal logics of sabotage revisited. *Journal of Logic and Computation*, 2016.
- [16] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.
- [17] P. Blackburn and J. Seligman. Hybrid Languages. *Journal of Logic, Language and Information*, 4(3):251–272, 1995.
- [18] P. Blackburn and J. van Benthem. Modal Logic: A Semantic Perspective. In *Handbook of Modal Logic*, pages 1–84. Elsevier, 2007.
- [19] R. Fervari. *Relation-Changing Modal Logics*. PhD thesis, Universidad Nacional de Córdoba, Argentina, 2014.
- [20] R. Fervari and F. R. Velázquez-Quesada. Dynamic epistemic logics of introspection. In A. Madeira and M. Benevides, editors, *Dynamic Logic. New Trends and Applications - First International Workshop, DALI 2017, Brasilia, Brazil, September 23-24, 2017, Proceedings*, volume 10669 of *Lecture Notes in Computer Science*, pages 82–97. Springer, 2017.
- [21] M. Franceschet, M. de Rijke, and B. Schlingloff. Hybrid logics on linear structures: Expressivity and complexity. In *TIME-ICTL 2003, Cairns, Queensland, Australia*, pages 166–173, 2003.
- [22] D. Gabbay. *Fibering Logics*. Oxford logic guides. Clarendon Press, 1999.
- [23] D. Gabbay. Introducing reactive Kripke semantics and arc accessibility. In A. Avron, N. Dershowitz, and A. Rabinovich, editors, *Pillars of Computer Science, Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday*, volume 4800 of *Lecture Notes in Computer Science*, pages 292–341. Springer, 2008.
- [24] D. Gabbay. *Reactive Kripke Semantics*. Cognitive Technologies. Springer, 2013.

- [25] N. Gierasimczuk, L. Kurzen, and F. R. Velázquez-Quesada. Learning and teaching as a game: A sabotage approach. In X. He, J. F. Horty, and E. Pacuit, editors, *LORI*, volume 5834 of *Lecture Notes in Computer Science*, pages 119–132. Springer, 2009.
- [26] V. Goranko and S. Passy. Using the universal modality: Gains and questions. *Journal of Logic and Computation*, 2(1):5–30, 1992.
- [27] G. Hoffmann and C. Areces. Htab: A terminating tableaux system for hybrid logic. *Electronic Notes in Theoretical Computer Science*, 231:3–19, March 2009.
- [28] U. Hustadt, R. A. Schmidt, and C. Weidenbach. Optimised functional translation and resolution. In H. de Swart, editor, *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX'98, Oisterwijk, The Netherlands, Proceedings*, volume 1397 of *Lecture Notes in Computer Science*, pages 36–37. Springer, May 5–8 1998. Contribution to the comparison section of modal theorem provers.
- [29] Ch. Löding and P. Rohde. Model checking and satisfiability for sabotage modal logic. In P. Pandya and J. Radhakrishnan, editors, *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science: 23rd Conference, Mumbai, India, December 15-17, 2003. Proceedings*, pages 302–313, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
- [30] M. Martel. On the Undecidability of Relation-Changing Logics. Master’s thesis, Universidad Nacional de Río Cuarto, Argentina, 2015.
- [31] M. Marx. Narcissists, stepmothers and spies. In *Proceedings of DL02, volume 53. CEUR*, 2002.
- [32] S. Mera. *Modal Memory Logics*. PhD thesis, Université Henri Poincaré, Nancy, France and Universidad de Buenos Aires, Argentina, 2009.
- [33] P. Rohde. *On games and logics over dynamically changing structures*. PhD thesis, RWTH Aachen, 2006.
- [34] T. Schneider. *The Complexity of Hybrid Logics over Restricted Frame Classes*. PhD thesis, University of Jena, 2007.
- [35] E. Spaan. *Complexity of modal logics*. PhD thesis, ILLC, University of Amsterdam, 1993.
- [36] B. ten Cate. *Model theory for extended modal languages*. PhD thesis, University of Amsterdam, 2005. ILLC Dissertation Series DS-2005-01.
- [37] B. ten Cate and M. Franceschet. On the complexity of hybrid logics with binders. In L. Ong, editor, *Computer Science Logic: 19th International Workshop, CSL 2005, 14th Annual Conference of the EACSL, Oxford, UK, August 22-25, 2005. Proceedings*, pages 339–354, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
- [38] J. van Benthem. An Essay on Sabotage and Obstruction. In *Mechanizing Mathematical Reasoning*, pages 268–276, 2005.

- [39] H. van Ditmarsch, W. van der Hoek, and B. Kooi. *Dynamic Epistemic Logic*. Synthese Library. Springer, 2007.
- [40] P. Van Emde Boas. The convenience of tilings. In *Complexity, Logic, and Recursion Theory*, pages 331–363. Marcel Dekker Inc, 1997.