South American Journal of Logic
Vol. 4, n. 2, pp. 385–406, 2018
ISSN: 2446-6719

On the Uncomputability of Partial Meet Contraction for Linear-Time Temporal Logic

Paulo T. Guerra and Renata Wassermann

Abstract

The AGM theory of belief revision has been successfully applied to families of logics satisfying certain assumptions, as being compact, for example. In this work we discuss some issues related to the application of the AGM paradigm to temporal logics. We focus on the linear-time temporal logic (LTL) and we show that the simple definition of an AGM partial meet contraction for sets of LTL formulas is an undecidable problem. We discuss possible ways to address this problem and show that under certain restrictions, it is possible to correctly define an operator of partial meet contraction.

Keywords: Belief revision, Computability, Linear-time Temporal Logic.

Introduction

The AGM paradigm [2] is a theory about how idealized rational agents should change their beliefs when receiving new information. We focus here in the operation of *contraction*, which consists in removing a formula from the agent's beliefs by rationally choosing a subset of the original beliefs that does not imply the specified sentence. This is important, for example, when an agent realizes that some of its beliefs are not true.

Although originally developed for propositional logic, AGM paradigm has been successfully adapted to other formalisms, as Description Logic [8, 20] among others [23]. However, most the known generalizations of AGM rely on the underlying logic being compact.

Recent works as [10] and [19] investigate the application of the AGM paradigm on non-compact logics, such as temporal logics. In [10] the authors propose new set postulates to characterize change operations over temporal beliefs. In [19], the authors explore a new construction characterized by the

classical AGM postulates that while the classical approach employs remainders of the belief being removed, it uses its complements.

In this work we investigate why works as [10, 19] needed to avoid the classical approach of *partial meet contraction* when dealing with non-compact logics. The primary goal of this work is to investigate whether *partial meet construction* [2] can be applied to temporal logics. We look specifically into the problem of how to define contraction operators based on the partial meet for LTL. We show that, given a set of LTL formulas B and a LTL formula α, verifying whether there exists a maximal subset of B that does not imply α (a remainder set) is in general undecidable. This is a crucial step for partial meet constructions, hence we explore restrictions on the input for which the remainder sets can be computed. Although we focus on LTL for the sake of clarity, the undecidability result also holds for other temporal or dynamic logics, such as CTL [5] and PDL [7, 13].

The rest of this paper is organized as follows: in the next section, we provide the needed background in LTL, partial meet contraction and the type of automata we will use in the proof of the undecidability. The third section is devoted to the undecidability result and the fourth to exploring restrictions on the language for avoinding the undecidability. In the last section, we conclude and point towards future work.

1 Preliminaries

1.1 Linear-time Temporal Logic

Temporal logics are the basis of the area of formal system verification. In formal verification, a system is described in a formal language and then checked against a set of desired properties, usually described in some temporal logic formalism.

Linear-time Temporal Logic [18, 16], or LTL for short, is a temporal logic where the future is represented by a linear sequence of states, extending infinitely into the future.

In LTL, there are temporal connectives to express statements about events in time. The main temporal connectives are X ("next state"), F ("some future state"), G ("globally in the future states"), U ("true until") and R ("released by"). The formal syntax of a LTL formula is given by the following BNF

$$\phi ::= \bot \mid p \mid \neg\phi \mid \phi \vee \phi \mid \mathrm{X}\phi \mid \phi \mathrm{U} \phi \mid \phi \mathrm{R} \phi$$

where $\top \equiv \neg\bot$, $\phi_1 \wedge \phi_2 \equiv \neg(\neg\phi_1 \vee \neg\phi_2)$, $\phi_1 \rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2$, $\mathrm{F}\phi \equiv \top \mathrm{U} \phi$ and $\mathrm{G}\phi \equiv \bot \mathrm{R} \phi$.

The semantic of LTL is given through a labeled transition system that finitely describes the sequence of possible states.

Definition 1.1 *A LTL model is a labeled transition system* $\mathcal{M} = \langle AP, S, s_0, R, L\rangle$ *where*

1. AP *is a countable set of propositional atoms,*
2. S *is a finite set of states,*
3. s_0 *is the initial state*
4. $R \subseteq S \times S$ *is a transition relation such that* R *is serial (left-total) and univocal (functional)*[1] *on* S*,*
5. $L : AP \to \mathcal{P}(S)$ *is a labeling function that indicates in which states a proposition holds.*

Figure 1 illustrates a state diagram for a LTL model defined as $M = \langle\{p, q\}$, $\{s_0, s_1\}$, s_0, $\{(s_0, s_1), (s_1, s_1)\}, L\rangle$ where $L(p) = \{s_0, s_1\}$ and $L(q) = \{s_1\}$.

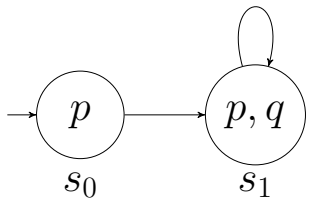

Figure 1: Example of a LTL model.

Although the functional property in Definition 1.1 turns this LTL model slight more restrictive than the usual Kripke structures, the Snipping Lemma [9] shows that such model exists for any satisfiable LTL formula. This definition of LTL model yields that each LTL model has an unique computation path, being a useful property for the proofs of this work.

Definition 1.2 *Let* $\mathcal{M} = \langle AP, S, s_0, R, L\rangle$ *be a LTL model. A computation path* $\pi = s_0 \to s_1 \to ...$ *in* $\mathcal{M}$ *is a sequence of states of* $\mathcal{M}$ *such that, for all* $i \geq 0$, $(s_i, s_{i+1}) \in R$.

The satisfaction relation $\models$ for LTL models is defined as follows.

[1] R is serial iff for all $a \in S$, $(a, b) \in R$. R is univocal iff for all $\{(a, b), (a, c)\} \subseteq R$, $b = c$.

Definition 1.3 *Let $\mathcal{M} = \langle AP, S, s_0, R, L\rangle$ be a LTL model and $s \in S$ a state of $\mathcal{M}$. The satisfaction relation $\mathcal{M}, s \models \phi$ is defined by structural induction on ϕ as follows*

1. $\mathcal{M}, s \not\models \bot$
2. $\mathcal{M}, s \models p$ *iff* $s \in L(p)$
3. $\mathcal{M}, s \models \neg\phi$ *iff* $\mathcal{M}, s \not\models \phi$
4. $\mathcal{M}, s \models \phi_1 \vee \phi_2$ *iff* $\mathcal{M}, s \models \phi_1$ *or* $\mathcal{M}, s \models \phi_2$
5. $\mathcal{M}, s \models \mathrm{X}\phi$ *iff* $(s, s') \in R$ *and* $\mathcal{M}, s' \models \phi$
6. $\mathcal{M}, s \models \phi_1 \mathrm{U} \phi_2$ *iff in the path fragment* $\pi = s_0 \to s_1 \to \ldots$ *, where* $s = s_0$*, holds that for some* s_n*,* $M, s_n \models \phi_2$ *and for all* $i < n$*,* $M, s_i \models \phi_1$*.*
7. $\mathcal{M}, s \models \phi_1 \mathrm{R} \phi_2$ *iff in the path fragment* $\pi = s_0 \to s_1 \to \ldots$ *, where* $s = s_0$*, holds that either (a)* $M, s_n \models \phi_1$ *for some* s_n *and* $M, s_i \models \phi_2$ *for all* $i \leq n$*, or (b)* $M, s_n \models \phi_2$ *for all* $i \geq 0$*.*

If $M, s_0 \models \phi$, we say that $\mathcal{M}$ satisfies ϕ, denoted by $\mathcal{M} \models \phi$. The model $\mathcal{M}$ in Figure 1, for example, satisfies the formulas $p \wedge \mathrm{X}q$, $p\mathrm{U}q$ and $\mathrm{FG}q$, but not $q \wedge \mathrm{X}p$ or $\mathrm{G}q$.

1.2 Remainder sets and partial meet contraction

Partial meet is a construction for contraction operators proposed by Alchourrón, Gärdenfors and Makinson [2]. This construction is based on the concept of a remainder set: given a belief set and an input belief, a remainder set is the set of all maximal subsets of the belief set that do not imply the given input.

Definition 1.4 *[2] Let K be a set of formulas, α a formula and Cn a consequence operator, the remainder set $K\bot\alpha$ of K and α is defined as follows. For any set X, $X \in K\bot\alpha$ if and only if*

1. $X \subseteq K$,
2. $\alpha \notin Cn(X)$,
3. *for all Y such that $X \subset Y \subseteq K$, $\alpha \in Cn(Y)$.*

The partial meet construction also makes use of the concept of selection function, a way of picking out the best elements of a remainder set according to an intended selection criterion.

Definition 1.5 *[2] Let K be a set of formulas and α a formula, a selection function for K e α is a function γ such that*

1. *If $K\perp\alpha \neq \emptyset$, then $\emptyset \neq \gamma(K\perp\alpha) \subseteq K\perp\alpha$,*
2. *Otherwise, $\gamma(K\perp\alpha) = \{K\}$.*

A partial meet contraction is then obtained by taking the intersection of the best elements picked out by a selection function.

Definition 1.6 *[2] Let K be a set of formulas, for any sentence α, the operation of partial meet contraction over K and α determined by a selection function γ is given by*

$$K -_{\gamma} \alpha = \bigcap \gamma(K\perp\alpha)$$

1.3 Linear Bounded Automaton

A *linear bounded automaton* (LBA) is a restricted type of Turing machines (TM) wherein the tape head in not allowed to move off the portion of the tape containing the input [21]. Proposed by [17], LBAs approximate Turing machines from actual computers, modeling the computation process with limited memory resources.[2]

Definition 1.7 *A linear bounded automaton (LBA) is a 7-tuple $\langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r\rangle$, where*

1. *Q is a finite set of states,*
2. *Σ is a finite input alphabet,*
3. *Γ is a finite tape alphabet, where $\Sigma \subseteq \Gamma$,*
4. *$\delta : Q' \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is the transition function, where $Q' = Q \setminus \{q_a, q_r\}$*
5. *$q_0 \in Q$ is the start state,*
6. *$q_a \in Q$ is the accept state, and*
7. *$q_r \in Q$ is the reject state, where $q_a \neq q_r$.*

[2]Definitions 1.7-1.10 are based on those given in [21] for Turing machines, with minor adaptations to address LBA.

and that δ is defined such that the LBA could not move its head off the positions occupied in the tape by the input.[3]

Figure 2 illustrates a state diagram of a LBA M formally defined as $M = \langle\{q_0, q_a, q_r\}, \{a, b\}, \{a, b\}, \delta, q_0,\ q_a, q_r\rangle$ where $\delta(q_0, a) = (q_a, a, R)$, $\delta(q_0, b) = (q_r, b, R)$ and undefined otherwise.

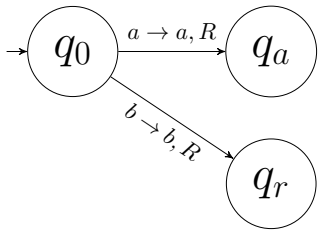

Figure 2: Diagram of a LBA

During a computation of a LBA, several changes may occur in the tape content, head location or the control state. These items define a *configuration* of a LBA and are usually represented by a single string, as given in Definition 1.8.

Definition 1.8 *Let $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r\rangle$ be a LBA and $C = c_1c_2...c_m$ a string over the alphabet $Q \cup \Gamma$, we say that C is a valid configuration of M if and only if there is a c_i such that $c_i \in Q$ and for all $i \neq j$, $c_j \in \Gamma$.*

For example, q_0aa and aq_aabb are valid configurations of M in Figure 2, but not q_0q_rb or $abab$.

A computation step of a LBA is a transition between two configurations such that one configuration yields the other according to its transition function.

Definition 1.9 *Let $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r\rangle$ be a LBA and C_i and C_j two valid configurations of M, we say that C_i yields C_j if, for $a, b, c \in \Gamma$, $u, v \in \Gamma^*$ and $q_i, q_j \in Q$, one of the following conditions holds:*

1. *$C_i = uq_iav$, $C_j = ubq_jv$ and $\delta(q_i, a) = (q_j, b, R)$.*
2. *$C_i = uaq_ibv$, $C_j = uq_jacv$ and $\delta(q_i, b) = (q_j, c, L)$.*

The start configuration of a LBA M on input w is q_0w. The *accepting configuration* is a configuration in which the state symbol is q_a. The *rejecting*

[3]This can be done by reserving two symbols in Σ to represent the left and right endmarkers of input such that the transition function may not overwrite the endmarkers with a different symbol, move to the left of the left endmarker, or move to the right of the right endmarker.

configuration is a configuration in which the state symbol is q_r. Accepting and rejecting configurations are *halting configurations* and do not yield further configurations.

Definition 1.10 *Let M be a LBA and w an input for M, a computation of M on w is a sequence of configuration $C_1, C_2, ..., C_l$ where*

1. *C_1 is a starting configuration,*
2. *Each C_i yields C_{i+1}.*

If C_l is an accepting configuration, we say that M *accepts* w. If C_l is a rejecting configuration, we say that M *rejects* w. In both cases, we say that M *halts* on w.

An interesting property of LBA is that, in contrast to the general case for Turing machines, the *acceptance problem* for LBA is decidable. Due to the limited number of distinct configurations of a LBA for an input, to decide whether a LBA accepts an input can be done in a finite amount of time. Since, for a LBA M with q states and g symbols in the tape alphabet, there are exactly qng^n distinct configurations for an input w of size n, if M accepts w it must take less than qng^n steps to reach an accepting configuration.

However many other undecidable problems for Turing machines remain undecidable for LBA. This is the case for the *emptiness problem* of LBA, that consists in determining whether a given LBA accepts any input.

We define the *language* of a LBA M as the set $L(M) = \{w \in \Sigma^* \mid M \text{ accepts } w\}$. Let E_{LBA} be the set of all LBA with an empty language,

$$E_{LBA} = \{\langle M \rangle \mid M \text{ is a LBA where } L(M) = \emptyset\}$$

The test of membership for E_{LBA} is undecidable problem[4].

In the next section, we establish the undecidability of our main problem, computing remainder sets for LTL, based on a reduction to the emptiness problem of LBA. We show that if our problem is decidable, we could decide whether any LBA M belongs to E_{LBA}, a contradiction.

2 Undecidability of partial meet contractions for LTL

In this section, we show that it is impossible to define an operator of partial meet contraction for LTL belief bases due to the undecidability of the *emptiness problem for $K \perp \alpha$*.

[4] A didactic proof for this can be seen in [21].

Definition 2.1 *Let K be a set of LTL formulas and $\alpha \in \mathcal{L}_{LTL}$, the emptiness problem for $K \perp \alpha$, $E_{K\perp\alpha}$, consists in determining whether there is any remainder set for K and α. We formally define $E_{K\perp\alpha}$ as*

$$E_{K\perp\alpha} = \{\langle K, \alpha\rangle \mid K\perp\alpha = \emptyset\},$$

where $K\perp\alpha = \emptyset$ if and only if $\langle K, \alpha\rangle \in E_{K\perp\alpha}$.

To prove that $E_{K\perp\alpha}$ is undecidable, we give a reduction from E_{LBA} using the concept of LBA-*equivalent* formulas.

2.1 Construction of LBA-*equivalent* formulas

A LBA-*equivalent* formula is a LTL formula ϕ_n^{M} that *encodes* a linear bounded automaton M in the sense that each model for ϕ_n^{M} describes a computation of M on inputs of n symbols.

For clarity, we first define a function $\sigma : \Sigma^* \to \mathrm{LTL}$ to convert arbitrary strings into LTL formulas,

$$\sigma(c_0c_1...c_m) = \bigwedge_{i=0}^{m} \mathrm{X}^i c_i, \text{ where } c_i \in \Sigma^*.$$

For example, $\sigma(abca) = a \wedge \mathrm{X}b \wedge \mathrm{XX}c \wedge \mathrm{XXX}a$, where $\Sigma = \{a, b, c\}$.

Let $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r\rangle$ be a LBA, # be a symbol that neither belongs to Q or Γ, and $AP = Q \cup \Gamma \cup \{\#\}$ be a set of propositional atoms, we build ϕ_n^{M} over AP based on all possible configurations $C_1, C_2, ..., C_l$ of M and all inputs with exactly n symbols. The main goal is to capture basic aspects of the computation of M, such as initial and halting configurations and its intermediate transitions.

Definition 2.2 *Let M be a LBA and $C_1, C_2, ..., C_l$ stand for all possible configurations of M with size $n + 1$. We define Π as the smallest set of LTL formulas where*

1. $\bigvee_{\beta\in\Delta} \beta \in \Pi$, *where* $\Delta = \{\sigma(\#q_0w\#) \mid w \in \Sigma^*$ *and w has exactly n symbols*$\}$ (*I*)
2. *For each C_i in $C_1, C_2, ..., C_l$,*

- *If $C_i = uqv$, where $u, v \in \Gamma^*$, $q \in \{q_a, q_r\}$, then*

$$G(\sigma(\#uqv\#) \to \sigma(\#uqv\#uqv\#)) \in \Pi \quad \text{(H)}$$

- *If* $C_i = uqav$ *and* $\delta(q,a) = (r,b,R)$, *where* $a,b \in \Gamma$, $u,v \in \Gamma^*$, $q,r \in Q$, *then*

$$G(\sigma(\#uqav\#) \rightarrow \sigma(\#uqav\#ubrv\#)) \in \Pi \quad \text{(R)}$$

- *If* $C_i = ubqav$ *and* $\delta(q,a) = (r,c,L)$, *where* $a,b,c \in \Gamma$, $u,v \in \Gamma^*$, $q,r \in Q$, *then*

$$G(\sigma(\#ubqav\#) \rightarrow \sigma(\#ubqav\#urbcv\#)) \in \Pi \quad \text{(L)}$$

3. $G((\bigvee_{p \in AP} p) \wedge (\bigwedge_{p \in AP} (p \rightarrow (\bigwedge_{q \in AP \setminus \{p\}} \neg q\ )))) \in \Pi \quad (X)$

The formula ϕ_n^M *is then defined as the conjunction of all formulas in* Π

$$\phi_n^M = \bigwedge_{\psi \in \Pi} \psi.$$

We call ϕ_n^M *the LBA-equivalent formula to* M *on inputs of size* n.

The clause (I) ensures that each model $\mathcal{M}$ that satisfies ϕ_n^{M} encodes a valid initial configuration of M in its initial sequence of states. The clause (H) ensures that if C_i is a halting configuration, then $\sigma(\#C_i\#)$ must occur infinitely often in $\mathcal{M}$. The clauses (R) and (L) ensure that, if a configuration C_i yields C_j, all path fragments in $\mathcal{M}$ that satisfies $\sigma(\#C_i\#)$ are immediately followed by states that satisfies $\sigma(\#C_j\#)$. Finally, the clause (X) ensures that only one proposition holds in each state[5].

Suppose, for example, that ϕ_2^{M} is the LBA-equivalent formula to the automaton M in Figure 2 on inputs of 2 symbols, according to Definition 2.2. By clause (I) in ϕ_2^{M}, all models that satisfy ϕ_2^{M} must satisfy

$$\sigma(\#q_0aa\#) \vee \sigma(\#q_0ab\#) \vee \sigma(\#q_0ba\#) \vee \sigma(\#q_0bb\#)$$

If a model $\mathcal{M}$ satisfies ϕ_2^{M}, its first states must encode a valid initial configuration of M. Figure 3 illustrates a possible initial sequence of states for $\mathcal{M}$.

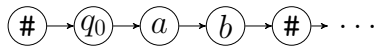

Figure 3: Possible initial sequence of states of $\mathcal{M}$

Since $\mathcal{M}$ also satisfies (R) in ϕ_2^{M}, $\sigma(\#q_0ab\#)$ holds in the initial state of $\mathcal{M}$ and $\delta(q_0,a) = (q_a,a,R)$ is a transition in M, the model $\mathcal{M}$ must also satisfy

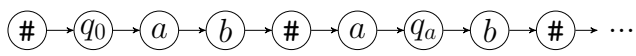

Figure 4: Following states in $\mathcal{M}$

$\sigma(\#q_0ab\#aq_ab\#)$ in its initial state. Figure 4 illustrates what should be the next sequence of states in $\mathcal{M}$.

The configuration aq_ab is an accepting configuration of M. Since $\mathcal{M}$ satisfies the clauses (H) in ϕ_2^{M}, this encoding must repeat infinitely often in $\mathcal{M}$. This could be achieved by defining a loop on this set of states.

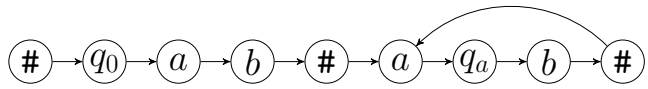

Figure 5: Diagram of a model $\mathcal{M}$ for ϕ_2^{M}.

The model shown in Figure 5 is a model of ϕ_2^{M} that encodes an accepting computation history for M with input ab.

2.2 Undecidability of $E_{K\perp\alpha}$

We show in Lemma 2.3 that the acceptance of an input by M is directly related to the satisfaction of a formula $\phi_n^{\mathrm{M}} \wedge \mathrm{F}q_a$.

Lemma 2.3 *Let $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r\rangle$ be a LBA and w an input for M of n symbols,*

M accepts w if and only if $\phi_n^M \wedge \mathrm{F}q_a$ is satisfiable.

Proof.

From left to right. If M accepts an input w, then there is a sequence of configurations $C_1, C_2, ..., C_l$ that M goes through as it accepts the input w. Let $\mathcal{C} = \#C_1\#...\#C_l\#$ be a concatenation of all $C_1, C_2, ..., C_l$ each one separated by the symbol $\# \notin Q \cup \Gamma$. We build a model $\mathcal{M} = \langle AP, S, s_0, R, L\rangle$ from $\mathcal{C}$ such that

- $AP = \Gamma \cup Q \cup \{\#\}$,
- $S = \{s_0, s_1, ..., s_{|\mathcal{C}|}\}$,
- $R = \{(s_i, s_{i+1}) \mid 0 \leq i < |\mathcal{C}|\} \cup \{(s_{|\mathcal{C}|}, s_{|\mathcal{C}|-(n+1)})\}$

[5]This clause is important in the later sections to build the an unique equivalence between LTL model and computation history.

– for all $s_i \in S$, $L(s_i) = \{c_i\}$, where c_i is the i-th symbol of $\mathcal{C}$.

Since M accepts w, we have that q_a is a symbol of $\mathcal{C}$. By construction, $s_i \in L(q_a)$ for some $s_i \in S$ and it can be reached from the initial state, thus it holds that $\mathcal{M} \models \mathrm{F}q_a$. Let n be the number of symbols of w, as C_1 is a valid initial configuration, the model $\mathcal{M}$ satisfies the clause (I) of ϕ_n^{M}. By construction, each state of $\mathcal{M}$ is related to one single proposition, thus $\mathcal{M}$ satisfies the clause (X) of ϕ_n^{M}. Since each C_{i+1} legally follows from C_i, for each state of $\mathcal{M}$ either $\sigma(\#C_i\#)$ does not hold on it or it is also true that $\sigma(\#C_i\#C_{i+1}\#)$. In both cases, the implications in clauses (R) or (L) are globally satisfied in $\mathcal{M}$. Finally, since C_l is an accepting configuration, the transition $(s_{|\mathcal{C}|}, s_{|\mathcal{C}|-(n+1)}) \in R$ ensures the satisfiability of the clause (H). Thus, $\mathcal{M} \models \phi_n^{\mathrm{M}}$ and hence $\phi_n^{\mathrm{M}} \wedge \mathrm{F}q_a$ is satisfiable.

From right to left. If $\phi_n^{\mathrm{M}} \wedge \mathrm{F}q_a$ is satisfiable, then there is a model $\mathcal{M} = \langle AP, S, s_0, R, L\rangle$ such that $\mathcal{M} \models \phi_n^{\mathrm{M}} \wedge \mathrm{F}q_a$. As $\mathcal{M} \models \mathrm{F}q_a$, there is a computation path $\pi = s_0 \to s_1 \to \cdots \to s_j \to \cdots \to s_k \to \ldots$ such that $s_i \in L(q_a)$ for some $i \geq 0$. Let s_j be the first state in this path such that $s_j \in L(q_a)$ and s_k the first following state such that $s_k \in L(\#)$. Let $\mathcal{C} = c_1c_2...c_k$ be a string such that $s_i \in L(c_i)$, for $1 \leq i \leq k$. Breaking $\mathcal{C}$ according to the delimiters #, we obtain a sequence of configurations $C_1, C_2, ..., C_l$.It remains to show that this is an accepting computation history for M for some input of n symbols. By construction of ϕ_n^{M}, each C_{i+1} must legally follow from C_i, since the clauses (R) and (L) were defined from the function δ of M. According to our choice, C_l has q_a as state symbol and by clause (H) it should be a valid accepting configuration. Finally, by clause (I) of ϕ_n^{M}, C_1 has q_0 as state symbol and it is followed by a string $w = w_1...w_n$ composed only by symbols of the input alphabet. Thus, $\mathcal{C}$ is an accepting computation history of M accepting w.

■

We show in Theorem 2.4 that $E_{K\perp\alpha}$ is undecidable. The undecidability result comes from the fact that if $E_{K\perp\alpha}$ is decidable, then the emptiness problem for LBA would also be decidable, a contradiction.

Theorem 2.4 *$E_{K\perp\alpha}$ is undecidable.*

Proof. This proof is by reduction from E_{LBA}. Suppose that the TM R decides $E_{K\perp\alpha}$. Construct a Turing machine S that decides E_{LBA} as follows.

S = "On input $\langle M \rangle$, where M is a LBA:

1. Run R on input $\langle K, \alpha \rangle$, where $K = \{\phi_i^{\mathrm{M}} \mid i \geq 0\}$ and $\alpha = \neg \mathrm{F} q_a$.
2. If R accepts, *accept*. If R rejects, *reject*."

Every element of $K \bot \neg \mathrm{F} q_a$ must be a singleton, since for all $\phi_i^{\mathrm{M}} \neq \phi_j^{\mathrm{M}}$, there is no LTL model that simultaneously satisfies clauses (I) and (X) in both ϕ_i^{M} and ϕ_j^{M}.

If R rejects $\langle K, \alpha \rangle$, then there is a $\{\phi_i^{\mathrm{M}}\} \in K \bot \neg \mathrm{F} q_a$ and a model $\mathcal{M}$ that satisfies ϕ_i^{M} and $\mathrm{F} q_a$. By Lemma 2.3, it holds that the LBA M accepts an input of size i, and thus the language of M is not empty.

If R accepts $\langle K, \alpha \rangle$, then $K \bot \neg \mathrm{F} q_a = \emptyset$. So, for all possible $K' = \{\phi_i^{\mathrm{M}}\}$, each LTL models $\mathcal{M}$ either $\mathcal{M} \not\models \phi_i^{\mathrm{M}}$ or $\mathcal{M} \models \neg \mathrm{F} q_a$. Since there is no ϕ_i^{M} such that $\phi_i^{\mathrm{M}} \wedge \mathrm{F} q_a$ is satisfiable, by Lemma 2.3 there is no input accepted by M, thus the language of M is empty.

If R is a decider for $E_{K \bot \alpha}$, then S is a decider for E_{LBA}. However, the existence of S contradict the undecidability of E_{LBA}, so it must be the case that $E_{K \bot \alpha}$ is undecidable. ■

We have the following corollary as a consequence of Theorem 2.4.

Corollary 2.5 *The AGM partial meet contraction is uncomputable for sets of LTL formulas.*

This result comes from the fact that, to define an AGM partial meet contraction for LTL, we need to check whether the set of remainder sets is empty, thus for LTL this construction would depend on an undecidable problem.

3 Restricted LTL Contraction

Although Theorem 2.4 shows that in general, constructing a partial meet contraction for LTL is not feasible, we now show that under certain restrictions, it is still possible to correctly define such an operator.

A possible restriction to this problem is to constrain it only to finite sets of temporal formulas. In this case, compactness follows trivially and monotonicity would be enough to ensure the correct definition of a partial meet contraction [11]. In the case of the representation of systems using temporal logics, however, infinite sets of formulas may appear even in fairly simple practical applications. Consider, for example, the model in Figure 6. The set $\{p, Xp, XXp, ...\}$ is a

very intuitive description of this system. Therefore, we do not want to pose such a strong restriction.

We show in what follows, that it is possible to define an operator of partial meet contraction with weaker restrictions and that could be used even for infinite sets of beliefs.

3.1 Restriction over K

Our restriction on K consists in constraining the problem to belief bases that could be *defined by a model*.

Definition 3.1 *Let $\mathcal{M}$ be a LTL model, a belief base K is defined by a model $\mathcal{M}$ if*

$$K = \{\phi \in \mathcal{L}_{LTL} \mid \mathcal{M} \models \phi\}$$

This representation is similar to that used by [14] where a belief base B is an infinite set of formulas characterized by single propositional formula ψ where $B = \{\phi \mid \psi \vdash \phi\}$.

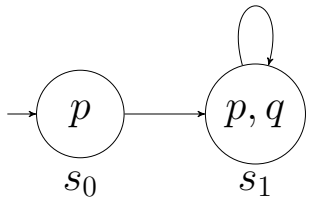

Figure 6: Example of a model.

3.2 Restriction over α

We restrict the input formula according to the type of temporal property it represents. In formal verification, the correctness of a system is ensured by systematically checking whether a given model satisfies or not a set of desired properties. Lamport [15] introduces two major class of properties, safety and liveness. Alpern and Schneider [3] show that every linear-time property can be written as an intersection of a safety and a liveness property. We then investigate the contraction problem according to the class of property expressed by the formula α.

One should notice that the restriction over α, alone, is not enough to ensure that an operator of partial meet contraction always exists. In fact, the formula $\alpha = \neg \mathrm{F} q_a$ Theorem 2.4 is one of the simplest types of linear-time properties

called *invariants* (Definition 3.2), nevertheless the problem is shown undecidable.

We investigate the decidability of the vacuity problem $E_{K\perp\alpha}$, assuming restrictions in both K and α, when K is a belief base defined by a model and α represents either an invariant, a safety property or a liveness property.[6]

3.3 Contraction of invariants

Invariants are properties expected to be true continuously in a model. An invariant describes a condition that must be satisfied in all reachable states of a model [4]. In LTL, invariants are expressed by formulas $\mathrm{G}\phi$ where ϕ is a propositional formula.

Definition 3.2 *Let $\mathcal{M}$ be a LTL models, a propositional formula ψ is an invariant in $\mathcal{M}$ if $\mathcal{M} \models \mathrm{G}\psi$.*

To address the vacuity problem for invariants, we use the concept of *unfolding models*. First, we need to define how to represent a model by what we call *frame encoding*.

Definition 3.3 *Let $\mathcal{M} = \langle AP, S, s_0, R, L\rangle$ be a LTL model, the frame encoding $\lceil\mathcal{M}\rceil$ of $\mathcal{M}$ is a string uv^+ such that*

1. *$u = s_0 s_1 ... s_{m-1}$ and $v = s_m s_{m+1} ... s_n$, and*
2. *for all $i < n$, $(s_i, s_{i+1}) \in R$, and $(s_n, s_m) \in R$*

The unfolding of a model $\mathcal{M}$ is a bissimilar model $\mathcal{M}'$ such that, for $\lceil\mathcal{M}\rceil = uv^+$, the frame encoding of $\mathcal{M}'$ is $\lceil\mathcal{M}'\rceil = uvv^+$. This intuitively means an "extraction" of states from the loop of $\mathcal{M}$ to the portion of states out of this loop.

Definition 3.4 *Let $\mathcal{M}$ be a LTL model and $\lceil\mathcal{M}\rceil = uv^+$ its frame encoding, a model $\mathcal{M}_k$ is a k-unfold of $\mathcal{M}$ if*

1. *$\lceil\mathcal{M}_k\rceil = zv^+$, where $|z| = |uv^k|$, and*
2. *$\mathcal{M}$ and $\mathcal{M}_k$ are bissimilar.*

Theorem 3.5 shows that, for a belief base K defined by a model and a formula $\alpha = \mathrm{G}\phi$ representing an invariant, if ϕ is not a tautology, the remainder set $K\perp\alpha$ is nonempty.

[6] Although invariants are a special case of safety properties, we choose to address it separately given its importance in formal verification applications.

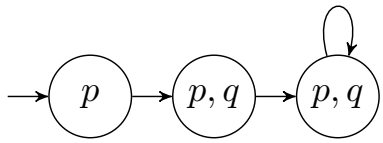

Figure 7: The 1-unfolding of $\mathcal{M}$ in Figure 1.

Theorem 3.5 *Let $\mathcal{M}$ be a LTL model, K a belief base defined by $\mathcal{M}$ and $\alpha = \mathrm{G}\psi$ a LTL formula, where ψ is an invariant in $\mathcal{M}$,*

$$\textit{if } \not\models \psi, \textit{ then } K\bot\alpha \neq \emptyset.$$

Proof. Let P_α be the set of propositional atoms of α and $\mathcal{M}_1$ be an 1-unfolding of $\mathcal{M}$ (This ensures that $\lceil \mathcal{M}_1 \rceil = uv^+$ and $|u| > 0$). We define a models $\mathcal{M}'$ from $\mathcal{M}_1$ such that $S' = S$, $R' = R$ and, for $B = \{l \mid l = p \text{ or } l = \neg p,\ p \in P_\alpha \text{ e } \mathcal{M} \models l\}$ and B' an element of $B\bot\phi$, L' is a labeling function such that

$$L'(p) = \begin{cases} L(p) \setminus \{s_0\}, & \text{if } p \in B \text{ and } p \notin B', \\ L(p) \cup \{s_0\}, & \text{if } \neg p \in B \text{ and } \neg p \notin B', \\ L(p), & \text{otherwise.} \end{cases}$$

The base $K' = \{\phi \in K \mid \mathcal{M}' \models \phi\}$ belongs to $K\bot\alpha$. By construction, $K' \subseteq K$ and $\phi \notin Cn(K')$, so $\alpha \notin Cn(K')$. It remains to show that there is no formula $\beta \in K$ such that $\beta \notin K'$ and $\alpha \notin Cn(K' \cup \{\beta\})$.

If $\beta = p$ or $\beta = \neg p$, since $B' \in B\bot\phi$, it holds that $\phi \in Cn(B' \cup \{\beta\})$. As we have $Cn(B') \subseteq Cn(K')$, also $\phi \in Cn(K' \cup \{\beta\})$. By construction, $\mathrm{X}\phi \in K'$, thus $\phi \wedge \mathrm{X}\phi \in Cn(K' \cup \{\beta\})$ and $\alpha = \mathrm{G}\phi \in Cn(K' \cup \{\beta\})$.

As an induction hypothesis, lets assume that for the formulas β_1 and β_2 it holds that $\phi \in Cn(K' \cup \{\beta_1\})$ and $\phi \in Cn(K' \cup \{\beta_2\})$.

If $\beta = \beta_1 \mathrm{R} \beta_2$, every model of $\beta_1 \mathrm{R} \beta_2$ satisfies β_2. Since $\beta_2 \in Cn(\beta_1 \mathrm{R} \beta_2)$ and, by induction hypothesis, $\phi \in Cn(K' \cup \{\beta_2\})$, it holds that $\phi \in Cn(K' \cup \{\beta\})$ and then $\alpha = \mathrm{G}\phi \in Cn(K' \cup \{\beta\})$.

If $\beta = \beta_1 \mathrm{U} \beta_2$, every model $\mathcal{M}''$ of $\beta_1 \mathrm{U} \beta_2$ satisfies either β_2, or $\beta_1 \wedge \mathrm{X}(\beta_1 \mathrm{U} \beta_2)$. By induction hypothesis, $\mathcal{M}''$ is a model of ϕ, thus holds that $\phi \in Cn(K' \cup \{\beta\})$ and then $\alpha = \mathrm{G}\phi \in Cn(K' \cup \{\beta\})$.

Therefore, for all $\beta \in K$, if $\beta \notin K'$, then $\alpha \in Cn(K' \cup \{\beta\})$. So, there is no set K'' such that $K' \subset K'' \subseteq K$ and $\alpha \notin Cn(K'')$, thus $K' \in K\bot\alpha$.

■

3.4 Contraction of safety properties

Safety properties are generalizations of invariants where we can describe the expected behavior of sequences of events in a model instead of analyzing each state independently. Typical safety properties includes mutual exclusion and deadlock freedom. In this sense, a safety property expresses that "something (bad) will not happen" [15]. We address then the vacuity problem of $K \perp \alpha$ when the input formula represents a safety property.

The formal definition of safety properties can be provided by means of counterexamples since each trace that refutes a safety property has a *finite* prefix that is a witness of the failure [3].

Definition 3.6 *We say that π is a finite counterexample for α in $\mathcal{M}$ if π is a finite path $s_0 \to s_1 \to ... \to s_n$ in $\mathcal{M}$ such that $\pi \models \neg\alpha$.*

Definition 3.7 *A LTL formula α expresses a safety condition if and only if for any LTL model $\mathcal{M}$, if $\mathcal{M} \not\models \alpha$, then there is a finite counterexample for α in $\mathcal{M}$.*

The definition stipulates that if a violation of a safety property occurs, there is an identifiable point at which this can be recognized [3].

In order to define a remainder set, we make use of what we call α-*suppression*: minimal changes in a model in order to introduce counterexamples to α.

Definition 3.8 *Let $\mathcal{M}$ be a LTL model, a model $\mathcal{M}'$ is an admissible α-suppression of $\mathcal{M}$ if*

1. $\lceil \mathcal{M}' \rceil = \lceil \mathcal{M} \rceil = uv^+$
2. $\mathcal{M}' \not\models \alpha$
3. *For all $p \in AP$*
   (a) $L'(p) = L(p)$ *if p is not a subformula of α,*
   (b) *for $v = v_0 v_1 ... v_n$, if $v_i \in L(p)$, then $v_i \in L'(p)$.*

This definition intuitively means that a model $\mathcal{M}'$ is an admissible α-suppression of $\mathcal{M}$ if it can be build from $\mathcal{M}$ by changing its labeling function only in those states out of the loop portion.

In Figure 8, $\mathcal{M}_1$ and $\mathcal{M}_2$ are examples of Gp-suppression in the model of Figure 7, $\mathcal{M}_3$ is not since it violates both conditions 3(a) and 3(b).

Definition 3.9 *Let K be a belief base defined by a model, we denote the set of elementary formulas of K by $\mathbb{E}_K$, where*

$$\mathbb{E}_K = \{\mathrm{X}^i \phi \in K \mid \phi = p \text{ or } \phi = \neg p, p \in AP \text{ and } i \geq 0\}$$

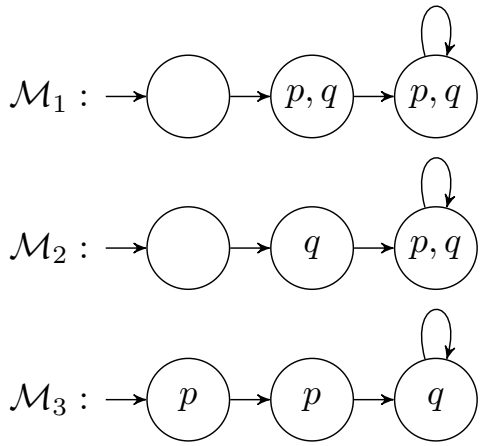

Figure 8: Modifications in the model of Figure 7.

We say that a $\mathcal{M}'$ is a minimal α-suppression of $\mathcal{M}$ if $\mathcal{M}' \not\models \alpha$ and maximally preserves the elementary formulas satisfied by $\mathcal{M}$.

Definition 3.10 *A model $\mathcal{M}'$ is a minimal α-suppression of $\mathcal{M}$ if*

1. *$\mathcal{M}'$ is an admissible α-suppression of $\mathcal{M}$*
2. *there is no admissible α-suppression $\mathcal{M}''$ of $\mathcal{M}$ such that $\{\phi \in \mathbb{E}_K \mid \mathcal{M}' \models \phi\} \subset \{\phi \in \mathbb{E}_K \mid \mathcal{M}'' \models \phi\}$*

We show in Theorem 3.11 that, for a belief base K defined by a model and formula α representing a safety property, if α is not a valid formula, the remainder set $K\perp\alpha$ is nonempty.

Theorem 3.11 *Let $\mathcal{M}$ be a LTL model, K a belief base defined by $\mathcal{M}$ and α a LTL formula expressing a safety property,*

$$\text{if } \not\models \alpha, \text{ then } K\perp\alpha \neq \emptyset.$$

Proof.

Let $\mathbf{M} = \{\mathcal{M}' \mid \mathcal{M}' \text{ minimal admissible } \alpha\text{-suppression of } \mathcal{M}_k\}$ where $\mathcal{M}_k$ is the smallest k-unfolding of $\mathcal{M}$ such that $\mathbf{M}$ is not empty[7]. For all $\mathcal{M}' \in \mathbf{M}$, it holds that $K' = \{\phi \in K \mid \mathcal{M}' \models \phi\} \in K\perp\alpha$.

By construction $K' \subseteq K$ and $\alpha \notin Cn(K')$, so it remains to show that for all $\beta \in K$, if $\beta \notin K'$, then $\alpha \in Cn(K' \cup \{\beta\})$.

[7]Since α is a safety property and thus it has finite counterexamples, it is guaranteed that, for some unfolding of $\mathcal{M}$, some an admissible change produces a model that does not satisfy α.

Let $\lceil \mathcal{M}' \rceil = uv^+$, as $\mathcal{M}'$ is an admissible change of some $\mathcal{M}_k$, for all formulas $\mathrm{X}^n\phi \in K$, where $n > |u|$, $\mathrm{X}^n\phi$ belongs to K'. (Since ϕ must hold in some loop state of $\mathcal{M}_k$ and these states are bissimilar to those in the loop of $\mathcal{M}'$). We show that it also holds for $n \leq |u|$.

Let $\beta = \mathrm{X}^n\phi \in K$ where $n = |u|$, we show that for $K'' = K' \cup \{\beta\}$, if $\beta \notin K'$, then $\alpha \in Cn(K'')$. (Um ponto chave que todo modelo de K'' bissimulvel por uma modificao admissvel $\mathcal{M}''$ de $\mathcal{M}$ tal que $\lceil \mathcal{M}' \rceil = \lceil \mathcal{M}'' \rceil$. Caso contrrio, existe $\phi = p$ ou $\phi = \neg p$ tal que $\mathrm{X}^i\phi \in K'$ e $\mathrm{X}^i\phi \notin K'$, para algum $i > |u|$ ou $p \notin P_\alpha$).

If $\beta = \mathrm{X}^n p$ or $\beta = \mathrm{X}^n \neg p$. Since $K' \subseteq K''$, all models of K'' must be bissimilar to some model $\mathcal{M}''$ that fulfills items 1 and 2 of Definition 3.8 with respect to $\mathcal{M}'$. If $\mathcal{M}'' \not\models \alpha$, the model $\mathcal{M}''$ is an admissible α-suppression to $\mathcal{M}_k$, since $\mathcal{M}'' \models \beta$ and $\beta \in \mathbb{E}_K$, $\mathcal{M}'$ can not be a minimal α-suppression of $\mathcal{M}_k$, a contradiction. Thus, must be the case where $\mathcal{M}'' \models \alpha$ and $\alpha \in Cn(K'')$.

As an induction hypothesis, lets assume that for the formulas β_1 and β_2 it holds that $\alpha \in Cn(K' \cup \{\mathrm{X}^n\beta_1\})$ and $\phi \in Cn(K' \cup \{\mathrm{X}^n\beta_2\})$.

If $\beta = \mathrm{X}^n(\beta_1 \mathrm{R} \beta_2)$, every model of $\mathrm{X}^n(\beta_1 \mathrm{R} \beta_2)$ satisfies $\mathrm{X}^n\beta_2$, thus $\mathrm{X}^n\beta_2 \in Cn(\beta)$. Since, by induction hypothesis, $\alpha \in Cn(K' \cup \{\mathrm{X}^n\beta_2\})$ and $Cn(K' \cup \{\mathrm{X}^n\beta_2\}) \subseteq Cn(K'')$, it holds that $\alpha \in Cn(K'')$.

If $\beta = \beta_1 \mathrm{U} \beta_2$, every model $\mathcal{M}''$ of $\mathrm{X}^n(\beta_1 \mathrm{U} \beta_2)$ satisfies either $\mathrm{X}^n\beta_2$, or $\mathrm{X}^n\beta_1 \wedge \mathrm{X}^{n+1}(\beta_1 \mathrm{U} \beta_2)$. In both cases, if $\mathcal{M}''$ is also a model for K', by induction hypothesis, $\mathcal{M}''$ is a model of α. Thus, it holds that $\alpha \in Cn(K'')$.

Therefore, for all $\beta \in K$, if $\beta \notin K'$, then $\alpha \in Cn(K' \cup \{\beta\})$. So, there is no set K'' such that $K' \subset K'' \subseteq K$ and $\alpha \notin Cn(K'')$, thus $K' \in K \perp \alpha$.

■

4 Undecidability of partial meet for other logics

Undecidability of $E_{K\perp\alpha}$ holds also for other temporal and modal logics, as CTL [5] and PDL [7, 13].

For CTL, the result can be shown with the same proofs presented here for LTL, only with minor changes in some definition to its correspondent form in CTL. For example, a LBA-equivalent formula in CTL is given by the same construction of Definition 2.2, except for the replacement of the temporal operators X and G by AX and AG, respectively.

We can show that a LBA M accepts w if and only if the CTL formula $\phi^{\mathrm{M}}_{|w|} \wedge \mathrm{EF}q_a$ is satisfiable. This property can be used as in Theorem 2.4, replacing $\neg \mathrm{F}q_a$ by $\neg \mathrm{EF}q_a$, to show that $E_{K\perp\alpha}$ is undecidable for CTL.

Similarly, we can show that $E_{K\perp\alpha}$ is undecidable for PDL. In this case, the construction of a LBA-equivalent formula is done by replacing the temporal

operators X and G by $[\pi]$ and $[\pi^*]$, respectively, where π is an arbitrary PDL program.

We can show that a LBA M accepts w if and only if the PDL formula $\phi^{\mathrm{M}}_{|w|} \wedge \langle\pi^*\rangle q_a$ is satisfiable. As before, this property can be used in a proof similar to that of Theorem 2.4, replacing $\neg \mathrm{F} q_a$ by $\neg\langle\pi^*\rangle q_a$, to show that $E_{K\perp\alpha}$ is also undecidable for PDL.

5 Conclusions

We show that the emptiness problem of remainder sets is undecidable by a reduction from the emptiness problem of linear bounded automata. This result implies that it is in general impossible to define an AGM partial meet contraction for sets of LTL formulas. This undecidability result is derived from the lack of compactness in LTL and its implications on how to define the satisfiability of sets of LTL formulas, which plays a central role on the definition of remainder sets.

We discuss possible ways to address this problem and show that by limiting the sets of LTL formulas to those defined by a model, and restricting the input formulas to those representing safety properties, we can avoid the undecidability result.

We show that, with the proposed restrictions, if the desired property is satisfiable, it is possible to find at least one element of the remainder set and then correctly perform partial meet contraction in LTL. We give a procedure that makes use of the finite nature of counterexamples for safety properties in order to find at least a possible candidate for the contraction result, even for non-compact LTL bases.

Recently, Van Zee et al.[22] have presented a logic for revision of temporal belief bases, together with a construction for revision based on the minimality of models. One can see that they avoid undecidability by considering beliefs only up to a certain point in time. A similar idea was proposed by Finger and Wassermann [6], where instead of full model checking, a bounded check was used. This corresponds to limiting the future and checking the models only up to a certain time. Along with [10, 19], these results support our claim that, despite the undecidability result, AGM theory could still be successfully applied to several domains involving temporal logic once we restrict the language appropriately.

Future work includes exploring other constructions for contraction such as safe [1] and kernel contractions [12], which are based on selecting minimal subsets of the belief base implying the input, instead of the maximal subsets not implying it.

Acknowledgments.

The first author was funded by grant #2010/15392-3, São Paulo Research Foundation (FAPESP). The second author was partially supported by the Brazilian Research Council (CNPq) grant #447178/2014-8.

References

[1] Carlos Alchourrón and David Makinson. On The Logic of Theory Change: Safe Contraction. *Studia Logica*, 44:405–422, 1985.

[2] Carlos E Alchourron, Peter Gärdenfors, and David Makinson. On The Logic of Theory Change: Partial Meet Contraction and Revision Functions. *Journal of Symbolic Logic*, 50(2):510–530, 1985.

[3] Bowen Alpern and Fred B. Schneider. Defining liveness. *Information Processing Letters*, 21(4):181–185, October 1985.

[4] Christel Baier and Joost-pieter Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[5] Edmund M Clarke and E Allen Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In Dexter Kozen, editor, *Logics of Programs*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer Berlin Heidelberg, 1982.

[6] Marcelo Finger and Renata Wassermann. Revising specifications with CTL properties using bounded model checking. In *Brazilian Symposium on Artificial Intelligence*, LNAI. Springer, 2008.

[7] Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194 – 211, 1979.

[8] Giorgos Flouris. *On Belief Change and Ontology Evolution*. PhD thesis, University of Crete, 2006.

[9] Dov M. Gabbay, Ian Hodkinson, and Mark Mark A. Reynolds. *Temporal logic : mathematical foundations and computational aspects. Volume 1*. Oxford logic guides. Clarendon Press, Oxford, 1994. Index : p. 641-653.

[10] Paulo Guerra and Renata Wassermann. Two AGM-style characterizations of model repair. In *Knowledge Representation and Reasoning Conference (KR)*, pages 645–646, 2018.

[11] Sven O Hansson and Renata Wassermann. Local change. *Studia Logica*, 70(1):49–76, 2002.

[12] Sven Ove Hansson. Kernel contraction. *The Journal of Symbolic Logic*, 59:845–859, 9 1994.

[13] David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic logic*. MIT press, 2000.

[14] Hirofumi Katsuno and Alberto O Mendelzon. On the Difference Between Updating a Knowledge Base and Revising it. In *Proc. of KR*, volume 52, pages 387–395. Morgan Kaufmann, 1991.

[15] L. Lamport. Proving the correctness of multiprocess programs. *IEEE Trans. Softw. Eng.*, 3(2):125–143, March 1977.

[16] Orna Lichtenstein and Amir Pnueli. Checking that finite state concurrent programs satisfy their linear specification. *Proceedings of the 12th ACM SIGACT-SIGPLAN symposium on Principles of programming languages - POPL '85*, pages 97–107, 1985.

[17] J. Myhill. *Linear Bounded Automata*. Report // University of Pennsylvania. Wright Air Development Division, Air Research and Technology Command, United States Air Force, 1960.

[18] Amir Pnueli. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, SFCS '77, pages 46–57, Washington, DC, USA, 1977. IEEE Computer Society.

[19] Jandson Ribeiro, Abhaya Nayak, and Renata Wassermann. Towards belief contraction without compactness. In *Knowledge Representation and Reasoning Conference (KR)*, pages 287–296, 2018.

[20] Márcio Moretto Ribeiro. *Belief Revision in Non-Classical Logics.* Springer Briefs in Computer Science. Springer, 2013.

[21] M. Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 2005.

[22] Marc Van Zee, Dragan Doder, Mehdi Dastani, and Leendert Van Der Torre. AGM revision of beliefs about action and time. In *Proc. International Conference on Artificial Intelligence*, pages 3250–3256. AAAI Press, 2015.

[23] Renata Wassermann. On AGM for non-classical logics. *Journal of Philosophical Logic*, 40(2):271–294, 2011.

Paulo T. Guerra
Federal University of Ceará (UFC)
Campus Quixadá
Av. José de Freitas Queiroz, 5003 - Cedro - 63902-580, Quixadá, Ceará, Brazil
E-mail: paulodetarso@ufc.br

Renata Wassermann
Institute of Mathematics and Statistics
University of São Paulo (USP)
Rua do Matão, 1010 - CEP 05508-090 - São Paulo, SP, Brazil
E-mail: renata@ime.usp.br