Model Checking Vs. Generalized Model Checking: Semantic Minimizations for Temporal Logics

Patrice Godefroid
Bell Laboratories, Lucent Technologies
2701 Lucent Lane, Lisle, IL 60532
god@bell-labs.com

Michael Huth
Department of Computing, Imperial College London
South Kensington campus, London, SW7 2AZ, England
M.Huth@doc.imperial.ac.uk

Abstract

Three-valued models, *in which properties of a system are either true, false or unknown, have recently been advocated as a better representation for reactive program abstractions generated by automatic techniques such as predicate abstraction. Indeed, for the same cost, model checking three-valued abstractions can be used to both prove and disprove any temporal-logic property, whereas traditional conservative abstractions can only prove universal properties. Also, verification results can be more precise with* generalized model checking, *which checks whether there exists a concretization of an abstraction satisfying a temporal-logic formula. Since generalized model checking includes satisfiability as a special case (when everything in the model is unknown), it is in general more expensive than traditional model checking. In this paper, we study how to reduce generalized model checking to model checking by a temporal-logic formula transformation, which generalizes a transformation for propositional logic known as* semantic minimization *in the literature. We show that many temporal-logic formulas of practical interest are self-minimizing, i.e., are their own semantic minimizations, and hence that model checking for these formulas has the same precision as generalized model checking.*

1 Introduction

Abstraction is key to extend the scope of formal verification to systems with infinite or very large state spaces, as illustrated by recent work on software model checking using predicate abstraction in tools such as SLAM [1] and BLAST [14], among others. The relation between a concrete system and its abstraction is traditionally a simulation, which allows the verification of universal properties only.

Recently, a new version of the "abstract-check-refine" process of [1, 14] has been advocated [12]:

1. Abstract: compute a *3-valued* abstraction M_A for which properties of the concrete system M_C are either $true$, $false$ or $\perp$ (denoting "unknown").
2. Check: given *any* temporal-logic formula ϕ,
   (a) (3-valued model checking) check whether M_A satisfies ϕ; if the result is $true$ ($false$) then stop, the property is proved (resp. disproved) for M_C; if the result is $\perp$ (unknown), go to Step 2(b).
   (b) (generalized model checking) check whether there exist concretizations M_T and M_F of M_A such that M_T satisfies ϕ and M_F satisfies $\neg\phi$; if M_F (resp. M_T) does not exist, ϕ is proved (resp. disproved) for M_C; otherwise go to Step 3.
3. Refine: refine M_A (possibly using a counter-example found in Step 2). Then go to Step 1.

This new procedure strictly generalizes the traditional one in several ways: any temporal-logic formula can be checked (not just universal properties), and all correctness proofs and counter-examples obtained are guaranteed to be sound (i.e., hold on M_C) for any property. Remarkably, Steps 1, 2(a) and 3 can be done with 3-valued models at the same cost as with traditional conservative 2-valued models [4, 11, 12]. In contrast, generalized model checking (Step 2(b)) can be more expensive than model checking [4], since it includes satisfiability as a special case (when everything in the model is unknown), but it can also be more precise. For instance, consider the program P

```
program P() {
  x,y = 1,0;
  x,y = 2*f(x),f(y);
  x,y = 1,0;
}
```

where `x` and `y` denote `int` variables, `f : int -> int` denotes some unknown function, and the notation "`x,y = 1,0`" means variables `x` and `y` are simultaneously assigned values `1` and `0`, respectively. Consider the LTL formula $\phi = \mathsf{F}q_y \wedge \mathsf{G}(q_x \vee \neg q_y)$ with the two predicates

q_x : "is x odd?" and q_y : "is y odd?", and where F means "eventually" while G means "always." (See [8] for a definition of the temporal logics used in this paper.) As shown in [12], model checking ϕ against P returns the value "unknown," while generalized model checking can prove that no concretization of program P (i.e., f) satisfies ϕ.

In this paper, we study for which temporal-logic formulas generalized model checking can improve precision over model checking. More generally, we study how to reduce generalized model checking for a model M and a formula ϕ to model checking $M \models \phi'$ via a temporal-logic formula transformation $\phi \mapsto \phi'$ and independently of any model M. This reduction generalizes a transformation for propositional logic [2] called *semantic minimization* in [28]. We show that propositional modal logic and the modal mu-calculus [22] (μL) are closed under semantic minimizations, but that the logics CTL, LTL and CTL* are not. We study the complexity of computing semantic minimizations, and show that the problem is as hard as the satisfiability problem. We then provide sufficient syntactic conditions for the efficient identification of many semantically self-minimizing temporal-logic formulas, i.e., formulas that are their own semantic minimizations. We observe that, for self-minimizing formulas, model checking has always the same precision as generalized model checking, and show that many temporal-logic formulas of practical interest are self-minimizing.

Related Work Blamey studies partial-valued logics and their applications to linguistics and model theory in [2]. In particular, he shows in Theorem I.3.3 of [2] that all formulas of propositional logic have (in the terminology of [28] adopted in this paper) a semantic minimization. The proof rests on the information-theoretic monotonicity of van Fraassen's super-valuational meaning [31], which corresponds to the thorough semantics in this paper.

Reps et al. [28] use BDD-based prime-implicant algorithms for an efficient implementation of computing semantic minimizations for formulas of propositional logic.

Temporal logics for which satisfiability is efficiently decidable have been widely studied in the literature. We mention work by Demri & Schnoebelen [6], Emerson et al. [9], and Henzinger et al. [15]. In contrast, generalized model checking for branching-time logics can be reduced to a satisfiability problem of the form SAT($\phi_M \wedge \phi$) [4], where ϕ_M is the characteristic formula of a model M, and our work can be viewed in this context as seeking to reduce satisfiability checks SAT($\phi_M \wedge \phi$) to model checks $M \models \phi'$ independently of M and ϕ_M.

Outline of paper In Section 2 we review the notions of partial models, their refinement, and their compositional and thorough temporal-logic semantics. We define and prove basic properties of semantic minimization for temporal logics in Section 3. The existence of semantic minimization for temporal logics without and with fixed points is the subject of Sections 4 and 5, respectively. In Section 6 we detect self-minimizing formulas using automata-theoretic techniques, provide grammars that efficiently construct such formulas, and show that these grammars are "optimal" and cover a wide range of specification patterns. Applications of self-minimization are briefly featured in Section 7 and Section 8 concludes.

2 Partial Kripke Structures and Refinement

Let $AP \neq \{\}$ be a finite set of atomic propositions. We endow a set $\{\mathit{true}, \bot, \mathit{false}\}$ of truth values with two partial orders: the *information ordering* $\leq_I$, where $\bot$ is the least element and true and false are maximal elements: $\bot \leq_I \mathit{true}, \mathit{false}$; and the *truth ordering* $\leq_T$ with a strict chain $\mathit{false} <_T \bot <_T \mathit{true}$. We identify the models of interest.

Definition 1 *1. A* partial Kripke structure *[3] $M = (S, R, L)$ consists of a finite nonempty set of states S, a total transition relation $R \subseteq S \times S$, and a labelling function $L\colon S \times AP \to \{\mathit{true}, \bot, \mathit{false}\}$; $L(s, q)$ is the truth value of q at state s. We refer to M as a "model" if this is convenient. A* pointed *model (M, s) is a model M with a distinguished start state s. The tree structure of (M, s) is called a* 3-valued labelled tree.

2. For models $M_i = (S_i, R_i, L_i)$ with $i = 1, 2$ the completeness preorder *[3] is the greatest relation $\preceq \subseteq S_1 \times S_2$ such that $s_1 \preceq s_2$ implies*

(a) $\forall\, q \in AP\colon L_1(s_1, q) \leq_I L_2(s_2, q)$,
(b) $\forall\, (s_1, s_1') \in R_1\, \exists\, (s_2, s_2') \in R_2\colon s_1' \preceq s_2'$, and
(c) $\forall\, (s_2, s_2') \in R_2\, \exists\, (s_1, s_1') \in R_1\colon s_1' \preceq s_2'$.

The partiality in models resides in labels $L(s, q) = \bot$ and the completeness preorder states that such labels may be completed into true or false in a co-inductive manner. (See [13, 3] for other ways to model partiality.) This partiality leads to two ways of checking properties written in propositional modal logic (PML), whose syntax is

$$\phi ::= q \mid \neg\phi \mid \phi \wedge \phi \mid \mathsf{EX}\phi \qquad (1)$$

where q ranges over AP. We write $\phi_1 \vee \phi_2$ for $\neg(\neg\phi \wedge \neg\phi_2)$, $\phi_1 \to \phi_2$ for $(\neg\phi_1 \vee \phi_2)$, $\phi_1 \leftrightarrow \phi_2$ for $(\phi_1 \to \phi_2) \wedge (\phi_2 \to \phi_1)$, and $\mathsf{AX}\phi$ for $\neg\mathsf{EX}\neg\phi$ subsequently.

A *compositional* semantics exploits algebraic structure on $\{\mathit{false}, \bot, \mathit{true}\}$: Kleene's negation [21] $\mathit{neg}(\mathit{false}) = \mathit{true}$, $\mathit{neg}(\mathit{true}) = \mathit{false}$, and $\mathit{neg}(\bot) = \bot$; a maximum max in the truth ordering $\leq_T$ (where $\mathit{max}(\emptyset) = \mathit{false}$); and a minimum min in the truth ordering. We define the truth value $[(M, s) \models \phi]$ of this compositional semantics at state s in model M as in [3]:

$$
\begin{aligned}
[(M,s) \models q] &= L(s,q) \\
[(M,s) \models \neg\phi] &= neg([(M,s) \models \phi]) \\
[(M,s) \models \phi_1 \wedge \phi_2] &= min_{i=1,2}[(M,s) \models \phi_i] \\
[(M,s) \models \mathsf{EX}\phi] &= max_{(s,s')\in R}[(M,s') \models \phi]\,.
\end{aligned} \tag{2}
$$

Similar 3-valued semantics can be defined for more expressive logics, e.g., LTL, CTL, CTL* and μL [4].

A second, *non-compositional* semantics is based on the completeness preorder. Since a Kripke structure is a partial Kripke structure whose labeling function L satisfies $L^{-1}(\{\bot\}) = \{\}$, we define $\mathcal{C}[M,s]$ as the set of pairs (K,t) where K is a Kripke structure with distinguished state t and $(M,s)\preceq(K,t)$. Elements of $\mathcal{C}[M,s]$ are called the *completions* of (M,s) [4]. The *generalized model-checking (GMC) problem* asks whether there exists a completion of a given partial Kripke structure that satisfies a given temporal-logic property. It is worth noticing that GMC generalizes both model checking (when the model is complete) and satisfiability (when the model is $(M_\bot, s_\bot)$, with a sole state $s_\bot$, sole transition $(s_\bot, s_\bot) \in R$, and $L(s_\bot, q) = \bot$ for all $q \in AP$ satisfying $(M_\bot, s_\bot)\preceq(N,t)$ for all (N,t)). The GMC problem is in turn used to define the non-compositional *thorough interpretation* [4]. Subsequently we write TL for any temporal logic (PML, LTL, CTL, etc.) with a semantics over 2-valued Kripke structures.

Definition 2 *Let (M,s) be a pointed model and $\phi \in$ TL.*

1. *The decision problem of* generalized model checking *[4]* $\mathrm{GMC}(M,s,\phi)$ *returns "$true$" if there is a completion of (M,s) satisfying ϕ, and "$false$" otherwise.*
2. *The* thorough interpretation *[4]* $[(M,s) \models \phi]_t$ *has value "$true$" if all $(K,k) \in \mathcal{C}[M,s]$ satisfy ϕ; "$false$" if no $(K,k) \in \mathcal{C}[M,s]$ satisfies ϕ; and $\bot$ otherwise.*

Kleene's alignment operator [21] $\sqcup\colon \{false, \bot, true\} \times \{false, \bot, true\} \to \{false, \bot, true\}$ returns $false$ if both arguments are $false$, returns $true$ if both arguments are $true$, and returns $\bot$ otherwise. For any formula ϕ of a *branching-time* temporal logic (we write BTL for any such logic subsequently), this operator connects the thorough interpretation to the GMC problem as

$$[(M,s) \models \phi]_t = neg(\mathrm{GMC}(M,s,\neg\phi)) \sqcup \mathrm{GMC}(M,s,\phi)$$

for all pointed models (M,s). (The formalization for LTL is omitted due to lack of space and is slightly different as $[(M,s) \models \phi]_t$ is captured by one instance of GMC [12].)

A similar decomposition for the compositional semantics for $m \in \{p, o\}$, $\neg p = o$, and $\neg o = p$ is [4]

- $(M,s) \models^p q$ iff $L(s,q) = true$
- $(M,s) \models^o q$ iff $L(s,q) \neq false$
- $(M,s) \models^m \neg\phi$ iff $(M,s) \not\models^{\neg m} \phi$
- $(M,s) \models^m \phi \wedge \psi$ iff $(M,s) \models^m \phi$ and $(M,s) \models^m \psi$
- $(M,s) \models^m \mathsf{EX}\phi$ iff $\exists (s,\alpha,s') \in R$, $(M,s') \models^m \phi$

where $\models^p$ ($\models^o$) is a *pessimistic (optimistic)* interpretation as it maps all $\bot$-labelings in M to $false$ (respectively, $true$) [4]. In [18], the superscript p is written a with the intent that $(M,s) \models^a \phi$ means ϕ is **a**sserted to hold in all completions of (M,s), while the superscript o is written c and $(M,s) \models^c \phi$ states that ϕ may be **c**onsistent in some completion of (M,s). For all M, s, and ϕ, we have $[(M,s) \models \phi] = ((M,s) \models^p \phi) \sqcup ((M,s) \models^o \phi)$ by [4, 19]. The semantics $[(M,s) \models \phi]$ is sound with respect to $[(M,s) \models \phi]_t$ [4]:

$$\forall (M,s), \phi\colon [(M,s) \models \phi] \leq_I [(M,s) \models \phi]_t\,. \tag{3}$$

So if $[(M,s) \models \phi]$ discovers that all (value $true$), respectively no (value $false$), completions of (M,s) satisfy ϕ, this is indeed so. However, the converse does not hold, as illustrated by the following example.

Example 1 *Let ϕ be $\mathsf{EX}q_1 \wedge (\mathsf{EX}q_2 \vee \neg\mathsf{EX}q_2)$, which is neither a tautology nor unsatisfiable. Let M have one state s, one transition $(s,s) \in R$, $L(s,q_2) = \bot$, and $L(s,q_1) = true$. Then $[(M,s) \models \phi] = \bot$ but $[(M,s) \models \phi]_t = true$.*

Thus, the compositional semantics $[(M,s) \models \phi]$ can be less precise than the thorough semantics $[(M,s) \models \phi]_t$. But computing $[(M,s) \models \phi]_t$ is generally more expensive. Indeed, computing $[(M,s) \models \phi]$ can be reduced to two standard model checking problems while computing $[(M,s) \models \phi]_t$ may require solving two GMC problems [4, 12]. For PL, PML and CTL, model checking can be solved in linear time, while GMC is NP-complete, PSPACE-complete and EXPTIME-complete (respectively) [4], which match the complexity of the satisfiability problems for these respective logics. For LTL, GMC is EXPTIME-complete [4], while model checking and satisfiability are "only" PSPACE-complete.

3 Semantic Minimization

We ask for which temporal-logic formulas ϕ the analysis $[(M,s) \models \phi]$ is as precise as $[(M,s) \models \phi]_t$, for all pointed models (M,s). If this is not the case, we ask whether this loss of precision can be restored through a change of ϕ into a new formula ϕ', *uniformly* for all (M,s), without changing the compositional semantics. Such a new formula ϕ' is called a *semantic minimization* of ϕ in [28], in the context of PL. We generalize this concept to temporal logics.

Definition 3 *Let ϕ be a formula of BTL.*

1. *An* optimistic semantic minimization ϕ^o *of* ϕ *is a formula of BTL such that, for all pointed models* (M, s),
$$(M, s) \models^o \phi^o \text{ iff } \mathrm{GMC}(M, s, \phi)\,. \quad (4)$$
2. *A* pessimistic semantic minimization ϕ^p *of* ϕ *is a formula of BTL with, for all pointed models* (M, s),
$$(M, s) \models^p \phi^p \text{ iff } neg(\mathrm{GMC}(M, s, \neg\phi))\,. \quad (5)$$
3. *A* semantic minimization *of* ϕ *is a pair* (ϕ^p, ϕ^o) *of formulas of BTL such that* ϕ^p *and* ϕ^o *are pessimistic and optimistic semantic minimizations of* ϕ*, respectively.*
4. *Formula* ϕ *is* optimistically (pessimistically) self-minimizing *iff* ϕ *is an optimistic (pessimistic) semantic minimization of itself (respectively). We say that* ϕ *is* semantically self-minimizing *if it is both optimistically and pessimistically self-minimizing.*
5. *We write* $\phi\#\psi$ *to state that* ϕ *and* ψ *share no* $q \in AP$*, and write* $\phi_\exists$ $(\phi_\forall)$ *if the negation normal form of* ϕ *is known to be an existential (resp., universal) one.*

Below we use $\phi\#\psi$ for proving the self-minimization of some patterns. E.g. if $\phi, \psi \in AP$ and $\phi \neq \psi$, then $\phi\#\psi$ and $\phi = \phi_\exists = \phi_\forall$. By (3) we may establish that ϕ is optimistically (pessimistically) self-minimizing by proving the only-if-part of (4) (the if-part of (5), respectively) only. If ϕ has a semantic minimization (ϕ^p, ϕ^o), all thorough checks of ϕ can be reduced to two compositional checks such that this reduction is *independent* of the pointed model:

$$\forall M, s\colon [(M, s) \models \phi]_t = ((M, s) \models^p \phi^p) \sqcup ((M, s) \models^o \phi^o)\,.$$

In particular, $[(M, s) \models \phi]_t = [(M, s) \models \phi]$ whenever ϕ is semantically self-minimizing.

To illustrate the nature of the problem of finding pessimistic and optimistic semantic minimizations for temporal-logic formulas, we now present some formulas and their semantic minimizations. We write $(\neg\phi)^p = \neg\phi^o$ for "the negation of an optimistic semantic minimization of ϕ is a pessimistic semantic minimization of $\neg\phi$" etc.

Proposition 1 *Let* $\phi, \psi, \eta, \gamma \in$ *BTL. Then*

1. $(\neg\phi)^p = \neg\phi^o$ *and* $(\neg\phi)^o = \neg\phi^p$,
2. $(\phi \wedge \psi)^p = \phi^p \wedge \psi^p$ *and* $(\phi \vee \psi)^o = \phi^o \vee \psi^o$,
3. $(\mathsf{EX}\phi)^o = \mathsf{EX}\phi^o$ *and* $(\mathsf{EX}\phi)^p = \mathsf{EX}\phi^p$*, and*
4. $(\mathsf{AX}\phi)^o = \mathsf{AX}\phi^o$ *and* $(\mathsf{AX}\phi)^p = \mathsf{AX}\phi^p$.
5. *If* $\phi_\exists\#\psi_\exists$ *and* $\eta_\forall\#\gamma_\forall$*, then* $(\eta_\forall \vee \gamma_\forall)^p = \eta_\forall^p \vee \gamma_\forall^p$ *and* $(\phi_\exists \wedge \psi_\exists)^o = \phi_\exists^o \wedge \psi_\exists^o$.

Example 2 *Semantically self-minimizing are all* $q \in AP$ *and literals (by Proposition 1(1)). By item 2, all* $q \vee \neg q$ *are optimistically self-minimizing but do not meet the assumptions of item 5 and are indeed not pessimistically self-minimizing. So* $q \wedge \neg q$ *is pessimistically but not optimistically self-minimizing by item 1. By items 4 and 5,* $\mathsf{AX}q_1 \rightarrow \mathsf{EX}\neg q_2$ *is semantically self-minimizing.*

The absence of the dual of item 2 above for formulas with shared atomic propositions makes the notion of semantic minimization non-trivial in general. Semantic minimizations are invariant under 2-valued equivalence.

Proposition 2 *Let* ϕ *and* ϕ' *be semantically equivalent over 2-valued models. If* ψ *is an optimistic (pessimistic) semantic minimization for* ϕ*, then* ψ *is also an optimistic (pessimistic) semantic minimization for* ϕ' *(respectively).*

The formulas of propositional logic (PL) are obtained from the grammar for PML by dropping the clause for $\mathsf{EX}\phi$ in (1). Models are 3-valued functions $L\colon AP \rightarrow \{false, \bot, true\}$. We write 3^{AP} for the set of all such models and 2^{AP} for the set of those models that do not have $\bot$ in their image (where we occasionally identify $L \in 2^{AP}$ with its characteristic set $L^{-1}(true)$). For PL, the completeness preorder of Definition 1(2) between models L and L' is the point-wise one: $L \preceq L'$ iff for all $q \in AP$, $L(q) \leq_I L'(q)$.

Blamey [2] shows that the compositional semantics in (2) applied to models for PL is functionally complete for all functions $f\colon \{false, \bot, true\}^n \rightarrow \{false, \bot, true\}$ ($n \geq 1$) that are monotone with respect to the information ordering $\leq_I$. As $f = [L \models \phi]_t$ is monotone in that way, one can secure the following, implicit in Theorem I.3.3 of [2] and more explicit in [28], in our terminology.

Proposition 3 ([2, 28]) *Every formula* ϕ *of PL has an optimistic semantic minimization* ϕ^o *in PL and, by Prop. 1(1), a pessimistic semantic minimization* ϕ^p *in PL as well.*

4 Semantic Minimization for PML

We now generalize Proposition 3 from PL to PML. This proof relies on the bounded modal depth of PML formulas and determines the model complexity of GMC for PML.

To prove the existence of semantic minimizations, we use automata-theoretic techniques. We refer to [24] for notions of automata theory that will be used.

Theorem 1 *Every formula of PML has a semantic minimization in PML.*

Proof: (Sketch) Given a formula $\phi \in$ PML, we describe how to construct an optimistic semantic minimization $\phi^o \in$ PML. (The case for pessimistic semantic minimizations is similar as $\phi^p = \neg(\neg\phi)^o$ by Proposition 1(1).) The idea of the construction is simple: define a tree automaton A_ϕ^3 that accepts a 3-valued labeled tree T^3 *iff* there exists a 2-valued tree T such that $T^3 \preceq T$ and T satisfies ϕ; and then translate this automaton back into a PML formula ϕ^o.

To construct A_ϕ^3, we first build a nondeterministic tree automaton $A_\phi = (2^{AP}, D, S, s_0, \rho, F)$ that accepts exactly the computation trees satisfying ϕ [24, 26]. A_ϕ has 2^{AP}

for input alphabet, a set S of states (which may contain $O(2^{O(|\phi|)})$ states), an initial state $s_0 \in S$, a finite set $D \subset N$ of arities, a transition function $\rho(s,a,k) \subseteq S^k$ for each $s \in S$, $a \in 2^{AP}$ and $k \in D$, and an acceptance condition F. Given A_ϕ, we define the desired nondeterministic tree automaton $A^3_\phi = (3^{AP}, D, S, s_0, \rho^3, F)$ that accepts exactly the 3-valued computation trees of partial Kripke structures for which there exists a 2-valued Kripke structure completion satisfying ϕ. For any $a^3 \in 3^{AP}$ the transition function ρ^3 of A^3_ϕ is defined as

$$\rho^3(s, a^3, k) = \bigcup_{a^3 \preceq a} \rho(s, a, k)\,. \tag{6}$$

By construction and definition of the completeness preorder $\preceq$, it is immediate that A^3_ϕ accepts a 3-valued tree T^3 iff there exists a 2-valued tree T such that $T^3 \preceq T$ and T is accepted by A_ϕ. As $\phi \in$ PML, A_ϕ cannot distinguish trees at depths greater than $|\phi|$. By construction, this property carries over to A^3_ϕ, and allows for re-encoding A^3_ϕ as a $\phi^o \in$ PML of modal depth $O(|\phi|)$. ∎

We illustrate the construction of A^3_ϕ and ϕ^o. Below, "$\rho(s_0, a, k) = \mathit{true}$" means "$\rho(s_0, a, k) = \{(s_T, \ldots, s_T)\}$ with $s_T \in F$ and $\rho(s_T, a, k) = \{(s_T, \ldots, s_T)\}$" ($s_T$ is an *accepting* sink state), while "$\rho(s_0, a, k) = \mathit{false}$" means "$\rho(s_0, a, k) = \{(s_F, \ldots, s_F)\}$, $s_F \notin F$, and $\rho(s_F, a, k) = \{(s_F, \ldots, s_F)\}$" ($s_F$ is a *non-accepting* sink state).

Example 3 (Tautology in PL) *Let $\phi = q \vee \neg q$ and $AP = \{q\}$. For A_ϕ, $\rho(s_0, \{q\}, k) = \mathit{true}$ and $\rho(s_0, \{\}, k) = \mathit{true}$. Thus, by definition, A^3_ϕ is such that $\rho^3(s_0, \{q \mapsto \perp\}, k) = \mathit{true}$. Thus, ϕ^o is semantically equivalent to true.*

Example 4 (Non self-minimizing PML formula) *Consider the PML formula $\phi = \mathsf{EX}q_1 \wedge \mathsf{AX}(\neg q_1 \vee q_2)$, whose sub-formulas are neither tautologies nor unsatisfiable. Now A_ϕ is such that, for any $a \in 2^{AP}$, $\rho(s_0, a, k) = \{(s_1, \ldots, s_k) \mid s_i = s' \text{ and } s_j = s'' \ \forall j \neq i\}$ (intuitively, s' takes care of the EX case, while s'' corresponds to the default AX case); $\rho(s', a, k) = \mathit{true}$ if $a(q_1) = a(q_2) = \mathit{true}$, and $\rho(s', a, k) = \mathit{false}$ otherwise; while $\rho(s'', a, k) = \mathit{true}$ if $a(q_1) = \mathit{false}$ or $a(q_2) = \mathit{true}$, and $\rho(s'', a, k) = \mathit{false}$ otherwise. Therefore, A^3_ϕ is such that $\rho^3(s_0, a^3, k) = \{(s_1, \ldots, s_k) \mid s_i = s' \text{ and } s_j = s'' \ \forall j \neq i\}$; $\rho^3(s', a^3, k) = \mathit{true}$ if $a^3(q_1) \neq \mathit{false}$ and $a^3(q_2) \neq \mathit{false}$, but $\rho^3(s', a^3, k) = \mathit{false}$ otherwise; $\rho^3(s'', a^3, k) = \mathit{true}$ if $a^3(q_1) \neq \mathit{true}$ or $a^3(q_2) \neq \mathit{false}$, and $\rho^3(s'', a^3, k) = \mathit{false}$ otherwise. We thus obtain $\phi^o = \mathsf{EX}(q_1 \wedge q_2) \wedge \mathsf{AX}(\neg q_1 \vee q_2)$. (Indeed, ϕ is not self-minimizing: for instance, for M with sole state s and sole transition (s, s) such that $L(s, q_1) = \perp$ and $L(s, q_2) = \mathit{false}$, we have $[(M, s) \models \phi] = \perp$ while $[(M, s) \models \phi]_t = \mathit{false}$; note $(M, s) \not\models^o \phi^o$ as expected.)*

For PL, our tableaux-based procedure for computing ϕ^o is simpler than that of [2] and [28], although it may generate larger formulas. In the worst case, the size of ϕ^o and A^3_ϕ can be exponentially larger than the size of ϕ. This is unavoidable as computing ϕ^o for ϕ is at least as hard as the GMC problem for ϕ, which itself is as hard as satisfiability for ϕ [4]: computing ϕ^o is NP-hard in $|\phi|$ for PL and is PSPACE-hard in $|\phi|$ for PML.

Theorem 1 also reveals the model complexity of the GMC problem for PML.

Corollary 1 *The generalized model checking problem for PML is in ALOGTIME in the size of the model.*

Proof: Theorem 1 provides a reduction from the GMC problem for a PML formula ϕ to the model checking problem for a PML formula ϕ^o, independently of any model M. Thus, since model checking for PML is in ALOGTIME in the size of M (e.g., [5]), so is GMC. ∎

5 Semantic Minimization for Fixed Points

Unlike for PML, we now show that CTL, LTL and CTL* are not closed under semantic minimizations.

As noted in [10] through a correspondence between GMC and module checking [23], $\mathrm{GMC}(M, s, \phi)$ is PTIME-hard in the size of the model M whenever ϕ ranges over CTL formulas. A more direct proof can be obtained by reducing the *monotone circuit value* problem, known to be PTIME-complete, for a circuit C to the GMC problem for a partial Kripke structure M_C defined from C and for a CTL formula of the form $\mathsf{A}[(\mathsf{EX}q_1)\mathsf{U}(q_1 \rightarrow q_2)]$. This reduction in turn implies the following.

Theorem 2 *The existence of an optimistic semantic minimization in CTL or CTL* for $\mathsf{A}[(\mathsf{EX}q_1)\mathsf{U}(q_1 \rightarrow q_2)] \in \mathrm{CTL}$ implies NLOGSPACE = PTIME.*

In other words, not all CTL and CTL* formulas (since CTL* includes CTL) have semantic minimizations in CTL*, unless NLOGSPACE = PTIME. The same result holds for LTL since GMC for LTL can also be shown to be PTIME-hard in the size of the model using a reduction from the monotone circuit value problem [10].

In the case of μL, we can prove a result similar to the PML case. To obtain μL we extend the grammar of PML with clauses Z for recursion variables and $\mu Z.\phi$ for (least fixed-point) recursion. For a 3-valued semantics, valuations $\mathcal{V}$ map variables Z to pairs $(\mathcal{V}_p(Z), \mathcal{V}_o(Z))$ of subsets of states. For closed ϕ, the 3-valued semantics $[M, \phi]_{\mathcal{V}}$ of [4] computes a pair (P, N) of subsets of S such that $P = \{s \in S \mid (M, s) \models^p \phi\}$ and $N = \{s \in S \mid (M, s) \not\models^o \phi\}$.

Theorem 3 *Every formula of μL has a semantic minimization in μL.*

Proof: (Sketch) The proof is similar to that of Theorem 1. For $\phi \in \mu L$, build a nondeterministic *parity* tree automaton A_ϕ that accepts exactly the infinite trees satisfying ϕ [24, 26]. Then, using a construction similar to (6), one obtains a nondeterministic parity tree automaton A^3_ϕ that accepts a 3-valued tree T^3 iff there exists a 2-valued tree T such that $T^3 \preceq T$ and T is accepted by A_ϕ. This automaton A^3_ϕ can then be re-encoded as a μL formula ϕ^o [27]. ∎

A corollary of the previous theorem (and of Proposition 2) is that all CTL, LTL and CTL* formulas have semantic minimizations in μL (since μL semantically includes CTL, LTL and CTL*). Since computing a semantic minimization is at least as hard as GMC, which has itself the same complexity (EXPTIME-complete) as satisfiability in the case of μL [4], computing ϕ^o is EXPTIME-hard in $|\phi|$ for ϕ in μL.

Thanks to Theorem 3, we can now prove the following result, which strengthens Theorem 2.

Theorem 4 *The optimistic semantic minimization of the CTL formula $\phi = \mathsf{A}[(\mathsf{EX}q_1)\mathsf{U}(q_1 \rightarrow q_2)]$ is the μL formula $\phi^o = \mu Z_1.(q_1 \rightarrow q_2) \vee [\mu Z_2.\mathsf{AX}Z_1 \wedge \mathsf{EX}(q_1 \wedge (q_2 \vee Z_2))]$, which is not expressible in CTL*.*

Proof: (Sketch) Using the construction of the proof of Theorem 3, we obtain an automaton A^3_ϕ such that $\rho^3(s_0, a^3, k) = \mathit{true}$ if $a^3(q_1) \neq \mathit{true}$ or $a^3(q_2) \neq \mathit{false}$, and $\rho^3(s_0, a^3, k) = \{(s_1, \ldots, s_k) \mid s_i = s' \text{ and } s_j = s_0\ \forall j \neq i\}$ otherwise; $\rho^3(s', a^3, k) = \mathit{true}$ if $a^3(q_1) \neq \mathit{false}$ and $a^3(q_2) \neq \mathit{false}$, $\rho^3(s', a^3, k) = \mathit{false}$ if $a^3(q_1) = \mathit{false}$, and $\rho^3(s', a^3, k) = \{(s_1, \ldots, s_k) \mid s_i = s' \text{ and } s_j = s_0\ \forall j \neq i\}$ otherwise. A^3_ϕ can then be re-encoded as ϕ^o, which can be shown not to be expressible in CTL*. ∎

Going beyond μL, we note that a semantic minimization for all formulas of *first-order logic* over a binary relation R and unary relations q cannot exist due to a *decidability* gap: $(M, s) \models^o \phi^o$ is decidable whereas $\mathrm{GMC}(M, s, \phi)$ is not.

6 Semantic Self-Minimization

6.1 Checking for Self-Minimization

We now present a procedure for checking whether any $\phi \in \mu L$ is optimistically self-minimizing. The procedure consists of comparing the automaton A^3_ϕ defined in the previous section with an automaton $A^3_{\models^o \phi}$ that accepts exactly all the 3-valued trees T^3 for which $T^3 \models^o \phi$ holds. Such a $A^3_{\models^o \phi}$ with transition function ρ^o can be defined as an alternating parity tree automaton A^{alt}_ϕ with transition function ρ (with $O(|\phi|)$ states) that accepts exactly the computation trees satisfying ϕ [24], *except* that ρ^o satisfies

- $\rho^o(q, a^3, k) = (a^3(q) \neq \mathit{false})$ for all $q \in AP$, and
- $\rho^o(\neg q, a^3, k) = (a^3(q) \neq \mathit{true})$ for all $q \in AP$.

For any other state s not corresponding to q or $\neg q$, $\rho^o(s, a^3, k)$ may be defined as $\rho(s, a, k)$ in A^{alt}_ϕ [24] with any a such that $a^3 \preceq a$ since transitions for all such a in A^{alt}_ϕ are of the same form.

By construction and (3), $L(A^3_\phi) \subseteq L(A^3_{\models^o \phi})$, where $L(A)$ denotes the language (set of 3-valued trees) accepted by automaton A. Checking whether ϕ is optimistically self-minimizing then reduces to checking whether

$$L(A^3_{\models^o \phi}) \subseteq L(A^3_\phi)\,. \tag{7}$$

This *semantic* test is exact but expensive since the sizes of the automata involved can be exponential in $|\phi|$ as previously discussed. In the next subsections, we study partial but much cheaper tests based on *syntactic* characterizations of self-minimizing formulas. Note that any syntactic characterization is bound to be *incomplete* since a linear syntactic check on a formula cannot be as precise as the exponential semantic check in (7) of that formula. Such tests can be used for optimizing the abstract-check-refine process described in Section 1 by eliminating Step 2(b) for formulas that are detected to be self-minimizing, since Step 2(a) is guaranteed to have the same precision as Step 2(b) for those formulas.

6.2 Self-Minimization and Monotonicity

We start with a simple *syntactic* criterion that is *sufficient* to identify self-minimizing formulas, and is much cheaper to check than the exact procedure of Section 6.1. This criterion is closely related to reduction results of satisfiability to model checks for monotone/positive fragments of logics.

Proposition 4 *Let ϕ be a closed formula of μL such that no $q \in AP$ occurs in the negation normal form of ϕ in mixed polarity. Then ϕ is semantically self-minimizing.*

Since PL, PML, CTL, LTL, and CTL* embed into μL by preserving the polarity of atomic propositions, this result also applies to these temporal logics. The syntactic condition in Proposition 4 is sufficient but not necessary, as shown by the next example.

Example 5 *The formula $(\neg q_1 \vee q_2) \wedge (\neg q_2 \vee q_1)$, the "iff" connective $q_1 \leftrightarrow q_2$, is semantically self-minimizing but contains atoms with mixed polarity. Its 2-valued semantics is not formally monotone, so any formula of μL equivalent to it requires some atom of mixed polarity in its negation normal form.*

6.3 Temporal Patterns of Self-Minimization

We now offer grammars and patterns that certify formulas to be semantically self-minimizing and are beyond the scope of Proposition 4. A grammar should be "optimal" in that constraints of clauses cannot be relaxed without making them unsound. We build up such grammars and discuss how they adjust to the case of semantic minimization.

In Figure 1 we write ps and os for syntactic categories that generate only formulas that are pessimistically (respectively, optimistically) self-minimizing. The clause $\mathcal{M}$ (for "**M**onotone") is syntactically checkable and sound as it stands for any formula of μL meeting the assumptions of Proposition 4. The clauses in Figure 1 for propositional connectives, EX and AX are sound due to Proposition 1.

As for the clause $\mathcal{R}$ in Figure 1, for each pointed model (M, s) there is a formula $\phi_{(M,s)} \in \mu L$ with

$$\forall (N,t)\colon (N,t) \models^p \phi_{(M,s)} \text{ iff } (M,s) \preceq (N,t) \qquad (8)$$

by [17] and [13]. In [17] it is shown, for modal transition systems, that $(M,s) \preceq (N,t)$ iff $\mathcal{C}[N,t] \subseteq \mathcal{C}[M,s]$ for all pointed models (the if-part being non-trivial); and that this is equivalent to, in our terminology, all $\phi_{(M,s)}$ being pessimistically self-minimizing. By [13], these results also apply to our models so $\mathcal{R}$, ranging over all such $\phi_{(M,s)}$, is a sound ground clause for ps. As for the soundness of $\mathcal{R}$ for os, $\mathrm{GMC}(N, t, \phi_{(M,s)})$ holds iff $\mathcal{C}[N,t] \cap \mathcal{C}[M,s] \neq \{\}$. But if $(N,t) \models^o \phi_{(M,s)}$ holds, the parity game of that model check determines a common refinement witness showing $\mathcal{C}[N,t] \cap \mathcal{C}[M,s] \neq \{\}$ by Theorem 3 of [16].

Next we convey the main ideas and techniques of our soundness proofs for temporal operators other than EX and AX for mode o, the ideas and proofs for mode p being dual. These proofs exploit three facts: we may assume (M, s) to be an infinite, 3-valued, labelled tree; (K, k) to be a labelled tree if $(K, k) \in \mathcal{C}[M, S]$ satisfies some $\phi \in \mu L$; and the equations (9) and (10) below to hold for the judgments $\models^o$ and $\models^p$, not just for 2-valued satisfaction. We use CTL* connectives as syntactic sugar in μL.

For unary temporal operators f we need to show that $\phi = \phi^o$ implies $f(\phi)^o = f(\phi)$. For f being AF or EF, we do this by completing the infinite, 3-valued, labelled tree (M, s) in any way up to a witness state for ϕ, whose subtree is replaced with a completion satisfying ϕ (which exists as $\phi = \phi^o$). For AF this technique applies to all paths, for EF to some path and all other paths are completed arbitrarily.

The clause for $\mathsf{EG}\eta_\exists$ requires a different approach. We take a witness path for $(M, s) \models^o \mathsf{EG}\eta_\exists$ and infer that all states s_i on that path π have completions satisfying $\eta_\exists$, as $\eta_\exists = \eta_\exists^o$. We then identify those states s_i with the initial states of these completions and "glue" these completions as new paths into M. The syntactic form of $\eta_\exists$ ensures that the glued completions are still witnesses for $\eta_\exists$ in the resulting model. Moreover, the larger model is a completion of (M, s) since we keep the "backbone" of M and add only completion paths. This construction will not succeed for formulas of the form $\mathsf{AG}\eta_\exists$ or $\mathsf{AG}\eta_\forall$ as all paths of glued completions would still be obliged to satisfy $\mathsf{G}\eta$, not just η. The clause for $\mathsf{E}[\eta_\exists \mathsf{U} \psi]$ blends the techniques used for F and G above: the G-technique for the invariant $\eta_\exists$ up until ψ is true, where we complete according to F.

$$\begin{array}{lcl}
\mathrm{ps} & ::= & \mathcal{M} \mid \mathcal{R} \mid \neg\mathrm{os} \mid \mathrm{ps} \wedge \mathrm{ps} \mid \mathrm{ps}_{\forall\#} \vee \mathrm{ps}_{\forall\#} \\
 & & \mathsf{EX}\mathrm{ps} \mid \mathsf{AX}\mathrm{ps} \mid \mathsf{EG}\mathrm{ps} \mid \mathsf{AG}\mathrm{ps} \\
 & & \mathsf{AF}\mathrm{ps}_\forall \mid \mathsf{A}[\mathrm{ps}_{\forall\#} \mathsf{U} \mathrm{ps}_{\forall\#}] \\
\mathrm{os} & ::= & \mathcal{M} \mid \mathcal{R} \mid \neg\mathrm{ps} \mid \mathrm{os} \vee \mathrm{os} \mid \mathrm{os}_{\exists\#} \wedge \mathrm{os}_{\exists\#} \\
 & & \mathsf{EX}\mathrm{os} \mid \mathsf{AX}\mathrm{os} \mid \mathsf{EF}\mathrm{os} \mid \mathsf{AF}\mathrm{os} \\
 & & \mathsf{EG}\mathrm{os}_\exists \mid \mathsf{E}[\mathrm{os}_\exists \mathsf{U} \mathrm{os}] \mid \mathit{ref}(OS)
\end{array}$$

Figure 1. ps **(**os**) generates pessimistically (optimistically) self-minimizing formulas (resp.); $\mathcal{M}$ ranges over monotone formulas of μL, $\mathcal{R}$ over formulas in (8); $\#$ and $\forall$ ($\exists$) are as in Definition 3(5); OS ranges over finite subsets of** os**; and $\mathit{ref}(\cdot)$ is as in Definition 4.**

Theorem 5 1. *Let $\phi, \psi, \eta_\exists \in \mu L$ be optimistically self-minimizing. Then $\mathsf{EF}\phi$, $\mathsf{AF}\phi$, $\mathsf{EG}\eta_\exists$, and $\mathsf{E}[\eta_\exists \mathsf{U} \psi]$ are optimistically self-minimizing.*

2. *Let $\phi, \psi, \eta_\forall \in \mu L$ be pessimistically self-minimizing. Then $\mathsf{EG}\phi$, $\mathsf{AG}\phi$, and $\mathsf{AF}\eta_\forall$ are pessimistically self-minimizing. If in addition, $\eta_\forall \# \psi$ and $\psi = \psi_\forall$, then $\mathsf{A}[\psi_\forall \mathsf{U} \eta_\forall]$ is pessimistically self-minimizing as well.*

Even for PL, genuine completeness of grammars for ps and os cannot be hoped for. If $q_1 \leftrightarrow q_2$ from Example 5 is presented in conjunctive (disjunctive) normal form, it is generated by ps (os) and not by os (respectively, ps). Also, the relaxation of any constraints in non-ground clauses makes the grammars in Figure 1 unsound. One cannot remove $\#$ from the clause for $\vee$ in ps since $p \vee \neg p$ would otherwise be derivable but is not pessimistically self-minimizing. A dual comment applies to the $\wedge$ clause of os. From the proof constructions it is also evident that we cannot omit the annotations $\forall$ and $\exists$ in the clauses of ps and os that mention them. From Theorems 2 and 4, we cannot expect an os clause for AU (note that both arguments of $\mathsf{A}[(\mathsf{EX}q_1)\mathsf{U}(q_1 \to q_2)]$ are in os).

Remark 1 *The absence of an* os *clause for* AU*, and an* EU *clause for* ps*, has to do with the semantic equivalences*

$$\neg\mathsf{E}[\phi \mathsf{U} \psi] = \mathsf{A}[\mathsf{G}\neg\psi \vee \neg\psi \mathsf{U} \neg\phi \wedge \neg\psi] \qquad (9)$$

$$\neg\mathsf{A}[\phi \mathsf{U} \psi] = \mathsf{EG}\neg\psi \vee \mathsf{E}[\neg\psi \mathsf{U} \neg\phi \wedge \neg\psi] \qquad (10)$$

which also hold for $\models^o$ and $\models^p$. We sketch proof attempts for self-minimization and reasons for their failures.

- *For* EU *and* ps*, let all* $(K,k) \in \mathcal{C}[M,s]$ *satisfy* $\mathsf{E}[\phi \mathsf{U} \psi]$*. "Proof" by contradiction:* $(M,s) \not\models^p \mathsf{E}[\phi \mathsf{U} \psi]$ *means* $(M,s) \models^o \neg \mathsf{E}[\phi \mathsf{U} \psi]$ *so* $(M,s) \models^o \mathsf{A}[\mathsf{G}\neg\psi \vee \neg\psi \mathsf{U} \neg\phi \wedge \neg\psi]$*. But* $\mathsf{A}[\alpha \vee \beta] \to (\mathsf{A}\alpha \vee A\beta)$ *is not valid.*
- *For* AU *and* os*, let* $(M,s) \models^o \mathsf{A}[\phi \mathsf{U} \psi]$*. Then* $(M,s) \models^o \neg\neg \mathsf{A}[\phi \mathsf{U} \psi]$ *so* $(M,s) \not\models^p \mathsf{EG}\neg\psi \vee \mathsf{E}[\neg\psi \mathsf{U} \neg\phi \wedge \neg\psi]$*. Even if* $\phi \# \psi$*, we only get some* $(K_1,k_1) \in \mathcal{C}[M,s]$ *satisfying* $\neg\mathsf{EG}\neg\psi$ *and some* $(K_2,k_2) \in \mathcal{C}[M,s]$ *satisfying* $\neg\mathsf{E}[\neg\psi \mathsf{U} \neg\phi \wedge \neg\psi]$*. But a proof would need the same completion in both cases.*

The asymmetry in os and ps may only be apparent: $\mathit{ref}(OS)$ has a logical dual, mediated through the clauses for negation, which could be made explicit; and the $\#$ in the AU clause for ps, forced upon us by an inductive proof using (10), may not be necessary. Although one cannot relax constraints for inductive clauses of Figure 1 any more, one can define new inductive clauses that constrain the *interaction contexts* of such clauses, ad infinitum. We limit ourselves to specifying one such example, a generalization of a construct implicit in clause $\mathcal{R}$.

Definition 4 *[20] Let* $\mathcal{O}$ *be a finite set of formulas in* μL*. Then* $\mathit{ref}(\mathcal{O})$ *is defined as* $(\bigwedge_{O \in \mathcal{O}} \mathsf{EX} O) \wedge (\mathsf{AX} \bigvee \mathcal{O})$*.*

The 2-valued meaning of $\mathit{ref}(\mathcal{O})$ is "all formulas in $\mathcal{O}$ are true at some successor state and all successor states satisfy some formula in $\mathcal{O}$." This pattern corresponds to checking whether a game with "continuation" $\mathcal{O}$ cannot be lost with the next exchange of moves. Combining $\mathit{ref}(\mathcal{O})$ with greatest fixed points expresses all instances of $\mathcal{R}$ by [17] and [13]. We prove soundness of clause $\mathit{ref}(OS)$ for os.

Theorem 6 *Let* $\mathcal{O}$ *be a finite set of optimistically self-minimizing formulas in* μL*. Then* $\mathit{ref}(\mathcal{O}) = \mathit{ref}(\mathcal{O})^o$*.*

A clause $\mathit{ref}(\mathcal{P})$ for a finite set $\mathcal{P}$ of pessimistically self-minimizing formulas is unsound in general, given the disjunction under AX in $\mathit{ref}(\mathcal{P})$. The soundness of this disjunction for ps for the formulas $\phi_{(M,s)}$ that logically characterize refinement has been shown in [17].

Mode p is used for *proving* properties. Fortunately, the pessimistic self-minimization of many popular specification patterns can be certified by our ps and os. We illustrate this with results for "weak until," stimulus-response chains, and the "globally true before r" pattern. The temporal operator "weak until," $\mathsf{A}[\phi \mathsf{W} \psi]$, is often required in model checking instead of the ordinary "until" $\mathsf{A}[\phi \mathsf{U} \psi]$. The semantics of "weak until" is $\mathsf{A}[(\phi \mathsf{U} \psi) \vee \mathsf{G}\phi]$. So ψ can be false forever as long as ϕ is true forever, e.g. as in "The elevator door remains open until a service button is being pressed."

Corollary 2 *1. Let* $\phi_\forall$ *and* $\psi_\forall$ *be pessimistically self-minimizing and* $\phi_\forall \# \psi_\forall$*. Then* $\mathsf{A}[\phi_\forall \mathsf{W} \psi_\forall]$ *is pessimistically self-minimizing.*

2. Let $p_\exists, q_\exists, s_\exists, t_\exists \in \mu L$ *with* $p_\exists \# q_\exists$*,* $p_\exists \# s_\exists$*,* $p_\exists \# t_\exists$*,* $q_\exists \# s_\exists$*,* $q_\exists \# t_\exists$*, and* $s_\exists \# t_\exists$*. E.g.* $\{p_\exists, q_\exists, s_\exists, t_\exists\} \subseteq AP$ *and* $|\{p_\exists, q_\exists, s_\exists, t_\exists\}| = 4$*. Then the* **Globally-1-Stimulus-2-Response Chain** *pattern "*$s_\exists, t_\exists$ *respond to* $p_\exists$ *after* $q_\exists$*" is pessimistically self-minimizing.*

3. Let $\phi_\exists$ *be optimistically and* $\psi_\forall$ *pessimistically self-minimizing. Then "globally,* $\psi_\forall$ *becomes true before* $\phi_\exists$*" is pessimistically self-minimizing.*

Some more complex patterns may require an extension of ps and os with constrained interactions of existing clauses. Indeed, the proof for "globally, $\psi_\forall$ becomes true before $\phi_\exists$" uses duality and a semantic equivalence, an absorption law not expressed in ps and os.

Finally, ps and os are sound if interpreted for semantic minimization, the proofs require more or less cosmetic changes only over those for self-minimization. One then has to interpret the clauses of Figure 1 as follows: the clause EFos states that $\mathsf{EF}\phi^o = (\mathsf{EF}\phi)^o$ for all $\phi \in \mu L$, etc.

6.4 Distributive Formulas and Self-Minimization

Janin & Walukiewicz [20] show that all modal mu-calculus formulas have normal forms ("distributive formulas" [20]) with linear-time satisfiability checks. We customize their definitions to our setting and prove that many distributive formulas are optimistically self-minimizing.

Definition 5 *Distributive formulas are those formulas of* μL *(extended with the constant ff) generated by*

$$D ::= \mathit{ff} \mid q \mid Z \mid D \vee D \mid D_1 \wedge \cdots \wedge D_n \mid \sigma Z.D$$

where $q \in AP$*;* $\sigma \in \{\mu, \nu\}$ *and, for* $\sigma Z.D$*,* Z *occurs positively and not in any context* $\cdot \wedge \gamma$ *or* $\gamma \wedge \cdot$ *in* D*;* $n \geq 1$ *and each* D_i *in* $D_1 \wedge \ldots D_n$ *is either a literal (*q *or* $\neg q$*) or of the form* $\mathit{ref}(\mathcal{D})$ *for a finite set* $\mathcal{D}$ *of distributive formulas, and at most one of the* D_i *is of the form* $\mathit{ref}(\mathcal{D})$*.*

Example 6 *Let* ϕ *be* $q \wedge \mathsf{EX}\neg q$*. Since* EX *is expressible via* $\mathit{ref}(\cdot)$ *[20],* ϕ *is a distributive formula. For* $AP = \{q\}$ *and* $M = (\{s\}, \{(s,s)\}, L(s,q) = \bot)$ *we have* $(M,s) \models^o \phi$*. But there is also some* (K,k) *of* (M,s) *satisfying* ϕ*, with states* k *and* k'*, transitions* (k,k') *and* (k',k')*, and labelings* $L_K(k,q) = \mathit{true}$ *and* $L_K(k',q) = \mathit{false}$*. Below we prove that such formulas are optimistically self-minimizing.*

By [20] and [13], every formula of μL is semantically equivalent to some distributive formula D. Without loss of generality, we may assume that all $D = D_1 \wedge \cdots \wedge D_n$ are satisfiable (otherwise replace D with *ff*) and that their monomials of literals mention literals at most once.

We recall the syntactic unfoldings of fixed-points, where $\phi[Z/\eta]$ denotes the formula resulting from substituting all free occurrences of Z in ϕ by η:

$$\mu_0.Z.\phi = \mathit{ff} \qquad \mu_{n+1}Z.\phi = \phi[Z/\mu_n Z.\phi] \qquad (n \geq 0)$$
$$\nu_0.Z.\phi = \mathit{tt} \qquad \nu_{n+1}Z.\phi = \phi[Z/\nu_n Z.\phi] \qquad (n \geq 0)\,.$$

Theorem 7 *All closed distributive formulas without greatest fixed points are optimistically self-minimizing.*

For greatest fixed points, let $(M, s) \models^o \nu Z.D$. Then $(M, s) \models^o \nu_n Z.D$ for all $n \geq 1$. Induction would imply $\mathrm{GMC}(M, s, \nu_n Z.D)$ for all $n \geq 1$. But the latter is only sometimes sufficient for concluding $\mathrm{GMC}(M, s, \nu Z.D)$. For example, we can express $\mathsf{EG}\eta_\exists$ as $\nu Z.\eta_\exists \wedge \mathsf{EX}(Z)$ and our proof for clause $\mathsf{EG}\mathrm{os}_\exists$ can be interpreted as constructing a witness to $\mathrm{GMC}(M, s, \nu Z.\eta_\exists \wedge \mathsf{EX}(Z))$ from "incremental" witnesses of all $\mathrm{GMC}(M, s, \nu_n Z.\eta_\exists \wedge \mathsf{EX}(Z))$.

Theorem 7 can derive only the soundness of some clauses for ps and os. For example, $\mathsf{EF}\phi$ can be written as $\mu Z.\phi \vee \mathsf{EX}(Z)$ and expressed as a distributive formula meeting the assumptions of Theorem 7 as EX and AX are derived from $\mathit{ref}(\cdot)$ [20]. But $\mathsf{E}[\phi \mathsf{U} \psi]$ has fixed-point characterization $\mu Z.\psi \vee (\phi \wedge \mathsf{EX}(Z))$ which would generally need a non-trivial conversion into a distributive format.

The proofs of Theorem 7 also reveal that one could extend the grammar for os with certain least fixed-point clauses that have as side condition that all their finite unfoldings are generated by os as well.

7 Applications of Self-Minimization

We briefly look at specification patterns used in practice and discuss whether they are semantically self-minimizing.

Some instances of $\mathcal{M}$ are found in the classification of frequently-used temporal-logic patterns of [7]: **(Absence)** $\mathsf{AG}(q \rightarrow \mathsf{AG}(\neg p))$, **(Universality)** $\mathsf{AG}(q \rightarrow \mathsf{AG}p)$, **(Existence)** $\mathsf{EF}p$, **(Response)** $\mathsf{AG}(p \rightarrow \mathsf{AF}s)$, **(Response Chain)** $\mathsf{AG}(p \rightarrow \mathsf{AF}(s \wedge \mathsf{AX}(\mathsf{AF}t)))$, etc., where $p, q, s, t \in \mathit{AP}$.

The model checker LTSA [25] uses labeled transition systems and LTL over a finite set of actions Act containing error. Two core patterns are a safety property $\mathsf{G}\neg \mathit{error}$ and a liveness property $\mathsf{GF}\bigvee \mathit{Act}$. The embeddings of $\mathsf{G}\neg\mathit{error}$ and $\mathsf{GF}\bigvee \mathit{Act}$ into μL are in $\mathcal{M}$.

Self-minimizing temporal-logic formulas also occur in program and data-flow analysis. We mention two data-flow analyzes captured as model checking problems [29]:

$$\begin{aligned} \mathit{isLive}_x &= \mathsf{EF}_{\{a \mid a \neq \mathit{mod}_x\}} \langle \mathit{use}_x \rangle \mathit{tt} \\ \mathit{isDead}_x &= \mathsf{AG}_{\{a \mid a \neq \mathit{use}_x\}} (\neg \mathit{end} \wedge \langle \mathit{mod}_x \rangle \mathit{ff})\,. \end{aligned}$$

Formula isLive_x says "variable x is live at the current program point," where $\mathit{use}_x \in \mathit{AP}$ ($\mathit{mod}_x \in \mathit{AP}$) denotes that variable x is used (respectively, modified) at a program point. Formula isDead_x says "x is dead at a program point," where end is true at the end point of a program only. The patterns isLive_x and isDead_x can be expressed in μL directly and shown to be instances of $\mathcal{M}$.

Symbolic trajectory formulas [30], used in hardware verification, can be defined by the grammar $\phi ::= q \mid \neg q \mid \phi \wedge \phi \mid \eta \rightarrow \phi \mid \mathsf{AX}\phi$ where $q \in \mathit{AP}$ and η ranges over boolean formulas from AP. If all $\eta \rightarrow \phi'$ of ϕ are such that $\eta \# \phi'$, then ϕ is pessimistically self-minimizing, including for the models used in [30] which are symbolic trace presentations of certain partial Kripke structures.

8 Conclusions

We studied in this paper for which temporal-logic formulas model checking has the same precision as generalized model checking, independently of any model. We identified those formulas as semantically self-minimizing, characterized them using automata on 3-valued trees, and provided syntactic conditions for efficiently recognizing many of these, including many instances of frequently-used specification patterns. We also studied how to reduce the generalized model checking problem (including satisfiability as a special case) for a formula ϕ to regular model checking of a formula ϕ' obtained solely from ϕ, thus independently of the model. We proved the existence of such ϕ' for every ϕ in propositional modal logic (rendering the complexity of generalized model checking in the size of the model for this logic) and in the modal mu-calculus, extending a previously-known similar result for propositional logic. We also showed that, in contrast, the logics LTL, CTL, and CTL* are not closed under semantic minimizations.

Acknowledgments We thank Radha Jagadeesan for stimulating discussions, Michael Benedikt, Satish Chandra, Tom Melham for helpful comments, and the anonymous reviewers for their comments to improve the presentation of this paper. This work was funded in part by the NSF grant CCR-0341658.

References

[1] T. Ball and S. Rajamani. The SLAM Toolkit. In *Proceedings of the 13th Conference on Computer Aided Verification*, volume 2102 of *Lecture Notes in Computer Science*, pages 260–264, Paris, July 2001. Springer-Verlag.

[2] S. Blamey. *Partial-Valued Logic*. PhD thesis, University of Oxford, Oxford, England, 1980.

[3] G. Bruns and P. Godefroid. Model Checking Partial State Spaces with 3-Valued Temporal Logics. In *Proceedings of the 11th Conference on Computer Aided Verification*, volume 1633 of *Lecture Notes in Computer Science*, pages 274–287. Springer Verlag, July 1999.

[4] G. Bruns and P. Godefroid. Generalized Model Checking: Reasoning about Partial State Spaces. In *Proceedings of the*

11th International Conference on Concurrency Theory, volume 1877 of *Lecture Notes in Computer Science*, pages 168–182. Springer Verlag, August 2000.

[5] S. R. Buss. The Boolean Formula Problem Is in ALOGTIME. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 123–131, New York City, New York, 25-28 May 1987. ACM Press.

[6] S. Demri and P. Schnoebelen. The Complexity of Propositional Linear Temporal Logics in Simple Cases. *Information and Computation*, 174(1):84–103, 2002.

[7] M. B. Dwyer, G. S. Avrunin, and J. C. Corbett. Patterns in Property Specifications for Finite-state Verification. In *Proceedings ICSE 1999*. IEEE Computer Society Press, May 1999.

[8] E. A. Emerson. Temporal and Modal Logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*. Elsevier/MIT Press, Amsterdam/Cambridge, 1990.

[9] E. A. Emerson, M. Evangelist, and J. Srinivasan. On the limits of efficient temporal decidability. In *Proceedings of the 5th Annual IEEE Symposium on Logic in Computer Science*, pages 464–475, Philadelphia, Pennsylvania, 4-7 June 1990.

[10] P. Godefroid. Reasoning about Abstract Open Systems with Generalized Module Checking. In *Proceedings of the 3rd Conference on Embedded Software*, volume 2855 of *Lecture Notes in Computer Science*, pages 223–240, Philadelphia, October 2003. Springer Verlag.

[11] P. Godefroid, M. Huth, and R. Jagadeesan. Abstraction-based Model Checking using Modal Transition Systems. In *Proceedings of the 12th International Conference on Concurrency Theory*, volume 2154 of *Lecture Notes in Computer Science*, pages 426–440, Aalborg, August 2001. Springer-Verlag.

[12] P. Godefroid and R. Jagadeesan. Automatic Abstraction Using Generalized Model Checking. In E. Brinksma and K. G. Larsen, editors, *Proceedings of the 14th International Conference on Computer Aided Verification*, volume 2404 of *Lecture Notes in Computer Science*, pages 137–150, Copenhagen, Denmark, July 2002. Springer Verlag.

[13] P. Godefroid and R. Jagadeesan. On The Expressiveness of 3-Valued Models. In L. D. Zuck, P. C. Attie, A. Cortesi, and S. Mukhopadhyay, editors, *Proceedings of the 4th Conference on Verification, Model Checking and Abstract Interpretation*, volume 2575 of *LNCS*, pages 206–222, New York, January 2003. Springer Verlag.

[14] T. Henzinger, R. Jhala, R. Majumdar, and G. Sutre. Lazy Abstraction. In *Proceedings of the 29th ACM Symposium on Principles of Programming Languages*, pages 58–70, Portland, January 2002.

[15] T. A. Henzinger, O. Kupferman, and R. Majumdar. On the Universal and Existential Fragments of the μ-Calculus. In *Proceedings of the 9th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 2619 of *Lecture Notes in Computer Science*, pages 49–64. Springer Verlag, 2003.

[16] A. Hussain and M. Huth. On model checking multiple hybrid views. In *Preliminary Proceedings of the First International Symposium on Leveraging Applications of Formal Method*, pages 235–242, Paphos, Cyprus, 30 October - 2 November 2004. Technical Report TR-2004-6 Department of Computer Science, University of Cyprus.

[17] M. Huth. Refinement is complete for implementations. Accepted for publication in *Formal Aspects of Computing*, December 2004.

[18] M. Huth, R. Jagadeesan, and D. Schmidt. A domain equation for refinement of partial systems. *Mathematical Structures in Computer Science*, 14(4):469–505, 5 August 2004.

[19] M. Huth, R. Jagadeesan, and D. A. Schmidt. Modal transition systems: a foundation for three-valued program analysis. In D. Sands, editor, *Proceedings of the 10th European Symposium on Programming*, pages 155–169. Springer Verlag, April 2001.

[20] D. Janin and I. Walukiewicz. Automata for the modal mu-calculus and related results. In *Proceedings of the 20th International Symposium on Mathematical Foundations of Computer Science*, volume 969 of *Lecture Notes in Computer Science*, pages 552–562. Springer-Verlag, 1995.

[21] S. C. Kleene. *Introduction to Metamathematics*. Van Nostrand, 1952.

[22] D. Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[23] O. Kupferman and M. Vardi. Module Checking. In *Proceedings 8th Conference on Computer Aided Verification*, volume 1102 of *Lecture Notes in Computer Science*, pages 75–86, New Brunswick, August 1996. Springer-Verlag.

[24] O. Kupferman, M. Y. Vardi, and P. Wolper. An Automata-Theoretic Approach to Branching-Time Model Checking. *Journal of the ACM*, 47(2):312–360, March 2000.

[25] J. Magee. Behavioral analysis of software architectures using LTSA. In *Proceedings of the 21st IEEE International Conference on Software Engineering*, pages 634–637, Los Angeles, California, 16-22 May 1999. ACM Press.

[26] D. E. Muller and P. E. Schupp. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems by Rabin, McNaughton and Safra. *Theoretical Computer Science*, 141(1–2):69–108, 1995.

[27] D. Niwinski. Fixed Points vs. Infinite Generation. In *Proceedings of the 3rd Annual IEEE Symposium on Logic in Computer Science*, pages 402–409. IEEE Computer Society Press, 1988.

[28] T. Reps, A. Loginov, and M. Sagiv. Semantic Minimization of 3-Valued Propositional Formulae. In *Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science*, pages 40–51, Copenhagen, Denmark, 22-25 July 2002. IEEE Computer Society Press.

[29] D. A. Schmidt and B. Steffen. Data-flow analysis as model checking of abstract interpretations. In G. Levi, editor, *Proceedings of the 5th Static Analysis Symposium*, volume 1503 of *Lecture Notes in Computer Science*, pages 351–380, Pisa, Italy, 14-16 September 1998. Springer.

[30] C.-J. H. Seger and R. E. Brant. Formal Verification by Symbolic Evaluation of Partially-Ordered Trajectories. *Formal Methods in System Design*, 6(2):147–189, March 1995.

[31] B. van Fraassen. Singular terms, truth-value gaps, and free logic. *J. Phil.*, 63(17):481–495, September 1966.