Automata games for multiple-model checking

Altaf Hussain and Michael Huth [1,2]

Department of Computing, South Kensington campus, Imperial College London, London, SW7 2AZ, United Kingdom

Abstract

3-valued models have been advocated as a means of system abstraction such that verifications and refutations of temporal-logic properties transfer from abstract models to the systems they represent. Some application domains, however, require multiple models of a concrete or virtual system. We build the mathematical foundations for 3-valued property verification and refutation applied to sets of common concretizations of finitely many models. We show that validity checking for the modal mu-calculus has the same cost (EXPTIME-complete) on such sets as on the set of all 2-valued models, provide an efficient algorithm for checking whether common concretizations exist for a fixed number of models, and propose using parity games on variants of tree automata to efficiently approximate validity checks of multiple models. Structural properties of a universal topological model confirm that such approximations are reasonably precise only for tree-automata-like models.

Key words: model checking, consistency, parity games, focussed transition systems, tree automata.

1 Introduction

Model checking [34,6] creates and decides judgments $M \models \phi$, where M is a model of a computational system, ϕ is a property, and $\models$ a satisfaction relation specifying which models enjoy what properties. In this context, abstraction is widely perceived as a key technique in combatting the notorious state explosion problem, that the size of models is typically exponential in the number of system observables or processes. Recent years have seen an increased use of 3-valued system abstractions in model checking and program analysis (e.g. [8,9,35,3,14,15]). Such abstract models are 3-valued as static and dynamic information is specified in two modes: "*may* be true" and "*must* be true."

The main benefit of this approach is that both property verification ($M \models \phi$ holds) *and* refutation ($M \models \phi$ *doesn't* hold) on abstract models transfer

[1] Email: ah701@doc.imperial.ac.uk
[2] Email: M.Huth@doc.imperial.ac.uk

This is a preliminary version. The final version will be published in Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

soundly to the concrete systems they model, whereas this is only true for property *verifications* in the 2-valued case. The abstraction of a concrete system in predicate abstraction tools, such as SLAM [1] and BLAST [18], is traditionally a "safe simulation" and allows the verification of universal properties only. The 3-valued approach of abstraction is not limited in this way and properties that combine existential and universal quantifiers are more and more needed in exploiting the observed merging of testing, model checking, and simulation environments in formal methods.

Yet there are a range of situations in which reasoning about a single model is undesirable, unacceptable or impossible. We state some examples.

- In requirements engineering, stake holders formulate expectations and constraints for a system and each such viewpoint can be construed as a model.
- Federated databases provide the illusion of a single data repository but each local database may be interpreted as a single model of data.
- In software verification, a computer program may be abstracted by different tools or abstract domain, each of which produces a model of that program.
- Today's software products need a high degree of configurability and each of their customized deployments has its specific model.
- In UML modelling, one rarely has a single message sequence chart and the collection of all charts is the natural subject of analysis.

All of these examples share that one wants to reason about finitely many models $M_1, \ldots, M_n$ *collectively*, and that individual models M_i benefit from being 3-valued since states and events foreign to M_i can be incorporated as *may* information whereas local knowledge is represented as *must* information. For example, if a database M_i has no entry for a proposition p, it is safe to assume that p may be true, but is not known to be true, in M_i.

If $\mathcal{C}(M)$ is the set of 2-valued concretizations of a 3-valued model M, e.g. defined through refinement [29] or abstract interpretation [7], model checks on M need to reason soundly about the entire set $\mathcal{C}(M)$ as any $K \in \mathcal{C}(M)$ could be the actual system modelled by M. The collective reasoning about finitely many models therefore reasons about sets of the form

$$\bigcap_{i=1}^{k} \mathcal{C}(M_i)\,, \tag{1}$$

the principal object of study in this paper.

In 2-valued model checking $M \models \phi$ one reasons about the set of concretizations $\mathcal{C}(M)$ of M, a singleton in the Stone space of equivalence classes for bisimulation [20]. In 3-valued model checking, the set of concretizations $\mathcal{C}(M)$ turns out to be a compact set in that very Stone space [20]. Thus, the transition from 2-valued to 3-valued model checking may be interpreted topologically as the transition from singleton compact sets to more general compact sets generated from single 3-valued models and refinement.

Consequently, the sets in (1) are also compact in that quotient space as

finite intersections of compact sets. Our paper can be seen as extending 3-valued model checking to the compact sets in (1) by developing two familiar research issues from 3-valued model checking [3,4] in this setting.

- **Issue #1.** To understand the computational complexity of satisfiability and validity checking over sets in (1) for the modal mu-calculus.
- **Issue #2.** To seek efficient ways of approximating those decision problems.

In moving from single compact sets $\mathcal{C}(M)$ to finite intersections of such sets, we are also faced with a novel decision problem, that of *consistency*. Sets in (1) may be empty and so no common concretizations of all M_i may exist. We therefore identify a third research issue in this setting.

- **Issue #3.** To efficiently decide the non-emptiness of sets in (1) for fixed k.

Contributions of our paper. Our paper solves Issues #1 and #3 completely, reviews and assesses existing proposals for Issue #2, and proposes a novel solution for Issue #2.

Outline of paper. We use a state-based version of Larsen & Thomsen's modal transition systems [29] as 3-valued models and review the necessary background in Section 2. Section 3 states the three decision problems studied in this paper and proves tight bounds for two of them. In Section 4 we develop an efficient algorithm for deciding the non-emptiness of sets of the form (1) for fixed k. Section 5 discusses how Dams & Namjoshi's techniques [10] based on tree automata and parity games can yield more efficient approximations for the EXPTIME-complete validity checks in UP $\cap$ co-UP, Section 6 states some related work, and Section 7 concludes.

2 Basic notions and background

Throughout, we fix a unary modality $\Diamond$ and a finite set AP of propositions.

Definition 2.1 (i) A *Kripke model* K is a tuple (Σ, R, L) with state set Σ, transition relation $R \subseteq \Sigma \times \Sigma$, and labelling function $L : \Sigma \to \mathbb{P}(AP)$.

(ii) A *modal Kripke model* M is a tuple $(\Sigma, R^a, R^c, L^a, L^c)$, where (Σ, R^a, L^a) and (Σ, R^c, L^c) are Kripke models, $R^a \subseteq R^c$, and $L^a(s) \subseteq L^c(s)$ for all $s \in \Sigma$. We refer to modal Kripke models as "models" when appropriate.

(iii) Whenever convenient, we view a Kripke model (Σ, R, L) as a modal Kripke model (Σ, R, R, L, L) and vice versa.

(iv) We call (M, s) a *pointed* modal Kripke model M with initial state s.

The intuition behind modal Kripke models is that R^a and $R^c \setminus R^a$ specify *must* and *may* transitions of the model, respectively [29]; whilst the L^a and L^c labellings assert information that is *known* to be true, and *may* be true, respectively [22]. The complements of R^c and L^c specify impossibilities, e.g. $(s, s') \notin R^c$ expresses the impossibility of a transition from state s to s'.

$$\begin{aligned}
&\llbracket\, q \,\rrbracket^m_\rho \;=\; L^m(q) && \llbracket\, Z \,\rrbracket^m_\rho = \rho^m(Z)\\
&\llbracket\, \neg\phi \,\rrbracket^m_\rho = \Sigma \setminus \llbracket\, \phi \,\rrbracket^{\neg m}_\rho && \llbracket\, \phi_1 \wedge \phi_2 \,\rrbracket^m_\rho = \llbracket\, \phi_1 \,\rrbracket^m_\rho \cap \llbracket\, \phi_2 \,\rrbracket^m_\rho\\
&\llbracket\, \Diamond\phi \,\rrbracket^m_\rho = \mathrm{pre}^m(\llbracket\, \phi \,\rrbracket^m_\rho) && \llbracket\, \mu Z.\phi \,\rrbracket^m_\rho = \mathrm{lfp}\lambda A.\llbracket\, \phi \,\rrbracket^m_{\rho[Z\mapsto A]}\,.
\end{aligned}$$

Fig. 1. Semantics of μL over modal Kripke models for mode $m \in \{a, c\}$.

We use the modal mu-calculus [26] (μL) as property semantics. Many branching-time temporal logics, e.g. CTL [2], are expressible in μL given by

$$\phi ::= q \;\mid\; Z \;\mid\; \neg\phi \;\mid\; \phi \wedge \phi \;\mid\; \Diamond\phi \;\mid\; \mu Z.\phi \tag{2}$$

where $q \in AP$, and Z ranges over a countable set Var of recursion variables. We write $\Box\phi$ for $\neg\Diamond\neg\phi$. In the least fixed point formula $\mu Z.\phi$, μZ binds all occurrences of Z in ϕ with static scoping and we require that all free occurrences of Z in ϕ are under an even scope of negations. If $\phi[Z/\psi]$ denotes the formula obtained from ϕ by replacing all free occurrences of Z in ϕ with ψ, the greatest fixed point formula $\nu Z.\phi$ is derived as $\neg\mu Z.\neg\phi[Z/\neg Z]$. A formula ϕ is closed if it contains no free recursion variables.

The denotational semantics $\llbracket\, \cdot \,\rrbracket^m$ of μL over modal Kripke models maps formulas ϕ and environments ρ, functions $Z \mapsto (\rho^a(Z), \rho^c(Z))$ of type $Var \to \{(L, U) \mid L \subseteq U \subseteq \Sigma\}$, into sets of states for a mode of analysis $m \in \{a, c\}$ in Figure 1. The semantics given in that figure relies on the definitions $\neg a = c$, $\neg c = a$, and $\mathrm{pre}^m(A) = \{s \in \Sigma \mid \exists s' \in A\colon (s, s') \in R^m\}$ for $m \in \{a, c\}$.

Definition 2.2 We write $s \models^a_\rho \phi$ for $s \in \llbracket\, \phi \,\rrbracket^a_\rho$; similarly $s \models^c_\rho \phi$ means $s \in \llbracket\, \phi \,\rrbracket^c_\rho$. If ϕ is closed, we elide the then redundant environment ρ.

All least fixed points $\mathrm{lfp}\lambda A.\llbracket\, \phi \,\rrbracket^m_{\rho[Z\mapsto A]}$ are formed in the complete lattice $(\mathbb{P}(\Sigma), \subseteq)$. Note that for $m \in \{a, c\}$ we have $s \models^m \Box\phi$ iff for all $(s, s') \in R^{\neg\mathbf{m}}$, $s' \models^m \phi$. If K is a Kripke model (Σ, R, R, L, L), then $\llbracket\, \phi \,\rrbracket^a_\rho = \llbracket\, \phi \,\rrbracket^c_\rho$ holds in K for all ρ and ϕ of μL [22] so this defines the Kripke semantics $k \models_\rho \phi$ to be $k \in \llbracket\, \phi \,\rrbracket^a_\rho$ for all states k of K.

Example 2.3 In the modal Kripke model of Figure 4 we have $s_1 \models^a \Box p$, since all R^c transitions out of s_1 lead to states s' with $p \in L^a(s')$, and $s_1 \models^c \mu Z.\neg q \vee \Diamond Z$, since there is an R^c-path $s_1 R^c t_1 R^c u_1$ to a state u_1 with $u_1 \models^c \neg q$.

In specifying a modal Kripke model we implicitly describe a possibly infinite set of Kripke models $\mathcal{C}(M)$ through a refinement notion.

Definition 2.4 For $i = 1, 2$ let $(M_i, s_i) = ((\Sigma_i, R^a_i, R^c_i, L^a_i, L^c_i), s_i)$ be pointed modal Kripke models. Then (M_1, s_1) is *refined* by (M_2, s_2) iff there is a relation $Q \subseteq \Sigma_1 \times \Sigma_2$ such that $(s_1, s_2) \in Q$ and, for all $(s, t) \in Q$, we have

(i) for all $q \in AP$, $s \in L^a_1(q)$ implies $t \in L^a_2(q)$,

(ii) for all $q \in AP$, $t \in L^c_2(q)$ implies $s \in L^c_1(q)$,

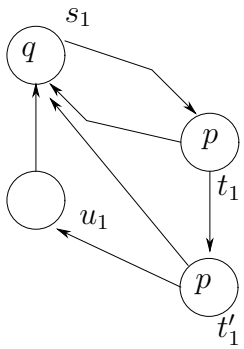

Fig. 2. A Kripke model K such that $(K, s_1) \in \mathcal{C}(M, s_1)$ for the modal Kripke model M in Figure 4.

(iii) if $(s, s') \in R_1^a$, then there is $(t, t') \in R_2^a$ with $(s', t') \in Q$,

(iv) if $(t, t') \in R_2^c$, then there is $(s, s') \in R_1^c$ with $(s', t') \in Q$.

Condition (iii) stipulates that refinement has to preserve *must* transitions; whilst condition (iv) expresses that refinement has to reflect *may* transitions; and labellings behave similarly. We write $(M_1, s) \prec (M_2, t)$ whenever there is such a Q with $(s, t) \in Q$ and denote by $\mathcal{C}(M, s)$ the *concretizations* of (M, s), defined as $\mathcal{C}(M, s) = \{(N, t) \mid (M, s) \prec (N, t),\ (N, t) \text{ is a Kripke model}\}$.

Example 2.5 Figure 2 shows a concretization (K, s_1) of the modal Kripke structure (M, s_1) in Figure 4 with refinement $\{(s_1, s_1), (t_1, t_1), (t_1, t'_1), (u_1, u_1)\}$.

As refinement is transitive, $(M_1, s) \prec (M_2, t)$ implies $\mathcal{C}(M_2, t) \subseteq \mathcal{C}(M_1, s)$. Refinement meshes well with, and is characterized by, our property semantics.

Theorem 2.6 ([22]) (i) *For all pointed modal Kripke models (M, s) and (N, t) we have that $(M, s) \prec (N, t)$ iff (for all closed, fixed-point free formulas ϕ of μL, $s \in [\![\phi]\!]^a$ implies $t \in [\![\phi]\!]^a$).*

(ii) *If $(M, s) \prec (N, t)$, then $s \in [\![\phi]\!]^a$ implies $t \in [\![\phi]\!]^a$, and $t \in [\![\psi]\!]^c$ implies $s \in [\![\psi]\!]^c$, for all closed ϕ, ψ of μL.*

This theorem secures soundness of $[\![\phi]\!]^m$ relative to the thorough semantics of Bruns & Godefroid in [4], where for any closed ϕ of μL the predicate for generalized model checking $GMC(M, s, \phi)$ is defined as "$k \models \phi$ for some $(K, k) \in \mathcal{C}(M, s)$."

This soundness is captured as a combined under-approximation and over-approximation in the following corollary, a reformulation of a result in [4].

Corollary 2.7 *For any closed $\phi \in \mu L$ and any state s of any model M:*

(i) *Under-approximation: If $s \in [\![\phi]\!]^a$, then $GMC(M, s, \neg\phi)$ is false.*

(ii) *Over-approximation: If $GMC(M, s, \phi)$ is true, then $s \in [\![\phi]\!]^c$.*

3 Multiple models and their decision problems

We can now define the decision problems studied in this paper. Subsequently, let $\mathcal{V} = \{(M_i, s_i) \mid 1 \leq i \leq k\}$ denote any finite set of pointed modal Kripke models (M_i, s_i), each having a finite set of states Σ_i. We identify the relevant decision problems.

Definition 3.1 Let

$$\mathcal{C}(\mathcal{V}) = \{(N, t) \mid \forall (M, s) \in \mathcal{V} \colon (N, t) \in \mathcal{C}(M, s)\} \tag{3}$$

be the set of *common concretizations* of $\mathcal{V}$. For closed ϕ of μL, we define parameterized boolean expressions $\mathbb{C}(\mathcal{V})$, $\mathbb{S}(\mathcal{V}, \phi)$, and $\mathbb{V}(\mathcal{V}, \phi)$:

(i) *Consistency:* $\mathbb{C}(\mathcal{V})$ holds iff all models of $\mathcal{V}$ have a common concretization, i.e. iff $\mathcal{C}(\mathcal{V}) \neq \{\}$.

(ii) *Satisfiability:* $\mathbb{S}(\mathcal{V}, \phi)$ is true iff there is a common concretization of $\mathcal{V}$ that satisfies ϕ, i.e. iff $\{(N, t) \in \mathcal{C}(\mathcal{V}) \mid t \models \phi\} \neq \{\}$.

(iii) *Validity:* $\mathbb{V}(\mathcal{V}, \phi)$ holds iff all common concretizations of $\mathcal{V}$ satisfy ϕ.

Since all pointed modal Kripke models $((\Sigma, R^a, R^c, L^a, L^c), s)$ have a concretization, e.g. $((\Sigma, R^a, L^a), s)$, $\mathbb{C}(\mathcal{V})$ holds iff all models of $\mathcal{V}$ have a common *refinement*. Note that $\mathbb{V}(\mathcal{V}, \phi)$ holds for all ϕ if $\mathcal{V}$ has no common refinement. Thus one should first establish $\mathbb{C}(\mathcal{V})$ before certifying $\mathbb{V}(\mathcal{V}, \phi)$.

We show that all three decision problems above are reducible to satisfiability checks of μL over Kripke models. Inspired by [27] we construct a closed formula $[M_i, s_i]$ of μL for each pointed and finite-state modal Kripke model (M_i, s_i) such that for all pointed modal Kripke models (N, t) we have

$$(N, t) \models^a [M_i, s_i] \qquad \text{iff} \qquad (M_i, s_i) \prec (N, t)\,. \tag{4}$$

The existence of such formulas and the reduction for $\mathbb{C}(\mathcal{V})$ have been shown for modal transition systems in Theorem 4.8(2) in [20]. The theorem below is merely a slight extension of that result.

Theorem 3.2 (i) *Each pointed and finite-state modal Kripke model (M_i, s_i) has a formula $[M_i, s_i]$ of μL satisfying (4) for all pointed modal Kripke models (N, t).*

(ii) *The decision problems $\mathbb{C}(\mathcal{V})$, $\mathbb{S}(\mathcal{V}, \phi)$, and $\mathbb{V}(\mathcal{V}, \phi)$ are in EXPTIME in the size of ϕ and reducible to satisfiability checks $\bigwedge_{i=1}^{k} [M_i, s_i]$, $\phi \wedge \bigwedge_{i=1}^{k} [M_i, s_i]$, and validity checks $\phi \vee \neg \bigwedge_{i=1}^{k} [M_i, s_i]$ of μL over Kripke models (respectively).*

Proof.

(i) For each state t_i in M_i we set, similar to (3) in [27]:

$$[M_i, t_i] = \Big(\bigwedge_{(t_i, t_i') \in R^a} \Diamond [M_i, t_i']\Big) \wedge \Box\Big(\bigvee_{(t_i, t_i') \in R^c} [M_i, t_i']\Big) \tag{5}$$
$$\wedge \bigwedge \{q \mid q \in L^a(t_i)\} \wedge \bigwedge \{\neg q \mid q \notin L^c(t_i),\ q \in AP\}$$

as a system of greatest fixed point equations. As M_i has only finitely many states, each $[M_i, t_i]$ is expressible in μL. The proof that $[M_i, s_i]$ satisfies (4) is basically the one given in [29].

(ii) We can reduce $\mathbb{C}(\mathcal{V})$ to a satisfiability check in μL by proving that $\mathcal{V}$ has a common concretization iff the closed formula

$$\sigma_{\mathcal{V}} = \bigwedge_{i=1}^{k} [M_i, s_i] \tag{6}$$

of μL is satisfiable over Kripke models. If $\sigma_{\mathcal{V}}$ is satisfiable, $k \models \sigma_{\mathcal{V}}$ for some pointed Kripke model (K, k). Since (K, k) can be cast into a pointed modal Kripke model, (4) and $k \models \sigma_{\mathcal{V}}$ render $(M_i, s_i) \prec (K, k)$ for all $i = 1, 2, \ldots, k$ and so $(K, k) \in \mathcal{C}(\mathcal{V})$. Conversely, if $\mathcal{V}$ has a common concretization (K, k) we have $(M_i, s_i) \prec (K, k)$ for all $i = 1, 2, \ldots, k$. Using (4) this implies $(K, k) \models \sigma_{\mathcal{V}}$ and so $\sigma_{\mathcal{V}}$ is satisfiable over Kripke models. The reductions for $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$ to satisfiability checks in μL are variations of the reduction for $\mathbb{C}(\mathcal{V})$. The check $\mathbb{S}(\mathcal{V}, \phi)$ holds iff $\phi \wedge \sigma_{\mathcal{V}}$ is satisfiable over Kripke models. The check $\mathbb{V}(\mathcal{V}, \phi)$ holds iff $\neg\phi \wedge \sigma_{\mathcal{V}}$ is unsatisfiable over Kripke models. But satisfiability checking of μL is in EXPTIME [12].

∎

The semantics of Figure 1, an approximation as specified in Corollary 2.7, is in UP ∩ co-UP via a reduction to 2-valued checks similar to the one in [4]. Such a reduction is not possibly in general for $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$ as these decision problems are EXPTIME-complete.

Theorem 3.3 *The decision problems $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$ are EXPTIME-complete in the size of ϕ.*

Proof. For $\mathcal{V} = \{(M, s)\}$, $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$ ask whether some (respectively, all) concretizations of (M, s) satisfy ϕ. So $\mathbb{S}(\mathcal{V}, \phi)$ is the generalized model checking problem $GMC(M, s, \phi)$ of Bruns & Godefroid in [4] and $\mathbb{V}(\mathcal{V}, \phi)$ its dual. Since $GMC(M, s, \phi)$ is EXPTIME-complete for formulas of the modal mu-calculus [4], $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$ are EXPTIME-hard for general $\mathcal{V}$ and ϕ of μL. By Theorem 3.2 the decision problems $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$ are in EXPTIME and so EXPTIME-complete. ∎

4 Efficient consistency checking

Practical considerations suggest to investigate whether the upper bound of Theorem 3.2(ii) can be lowered for $\mathbb{C}(\mathcal{V})$, which we now do for fixed k in (1).

Definition 4.1 (i) We denote $\prod_{i=1}^{k} \Sigma_i$ by $\Sigma_{\mathcal{V}}$, write $\boldsymbol{t}$ for $(t_1, t_2, \ldots, t_k) \in \Sigma_{\mathcal{V}}$, and use $\mathcal{V}_{\boldsymbol{s}}$ to stress that s_i is the initial state in each (M_i, s_i) of $\mathcal{V}$.

(ii) A *common refinement witness* is a relation $W \subseteq \Sigma_{\mathcal{V}}$ such that $\boldsymbol{t} \in W$ implies
   (a) for all i and $q \in AP$, if $t_i \in L^a(q)$ then $t_j \in L^c(q)$ for all $j \neq i$,
   (b) for all i, if $(t_i, t_i') \in R^a$, then there is some $\boldsymbol{t'} \in W$ such that $(t_j, t_j') \in R^c$ for all $j \neq i$.

Note that in clause (b) above the ith coordinate of $\boldsymbol{t'}$ is bound to the given t_i'. As the arbitrary union of common refinement witnesses is a common refinement witness, there is a greatest common refinement witness for each $\mathcal{V}_{\boldsymbol{s}}$, denoted by $W_{\mathcal{V}_{\boldsymbol{s}}}$. This relation captures the existence of common refinements.

Theorem 4.2 *For any $\mathcal{V}_{\boldsymbol{s}}$, the predicate $\mathbb{C}(\mathcal{V}_{\boldsymbol{s}})$ is equivalent to "$\boldsymbol{s} \in W_{\mathcal{V}_{\boldsymbol{s}}}$."*

Proof.

- We begin by showing that $W = \{\boldsymbol{t} \in S_{\mathcal{V}_{\boldsymbol{s}}} \mid \mathcal{C}(\mathcal{V}_{\boldsymbol{t}}) \neq \{\}\}$ is a subset of $W_{\mathcal{V}_{\boldsymbol{s}}}$. Given $\boldsymbol{t} \in W$, there is $(K, k) = ((S_K, R_K, L_K), k) \in \mathcal{C}(\mathcal{V}_{\boldsymbol{t}})$ by the definition of W.
  - Clause (b): For any i, if $(t_i, t_i') \in R^a$, then there is $(k, k') \in R_K$ with $(M_i, t_i') \prec (K, k')$ as $(M_i, t_i) \prec (K, k)$. Since $(M_j, t_j) \prec (K, k)$ for all $j \neq i$ and $(k, k') \in R_K$, there is $(t_j, t_j') \in R^c$ with $(M_j, t_j') \prec (K, k')$ for each $j \neq i$. In particular, $\mathcal{V}_{\boldsymbol{t'}}$ has (K, k') as common refinement so $\boldsymbol{t'} \in W$.
  - A similar reasoning applies to clauses (a) and (c) and so $W \subseteq W_{\mathcal{V}_{\boldsymbol{s}}}$.
- Now we prove the desired equivalence.

(i) If $\mathbb{C}(\mathcal{V}_{\boldsymbol{s}})$ holds, then $\boldsymbol{s} \in W$ by definition and $W \subseteq W_{\mathcal{V}_{\boldsymbol{s}}}$ by the item above.

(ii) Let $\boldsymbol{s} \in W_{\mathcal{V}_{\boldsymbol{s}}}$. We define $K = (W_{\mathcal{V}_{\boldsymbol{s}}}, R, L)$ as follows: $(\boldsymbol{t}, \boldsymbol{t'}) \in R$ iff (for all i, $(t_i, t_i') \in R^c$), and $\boldsymbol{t} \in L(q)$ iff (for all i, $t_i \in L^c(q)$) for $q \in AP$. We claim that $(K, \boldsymbol{s}) \in \mathcal{C}(\mathcal{V}_{\boldsymbol{s}})$ with refinement $\{(t_i, \boldsymbol{t}) \mid \boldsymbol{t} \in W_{\mathcal{V}_{\boldsymbol{s}}}\}$ showing $(M_i, t_i) \prec (K, \boldsymbol{t})$. By definition, any transition from $\boldsymbol{t} \in W_{\mathcal{V}_{\boldsymbol{s}}}$ in K or propositional label at $\boldsymbol{t}$ in K is "c-matched" for t_i in each M_i. Conversely, any a-transition (t_i, t_i') in M_i with $\boldsymbol{t} \in W_{\mathcal{V}_{\boldsymbol{s}}}$ ensures matching c-transitions (t_j, t_j') for all $j \neq i$ such that $\boldsymbol{t'} \in W_{\mathcal{V}_{\boldsymbol{s}}}$ as $\boldsymbol{t} \in W_{\mathcal{V}_{\boldsymbol{s}}}$. So $(\boldsymbol{t}, \boldsymbol{t'}) \in R$ as $R^a \subseteq R^c$ in M_i. Since $\boldsymbol{t'} \in W_{\mathcal{V}_{\boldsymbol{s}}}$ this works co-inductively. A similar argument applies to $t_i \in L^a(q)$ and $t_i \in L^a(n)$. Therefore $(M_i, s_i) \prec (K, \boldsymbol{s})$ for all $i = 1, \ldots, k$ and so $\mathbb{C}(\mathcal{V}_{\boldsymbol{s}})$ holds.

∎

Figure 3 shows an algorithm for computing $W_{\mathcal{V}_{\boldsymbol{s}}}$. This algorithm is related to the partition refinement algorithms for computing the greatest bisimulation relation, see e.g. [32], except that $W_{\mathcal{V}_{\boldsymbol{s}}}$ is not an equivalence relation and so no partition or splitting occurs; e.g., if $\mathcal{V}$ consists of two pointed Kripke models the algorithm non-optimally computes their greatest bisimulation.

Example 4.3 Figure 4 shows two modal Kripke models. Only states s_1 and s_2, and t_1 and t_2 have common refinements.

```
No = {};
let (bad (t, No)) = // fails clause (a) of Definition 4.1(ii):
  ((some i, j, q | t_i in L^a(q) && not t_j in L^c(q))
  ||                // fails clause (b) of Definition 4.1(ii):
   (some (t_i,x) in R^a | all t' in Sigma_V minus No |
      x = t'_i ==> some j | not (t_j,t_j') in R^c
  )) in
  { while (some t in Sigma_V minus No | (bad (t, No))) {
      No = No union {t};
  }
Yes = Sigma_V minus No;
```

Fig. 3. Computing $W_{\mathcal{V}_s}$ for a given set of views $\mathcal{V}_s$, where `union` and `minus` denote set-theoretic union and complement, respectively.

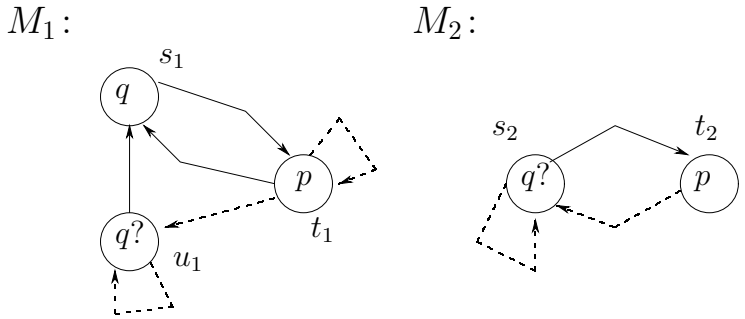

Fig. 4. Two modal Kripke models, where dashed (solid) lines denote elements of $R^c \setminus R^a$ (R^a, respectively). Elements of $L^a(s)$ are written inside states s, as are those of $L^c(s) \setminus L^a(s)$ but annotated with a "?". For example, $q \in L^c(s_2) \setminus L^a(s_2)$ and $p \in L^a(t_1)$. The set $\{(s_1, s_2), (t_1, t_2)\}$ is a common refinement witness and the greatest one as all other elements of $\Sigma_1 \times \Sigma_2$ have no common refinements. For example, for $(u_1, s_2) \in \Sigma_1 \times \Sigma_2$ there is $(s_2, t_2) \in R^a$ and no outgoing R^c transitions from u_1 to a state having a common refinement with t_2.

Theorem 4.4 *The algorithm of Figure 3 terminates after at most* $\mid \Sigma_{\mathcal{V}} \mid$ *iterations and assigns to* `Yes` *the set* $W_{\mathcal{V}_s}$.

Proof. For termination, `Sigma_V minus No` equals `Sigma_V` initially and `No` is a subset of `Sigma_V` that increases by one at each iteration so there cannot be more iterations than elements in `Sigma_V`. It remains to show correctness:

- For $W_{\mathcal{V}_s} \subseteq$ `Yes` it suffices to show $W \subseteq$ `Yes` for any non-empty common refinement witness $W \subseteq S_{\mathcal{V}}$, i.e., that $W \subseteq$ `Sigma_V minus No` is an invariant of the while-statement as $\{\} \neq W$ forces execution of the `if`-branch. The inclusion $W \subseteq$ `Sigma_V minus No` holds initially as then `No` is empty and $W \subseteq$ `Sigma_V`. Assume that $W \subseteq$ `Sigma_V minus No` holds right before an iteration of the while-statement. Given $t \in W$, the expression `(bad (t, No))` is false since t is in the common refinement witness W and the range of the quantifier `all t'` is the set `S_V minus No` and subsumes W by assumption. Thus, no $t \in W$ can be added to `No`.

- For `Yes` $\subseteq W_{\mathcal{V}_s}$ it suffices to show that a non-empty `Yes` is a common refinement witness. After the assignment to the non-empty `Yes`, the expression `(bad (t, No))` is false for all t in `Yes` and the Boolean guard of the `if`-statement is true, so this states that `Yes` is a common refinement witness.

∎

5 Automata games

We would like to obtain efficient approximations for the EXPTIME-complete judgments $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$. Since 2-valued model checking for μL is reducible to determining who has a winning strategy in a parity game [25,30], we seek approximations for $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$ that allow similar reductions.

One may seek such approximations based on the idea of model merging [37]. By imposing a determinacy condition similar to the one used in [28] on models, the process of merging models is able to produce a minimal common refinement $\hat{M}$ for consistent models so that

$$\mathcal{C}(\hat{M}) = \bigcap_{i=1}^{k} \mathcal{C}(M_i)\,. \tag{7}$$

Alas, such determinacy demands severely limit the expressiveness of models.

The idea of model merging can also be applied if no determinacy assumptions are being made. In [19], "summary" models $\mathcal{V}_-$ and $\mathcal{V}_+$ were constructed from the state space $W_{\mathcal{V}_s}$ computed by the algorithm in Figure 3 such that

$$(\mathcal{V}_-, \boldsymbol{s}) \prec (M_i, s_i) \prec (\mathcal{V}_+, \boldsymbol{s}) \tag{8}$$

for all $i = 1, 2, \ldots, k$. So $(\mathcal{V}_-, \boldsymbol{s})$ is a common abstraction and $(\mathcal{V}_+, \boldsymbol{s})$ is a common refinement. Unfortunately, their sets of concretizations are poor approximations of (1) in general, a point we elaborate upon in the next example.

Example 5.1 Consider any two models (M_1, s_1) and (M_2, s_2) whose respective embeddings $\langle\!|\, M_1, s_1 \,|\!\rangle$ and $\langle\!|\, M_2, s_2 \,|\!\rangle$ into the universal domain $\mathcal{D}$ of [23] — achieved by translating these models into modal transition systems first as in [15] and then using the embedding in [23] — are compact elements in $\mathcal{D}$. Since $\mathcal{D}$ is bifinite [23], the set in (1) with $k = 2$ corresponds to the subset $\bigcup_l \Uparrow l$ of $\mathcal{D}$, where l ranges over the finitely many minimal upper bounds of the set $\{\langle\!|\, M_1, s_1 \,|\!\rangle, \langle\!|\, M_2, s_2 \,|\!\rangle\}$. Any sound *under*-approximation of $\mathbb{S}(\mathcal{V}, \phi)$ based on a modal Kripke model would therefore benefit from picking any such minimal upper bound $\hat{l}$ as the "model" since soundness dictates that an upper bound be chosen. However, $\Uparrow \hat{l}$ loses plenty of precision compared to the union $\bigcup_l \Uparrow l$.

The example above demonstrates that the inability to obtain good reductions of $\mathbb{S}(\mathcal{V}, \phi)$ and $\mathbb{V}(\mathcal{V}, \phi)$ to model checks on a single modal Kripke model is linked to the fact that the domain-theoretic model $\mathcal{D}$ of [23] is unlikely to be bounded complete. Consequently, this approach can only deliver limited results. This is were we turn to tree-automata-like models and their refinement

games. For sake of brevity we focus on validity checks only and over-simplify the subsequent technical discussion which relies heavily on the work by Dams & Namjoshi in [10]. The key idea is that

- modal Kripke structures and formulas of μL alike have efficient representations as a kind of tree automata, the focussed transition systems of [10],
- that $\sigma_{\mathcal{V}}$ can be expressed as such a focussed transition system, and
- that the EXPTIME-hard language inclusion of focussed transition systems can be approximated in NP with a certain parity game.

Let $\sigma_{\mathcal{V}}$ be $\bigwedge_{i=1}^{k} [M_i, s_i]$ as in (6). This μL formula has an efficient encoding as a corresponding tree automata $A_{\mathcal{V}}$ that accepts exactly those Kripke models satisfying $\sigma_{\mathcal{V}}$. Similarly, we have a tree automaton A_ϕ for ϕ. These tree automata are then efficiently represented as focussed transition systems [10] $\overline{A_{\mathcal{V}}}$ and $\overline{A_\phi}$, respectively, as detailed in loc. cit. By Theorem 7 of loc. cit., $\overline{A_\phi} \models A_\phi$ for the model checking game in loc. cit., where $\models$ corresponds to our $\models^a$ and therefore under-approximates validity checks. By Theorem 6 of loc. cit., we get $\overline{A_{\mathcal{V}}} \models A_\phi$ provided that $A_{\mathcal{V}}$ refines A_ϕ (written $A_{\mathcal{V}} \sqsupseteq A_\phi$ in [10]) for the abstraction game moves in Figure 3 of loc. cit. Thus, one can under-approximate $\mathbb{V}(\mathcal{V}, \phi)$ with the (parity) game check $A_{\mathcal{V}} \sqsupseteq A_\phi$ of loc. cit.

6 Related work

Uchitel & Chechik [37] merge a variant of modal transition systems with overlapping but different sets of event signatures (the AP in our state-based setting) to obtain a minimal common refinement and suggest user participation to explore common behavior if no minimal common refinement exists. Their models are more general in that events may differ in views, but less general than ours in that we compute the space of all consistent tuples and make efficient model checking possible. They stress engineering activities in model elaboration, we use static analysis and identify the complexities of the relevant decision problems.

Dams & Namjoshi [11] propose modal μ-automata as abstractions of Kripke structures and use a simulation relation in NP for such automata to approximate EXPTIME-hard language inclusion. Our setting favors focussed transition systems as μ-automata correspond to distributive formulas in μL [24], which have linear satisfiability check, but neither $\sigma_{\mathcal{V}}$ nor $\sigma_{\mathcal{V}} \wedge \phi$ are distributive so their conversion into this format may be expensive.

Larsen et al. use projective views whose conjunction recovers the projected modal transition system [28].

Fitting uses a partial order of experts to constrain the consistency of experts' assertions about the truth and falsity of transitions and state observables in multiple-valued Kripke structures [13].

Chechik et al. endow Fitting's models with a semantics for negation drawn from a De Morgan lattice negotiated among experts. For these models they

devise a multiple-valued version of computation tree logic and its symbolic model checking algorithm [5].

Nentwich et al. developed the tool `xlinkit` that analyzes distributed XML documents for possible inconsistencies, based on rules written in first-order logic [31].

Guerra [17] proposes a specification framework for software artifacts, where specifications have defaults and allow for exceptions stemming from the reuse or evolution of system demands. In [17] specifications are written in linear-time temporal logic [33] and a non-monotonic semantics for this logic is defined based on default institutions [16], where the semantics of defaults is given by a generalized distance between interpretations.

For modal transition systems and the modal mu-calculus, the decision problems of this paper have already been defined in [21] and the reduction to satisfiability in the modal mu-calculus for common refinement checks has been stated in [20]. In loc. cit. it is also shown that the sets $\mathcal{C}(M)$ are compact in the quotient space of bisimulation for the natural metric based on testing formulas of μL without fixed points; in particular, all sets in (1) are compact even for infinite-state models.

Part of this paper's material, notably Sections 3 and 4, is a customization of results that appeared in a technical report [19]. In loc. cit. a more general notion of model was considered in which some propositions of 2-valued models are nominals, true at exactly one state. The modal mu-calculus used in loc. cit. was therefore the hybrid mu-calculus of Sattler and Vardi [36].

7 Conclusions

We determined the complexities of consistency, satisfiability, and validity checking on the sets of common concretizations of finitely many finite-state models as PTIME for a fixed number of models, EXPTIME-complete, and EXPTIME-complete (respectively). We discussed the limitations of existing approximations of the two EXPTIME-complete decision problems and pointed out that focussed transition systems and their refinement games should be more precise approximations than those found in the extant literature.

Acknowledgement

Sebastian Uchitel helped us with understanding the work in [37]. Glenn Bruns is thanked for discussions on databases and abstract views.

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, France, July 2001. Springer-Verlag.

[2] J. C. Bradfield. *Verifying Temporal Properties Of Systems.* Birkhäuser, Boston, Massachusetts, 1991.

[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] M. Chechik, B. Devereux, A. Gurfinkel, and S. Easterbrook. Multi-Valued Symbolic Model-Checking. *ACM Transactions on Software Engineering and Methodology*, 12(4):1–38, October 2003.

[6] E. M. Clarke and E. A. Emerson. Synthesis of synchronization skeletons for branching time temporal logic. In D. Kozen, editor, *Logic of Programs Workshop*, volume 131 of *LNCS*. Springer Verlag, 1981.

[7] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs. In *Proceedings of the 4th ACM Symp. on Principles of Programming Languages*, pages 238–252. ACM Press, 1977.

[8] D. Dams. *Abstract interpretation and partition refinement for model checking.* PhD thesis, Technische Universiteit Eindhoven, The Netherlands, 1996.

[9] D. Dams, R. Gerth, and O. Grumberg. Abstract interpretation of reactive systems. *ACM Transactions on Programming Languages and Systems*, 19(2):253–291, 1997.

[10] D. Dams and K. Namjoshi. The Existence of Finite Abstractions for Branching Time Model Checking. In *Proceedings of the Nineteenth Annual IEEE Symposium on Logic in Computer Science*, pages 335–344, Turku, Finland, 13-17 July 2004. IEEE Computer Society Press.

[11] D. Dams and K. S. Namjoshi. Automata as Abstractions. In R. Cousot, editor, *Proceedings of 6th International Conference on Verification, Model Checking and Abstract Interpretation*, volume 3385 of *Lecture Notes in Computer Science*, pages 216–232, Paris, France, 17-19 January 2004. Springer Verlag.

[12] E. A. Emerson and C. S. Jutla. The complexity of tree automata and logics of programs. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, pages 328–337, White Plains, New York, 1988.

[13] M. Fitting. Many-valued modal logics II. *Fundamenta Informaticae*, 17:55–73, 1992.

[14] 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.

[15] 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 4th Conference on Verification, Model Checking and Abstract Interpretation*, volume 2575 of *LNCS*, pages 206–222, New York, January 2003. Springer Verlag.

[16] J. A. Goguen and R. M. Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the ACM*, 39(1):95–146, January 1992.

[17] S. Guerra. Distance Functions for Defaults in Reactive Systems. In T. Rus, editor, *Proceedings of the 8th International Conference on Algebraic Methodology and Software Technology*, volume 1816 of *Lecture Notes in Computer Science*, pages 26–40, Iowa City, Iowa, May 2000. Springer Verlag.

[18] 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.

[19] 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 (invited journal version in preparation)*, pages 235–242, Paphos, Cyprus, 30 October - 2 November 2004. Technical Report TR-2004-6 Department of Computer Science, University of Cyprus.

[20] M. Huth. Labelled Transition Systems as a Stone Space. *Logical Methods in Computer Science*, 1(1):1–28, 26 January 2005. www.lmcs-online.org.

[21] M. Huth. Refinement is complete for implementations. Accepted for publication in *Formal Aspects of Computing*, December 2005.

[22] 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 European Symposium on Programming*, pages 155–169. Springer Verlag, April 2001.

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

[24] D. Janin and I. Walukiewicz. Automata for the modal mu-calculus and related results. In *MFCS '95: 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.

[25] M. Jurdziński. Deciding the winner in parity games is in UP ∩ co-UP. *Information Processing Letters*, 68(3):119–124, 1998.

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

[27] K. G. Larsen. Modal Specifications. In J. Sifakis, editor, *Automatic Verification Methods for Finite State Systems*, volume 407 of *Lecture Notes in Computer Science*, pages 232–246. Springer Verlag, June 12–14 1989. International Workshop, Grenoble, France.

[28] K. G. Larsen, B. Steffen, and C. Weise. A Constraint Oriented Proof Methodology Based on Modal Transition Systems. In E. Brinksma, R. Cleaveland, K. G. Larsen, T. Margaria, and B. Steffen, editors, *Tools and Algorithms for Construction and Analysis of Systems, 1st International Workshop*, volume 1019 of *Lecture Notes in Computer Science*, pages 17–40, Aarhus, Denmark, 19-20 May 1995. Springer Verlag.

[29] K. G. Larsen and B. Thomsen. A Modal Process Logic. In *Proceedings of the 3rd Annual Symposium on Logic in Computer Science*, pages 203–210. IEEE Computer Society Press, 1988.

[30] D. E. Long, A. Browne, E. M. Clarke, S. Jha, and W. R. Marrero. An improved algorithm for the evaluation of fixpoint expressions. In D. L. Dill, editor, *Proceedings of the 6th International Conference on Computer Aided Verification*, volume 818 of *Lecture Notes in Computer Science*, pages 338–350, Stanford, California, 1994. Springer Verlag.

[31] C. Nentwich, L. Capra, W. Emmerich, and A. Finkelstein. xlinkit: a consistency checking and smart link generation service. *ACM Transactions on Internet Technology*, 2(2):151–185, 2002.

[32] D. Peled. *Software Reliability Methods*. Springer Verlag, New York City, New York, 2001.

[33] A. Pnueli. The temporal logic of programs. In *Proceedings of the 18th IEEE Symposium on the Foundations of Computer Science*, pages 46–57, 1977.

[34] J. P. Quielle and J. Sifakis. Specification and verification of concurrent systems in CAESAR. In *Proceedings of the 5th International Symposium on Programming*, 1981.

[35] M. Sagiv, T. Reps, and R. Wilhelm. Parametric Shape Analysis via 3-Valued Logic. In *Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of programming languages*, pages 105–118, January 20-22, San Antonio, Texas 1999.

[36] U. Sattler and M. Vardi. The Hybrid μ-calculus. In R. Goré, A. Leitsch, and T. Nipkov, editors, *Proceedings of the 1st International Joint Conference on Automated Reasoning*, volume 2083 of *Lecture Notes in Computer Science*, pages 76–91, Siena, Italy, 18-23 June 2001. Springer Verlag.

[37] S. Uchitel and M. Chechik. Merging Partial Behavioural Models. *ACM SIGSOFT Notes*, 29(6):43–52, 2004.