On model checking multiple hybrid views

Altaf Hussain and Michael Huth

Department of Computing, South Kensington campus, Imperial College London,
London, SW7 2AZ, United Kingdom, {ah701, mrh}@doc.imperial.ac.uk

Abstract. We study consistency, satisfiability, and validity problems for collectively model checking a set of views endowed with labelled transitions, hybrid constraints on states, and atomic propositions. A PTIME algorithm for deciding whether a set of views has a common refinement (consistency) is given. We prove that deciding whether a common refinement satisfies a formula of the hybrid mu-calculus (satisfiability), and its dual (validity), are EXPTIME-complete. We determine two generically generated "summary" views that constitute informative and consistent common refinements and abstractions of a set of views (respectively). Finally we reflect on leveraging these results to applications in the future.

1 Introduction

In model checking [9, 33] one builds a model $M$ of an artefact’s state and behavior, expresses desired properties of such state and behavior in a temporal logic or some variant thereof, and verifies (refutes) such a property by verifying (refuting) the instance $M \models \phi$ of a satisfaction relation $\models$.

Often not all aspects of state or behavior are known at the time of model construction. For example, a concrete program may be abstracted by a Boolean program [1], or requirements for a design may leave details intentionally underspecified [24]. In such situations, models benefit from being 3-valued so that state and behavior can take on values true (guaranteed), false (impossible) or $\perp$ (possible). Key benefits are an explicit under-specification through $\perp$, soundness of abstraction-based model checks for properties that mix path quantifiers [25], and reliable feasibility checks for counter-examples and simulations (see e.g. [30]).

By now it is well understood that many such 3-valued notions of models are equally expressive as model-checking frameworks [14]. The additional value $\perp$ in models also blurs the boundaries between abstraction and parameterized model checking since a single model may express infinitely many non-equivalent 2-valued systems as its refinements. To appreciate this point we recall some of the most frequent uses of parameters in model checking: e.g. parameters may denote

- the number of protocols, for example in cache coherence protocols,
- the length of a queue, for example in the Bakery protocol, or
- the topology of a network.

Under-specified systems describe sets of completely specified systems and this abstraction may be used to achieve the above usages of parameters. See
Figure 1 for an illustration where the system to the left can be seen as specifying a parametric agent, a pub user, and the refining system to the right realizes a scenario with two actual pub users [18].

For 3-valued models $M$, the judgment $M \models \phi$ becomes 3-valued as well and may be written as two predicates $V(M, \phi)$ and $S(M, \phi)$ where the intention of $V(M, \phi)$ is to assert $\phi$ for $M$: "$\phi$ holds in all 2-valued refinements of $M"$ whereas its dual $S(M, \phi)$ states that $\phi$ is consistent for $M$: "some 2-valued refinement of $M$ satisfies $\phi."$ The judgment $S(M, \phi)$ is the generalized model checking of Bruns \& Godefroid [5] and EXPTIME-complete for formulas $\phi$ of the propositional modal mu-calculus and partial Kripke structures as 3-valued models $M$ [5]. The compositional approximation of these judgments, the 3-valued compositional model checking algorithm given in [4], can be reduced to two 2-valued checks and therefore completed in linear time [5]. Deciding whether a 3-valued model has a 2-valued refinement, the answer to $S(M, \text{true})$, is trivial as $S(M, \text{true})$ is always true due to the consistency implicit in the definition of partial Kripke structures or any other variant of such models [27,14].

This paper initiates a re-development of this programme of 3-valued compositional model checking, generalized model checking, and deciding the existence of refinements in a setting where not just one $M$ but finitely many 3-valued views $M_i$ ($1 \leq i \leq k$) are given and where we wish to reason about these views
collectively. Assuming a notion of refinement between 3-valued views, Bruns &
Godefroid’s generalized model checking then reads:

For a property $\phi$, is there a 2-valued refinement of all $\mathcal{M}_i$ satisfying $\phi$? \hfill (1)

In particular for $\phi$ being true, this specializes to

Is there a 2-valued refinement of all $\mathcal{M}_i$? \hfill (2)

In this paper we define views as certain mathematical models. This formality and
generality will pay off in leveraging the results presented here, for views occur
in many different settings. A view may be the local and relative perception
of asynchronous communication as modelled by a message sequence chart, the
‘view’ of a local database, the high-risk requirements of a certain stakeholder in
a development project, etc. Reasoning about the collection of such views, e.g.
determining their consistency or locating conflicts, is therefore of considerable
practical value. We hope that our model of views is expressive enough to deal
with such settings gracefully.

We show that the decision problem in (2) is in PTIME and that the one
in (1) is EXPTIME-complete for the hybrid mu-calculus of Sattler & Vardi [34].
Our notion of view is a variant of Kripke modal transition systems [20] where
some atomic state propositions are nominals, true at exactly one state. Nominals
allow for the modelling of agents and their movement, XML documents etc. The
satisfiability of $\phi$ reflects the hybrid constraints on nominals without increasing
the EXPTIME upper bound for the modal mu-calculus, due to a result in [34].

The PTIME algorithm for consistency checking developed below computes
those tuples of the product state space of all $\mathcal{M}_i$ that have a common refinement.
This subset is the state space of “summary” views that serve as informative
consistent common refinements and abstractions of all $\mathcal{M}_i$ (respectively).

Outline of this paper: In Section 2 we define 3-valued views and their 3-valued
compositional property semantics. In Section 3 we define refinement between
views and prove that the 3-valued compositional property semantics is sound for,
and logically characterizes, refinement. In Section 4 we define generalized model
checking and consistency checking for a finite set of views and show that these
problems are reducible to satisfiability checking in the hybrid mu-calculus and
that generalized model checking is EXPTIME-complete and so no more complex
than generalized model checking for the modal mu-calculus and a single model.
In Section 5 we develop an efficient algorithm for consistency checking which
computes all tuples of states that have a common refinement. The algorithm for
consistency checking is used in Section 6 to define “summary” views that are
informative common refinements (respectively, abstractions) of a given set of
views. In Section 7 we state related work. We sketch future work in the context
of leverage to applications in Section 8 and conclude in Section 9.
2 Views and their property semantics

We define the 2-valued models that express hybrid views of an artefact, essentially those given by Sattler & Vardi in [34]. For the remainder of this paper let $AP$, $Nom$, and $Act$ be mutually disjoint finite sets of state propositions, nominals, and events (respectively) with a designated universal event $u \in Act$ which has transitions between any pair of states. The idea is that $AP \cup Nom$ and $Act$ are observables that annotate states and transitions (respectively). The hybrid nature of nominals $n \in Nom$ comes from restricting the class of 2-valued views to those at which each $n \in Nom$ is true (i.e. annotated) at exactly one state. The universal event $u$ enables one to express familiar hybrid logic operators in the hybrid mu-calculus. For example “at nominal $n$, $\phi$ holds” is $[u](\neg n \lor \phi)$ since all states have transitions labelled with $u$ to all states [34].

**Definition 1.** A 2-valued view $M$ is a tuple $(S, R, L)$ where $S$ is a set of states, $R \subseteq S \times Act \times S$ is a transition relation with $\{(s, s') \in S \times S \mid (s, u, s') \in R\} = S \times S$, and $L: AP \cup Nom \rightarrow \mathcal{P}(S)$ is a labelling function such that the subset $L(n)$ of $S$ is a singleton for all $n \in Nom$.

**Example 1.** Figure 2 depicts a 2-valued view with initial state $(s_1, s_2)$, $L(n) = \{(u_1, t_2)\}$, $((s_1, s_2), \alpha, (u_1, t_2)) \in R$, etc. Throughout this paper, figures omit transitions for the universal event $u$.

![Fig. 2. A 2-valued view with $AP = \{q_1, q_2\}$, $Nom = \{n\}$, $Act = \{\alpha, \beta, u\}$, and three states $(s_1, s_2)$, $(u_1, t_2)$, and $(t_1, s_2)$. It is a common completion of the views in Figure 6. Throughout this paper, figures omit transitions for the universal event $u$.](image)

The hybrid mu-calculus of [34] is defined in negation normal form and its expressiveness relates it closely to description logics [28] which are used in knowledge representation. Here we define its grammar with an unrestricted clause for negation. Hybrid extensions of branching-time temporal logics, e.g. hybrid CTL [12, 17] and many domain-specific property languages, are all expressible in the hybrid mu-calculus (hMC) whose grammar is

$$\phi ::= q \mid Z \mid \neg \phi \mid \phi \land \phi \mid \langle \alpha \rangle \phi \mid \mu Z. \phi \tag{3}$$
where $q \in AP \cup Nom$, $\alpha \in Act \cup Act^\ast$ with $\overline{Act} = \{ \overline{\alpha} \mid \alpha \in Act \}$, and $Z$ ranges over a countable set of recursion variables. For $\alpha \in Act$, $\overline{\alpha}$ denotes the “inverse” event; its semantics is given by extending the definition of $R$ for a 2-valued view $M = (S, R, L)$ through $((s, \overline{\alpha}, s') \in R)$ iff $(s', \alpha, s) \in R)$. For $\alpha \in Act$, we let $\overline{\alpha}$ be $\alpha$ again. We write true for $q \lor \neg q$, $q \lor \neg q$ for $\neg (\neg \phi \land \neg \phi_2)$, $\phi_1 \lor \phi_2$ for $\neg (\phi_1 \land \neg \phi_2)$, and, for all $\alpha \in Act \cup Act^\ast$, $[\nu \phi]$ for $\neg (\alpha \land \neg \phi)$. In the least fixed-point formula $\mu Z. \phi$, $\mu Z$ binds all occurrences of $Z$ in $\phi$ with static scoping and we require that all free occurrences of $Z$ in $\phi$ are under an even scope of negations. For example, $\mu Z. \neg (p \lor Z)$ is not allowed as $Z$ in $\neg (p \lor Z)$ is under the scope of one (odd) negation. If $\phi[Z/\psi]$ denotes the formula obtained from $\phi$ by replacing all free occurrences of $Z$ in $\phi$ with $\psi$, the greatest fixed-point formula $\nu Z \phi[Z/\psi]$ is derived as $\nu Z. \phi[Z/\psi]$. A formula $\phi$ is closed if it contains no free recursion variables. The semantics of formulas over 2-valued views is a special case of that for 3-valued views so we first define those models and their semantics.

**Definition 2.** 1. A 3-valued view $M$ is a pair $(M^a, M^c)$ with $M^a = (S, R^a, L^a)$ and $M^c = (S, R^c, L^c)$ where $S$ is a set of states, the relations $R^a, R^c \subseteq S \times (Act \cup Act^\ast) \times S$ specify transitions such that $R^c \subseteq R^a$, $\{(s, s') \in S \mid (s, u, s') \in R^a\} = S \times S$, and $\{(s, \overline{\alpha}, s') \in R^a \}$ for $m \in \{a, c\}$, and $L^a, L^c$: $AP \cup Nom \rightarrow \mathcal{P}(S)$ are labelling functions with $L^a(q) \subseteq L^c(q)$ for all $q \in AP \cup Nom$ subject to the following constraints: (a) for all $n \in Nom$, $L^a(n)$ contains at most one state and $L^c(n) \neq \{\}$, (b) for all $n \in Nom$, if $L^a(n)$ is non-empty, $L^a(n) = L^c(n)$.

2. A 3-valued view $M$ with an associated initial state $s \in S$ is pointed: $(M, s)$.

The intuition behind 3-valued views is that $a$-structure specifies asserted (guaranteed, valid etc) information, whereas $c$-structure declares consistent (possible satisfiable etc) information [27]. The complement of $c$-structure specifies an impossibility, e.g. $(s, \overline{\alpha}, s') \notin R^c$ expresses that event $\alpha$ cannot lead from state $s$ to state $s'$. Note that the universal event $u$ is asserted to be so since Definition 2 enforces $\{(s, s') \in S \mid (s, u, s') \in R^a\} = S \times S$. The inclusions $R^a \subseteq R^c$ ensure logical consistency of the semantics given for hMC below. The constraints (a) and (b) on nominal labels are based on the reading of the expressions $s \in L^a(n)$ and $s \in L^c(n)$ as “$n$ is true at $s$” and “$n$ may be true at $s$” respectively [17].

**Example 2.** Figure 5 shows 3-valued views $(M_1, s_2)$ and $(M_2, s_2)$. In all figures, transitions for all $\overline{\alpha} \in Act^\ast$ are implied, and solid and dashed lines denote $R^a$- and $R^c$-transitions (respectively). For $q \in AP \cup Nom$, a label $q$, or its absence at state $t$ denote $t \in L^a(q)$, $t \in L^c(q)$ \ $L^a(q)$, and $t \notin L^c(q)$ (respectively).

The denotational semantics $\llbracket \cdot \rrbracket^m$ of hMC over 3-valued views maps formulas $\phi$ and environments $\rho$, functions $Z \mapsto (\rho[Z], \rho'(Z))$ of type $Var \rightarrow \{(L, U) \subseteq S \times S \mid L \subseteq U\}$, into sets of states for a mode of analysis $m \in \{a, c\}$ and is illustrated in Figure 3. Note that $\neg a = c$, $\neg e = a$, and $\preceq^m(A) = \{s \in S \mid \exists s' \in A: (s, \alpha, s') \in R^m\}$ for all $\alpha \in Act \cup Act^\ast$ and $m \in \{a, c\}$.

**Definition 3.** For $m \in \{a, c\}$, we often write $(M, s) \models^m \phi$ or $s \models^m \phi$ for $s \in \llbracket \cdot \rrbracket^m_\rho$, and omit $\rho$ in $\models^m_\rho$ if $\phi$ is closed.
\[ q^{\rho} = L^{\rho}(q) \]
\[ \phi^{\rho} = \Sigma \setminus \phi^{\rho_{m}} \]
\[ \langle\alpha\rangle^{\rho} = \text{pre}^{\alpha}(\phi^{\rho_{\alpha}}) \]
\[ Z^{\rho} = \rho^{\alpha}(Z) \]
\[ \phi_{1} \land \phi_{2}^{\rho_{m}} = \phi_{1}^{\rho_{m}} \cap \phi_{2}^{\rho_{m}} \]
\[ \mu Z.\phi^{\rho} = \text{Ifp} \lambda A.\phi^{\rho_{\{Z \mapsto A\}}} \]

**Fig. 3.** Semantics of hMC over 3-valued views for mode \( m \in \{a, c\} \). The environment \( \rho[Z \mapsto A] \) equals \( \rho \) except that \( \rho[Z \mapsto A]^{\alpha}(Z) = A \).

Least fixed points \( \text{Ifp} \lambda A.\phi^{\rho_{\{Z \mapsto A\}}} \) are formed in the complete lattice \( (\mathcal{P}(\Sigma), \subseteq) \). Note that for \( m \in \{a, c\} \) we have \( s|^{m} = [\alpha]_{\phi} \) iff for all \( (s, \alpha, s') \in R^{m} \), \( s'|^{m} = [\phi]_{\alpha} \).

If \( K \) is a 2-valued view, we may cast it up into a 3-valued view \( M \) with \( M^{a} = K \). Then \( \phi^{\rho} = \phi^{\rho_{\alpha}} \) holds in \( M \) for all \( \rho \) and \( \phi \) of hMC so this defines the 2-valued semantics \( k \models \phi \) to be \( k \in \phi^{\rho} \) for all states \( k \) of \( K \).

**Example 3.** For \((M_{2}, s_{2})\) in Figure 5 we have \( s_{2}|^{m} = [\beta](\alpha)_{\text{true}} \) since we don’t have \( s_{2}|^{m} = [\beta](\alpha)_{\text{true}} \), for \((s_{2}, \alpha, x) \in R^{a} \) implies \( x = t_{2}, (t_{2}, \beta, t_{2}) \in R^{c} \), but there is no \( R^{a}\)-transition labelled with \( \alpha \) out of \( t_{2} \). We have \( s_{2}|^{m} = [\beta]_{\langle\beta\rangle(\alpha)_{\text{true}}} \) and \( \langle\beta\rangle(\alpha)_{\text{true}} \) since there is no \( (s_{2}, \beta, x) \in R^{c} \) and so \( s_{2}|^{m} = [\beta]_{\langle\beta\rangle(\alpha)_{\text{true}}} \), and there is a \( R^{a}\)-cycle \((s_{2}, \beta, t_{2})(t_{2}, \beta, s_{2})\) from \( s_{2} \) to \( s_{2} \) that generates the word \( \beta \alpha \).

### 3 Refinement between views

In specifying a 3-valued view we implicitly describe a possibly infinite set of 2-valued views. Such intuitions can be formalised in Cousot & Cousot’s framework of abstract interpretation [10] or through a co-inductive definition of refinement, as carried out by Larsen & Thomsen in [27]. We pursue the latter approach here.

**Definition 4.** For \( i = 1, 2 \) let \((M_{i}, s_{i}) = (\{(S_{i}, R_{i}^{c}, L_{i}^{a}), (S_{i}, R_{i}^{a}, L_{i}^{c})\}, s_{i})\) be pointed 3-valued views. Then \((M_{1}, s_{1})\) is refined by \((M_{2}, s_{2})\) iff there is a relation \( Q \subseteq S_{1} \times S_{2} \) such that \((s_{1}, s_{2}) \in Q \) and, for all \((s, t) \in Q \), we have

1. for all \( q \in AP \cup \text{Nom} \), \( s \in L_{1}^{a}(q) \) implies \( t \in L_{2}^{a}(q) \),
2. for all \( q \in AP \cup \text{Nom} \), \( t \in L_{2}^{a}(q) \) implies \( s \in L_{1}^{a}(q) \),
3. for all \( \alpha \in \text{Act} \cup \text{Act}^\alpha \), if \((s, \alpha, s') \in R_{1}^{a} \), there is \((t, \alpha, t') \in R_{2}^{a} \) with \((s', t') \in Q \),
4. for all \( \alpha \in \text{Act} \cup \text{Act}^\alpha \), if \((t, \alpha, t') \in R_{2}^{a} \), there is \((s, \alpha, s') \in R_{1}^{a} \) with \((s', t') \in Q \).

We write \((M_{1}, s_{1}) \prec (M_{2}, t)\) whenever there is such a \( Q \) with \((s, t) \in Q \) and denote by \( C(M, s) \) the completions of \((M, s)\), the set of those refinements \((N, t)\) of \((M, s)\) that are 2-valued up to casting.

**Example 4.** The pointed 3-valued view \((V_{+}, (s_{1}, s_{2}))\) of Figure 7 refines the views \((M_{1}, s_{2})\) and \((M_{2}, s_{2})\) of Figure 6 where refinement relates states from hybrid views to their tuple state: \( s_{1} \) to \((s_{1}, s_{2})\), \( t_{2} \) to \((u_{1}, t_{2})\) etc.
As refinement is transitive, \( (M_1, s) \prec (M_2, t) \) implies \( C(M_2, t) \subseteq C(M_1, s) \) and the converse has been proved for views without nominals [19]. Proofs from the non-hybrid setting, showing that the closed fixed-point free formulas characterize refinement and that refinement is sound with respect to the compositional 3-valued property semantics [25], also apply to our setting.

**Theorem 1.** For pointed 3-valued views \( (M, s) \) and \( (N, t) \) we have \( (M, s) \prec (N, t) \) iff (for all closed, fixed-point free formulas \( \phi \) of hMC, \( s \in \models \phi \models^a \) implies \( t \in \models \phi \models^a \)). If \( (M, s) \prec (N, t) \), then \( s \in \models \phi \models^a \) implies \( t \in \models \psi \models^c \), and \( t \in \models \psi \models^c \) implies \( s \in \models \psi \models^c \), for all closed \( \phi, \psi \) of hMC.

This logical characterization and soundness secure soundness of \( \models \phi \models^m \) relative to the thorough semantics of Bruns & Godefroid in [5] adapted to our hybrid setting, where for any closed \( \phi \) of hMC the predicate \( GMC(M, s, \phi) \) means “\( k \models \phi \) for some \( (K, k) \in C(M, s) \).”

**Corollary 1.** For any closed \( \phi \) in hMC and state \( s \) of any 3-valued view \( M \):

1. If \( s \in \models \phi \models^a \), then \( GMC(M, s, \neg \phi) \) is false.
2. If \( GMC(M, s, \phi) \) is true, then \( s \in \models \phi \models^c \).

**Example 5.** For \( (M_2, s_2) \) from Figure 6 we have \( s_2 \models^a (\beta) \) true \( \lor \neg (\beta) \) true as there is no \( (s_2, \beta, x) \in R^a \) but \( (s_2, \beta, t_2) \in R^c \). The formula is a tautology and so true for all completions of \( (M_2, s_2) \). Thus the converse of item 1 above is not true.

## 4 Multiple views and their decision problems

We can now define the decision problems studied in this paper. Let \( V = \{ (M_i, s_i) \mid 1 \leq i \leq k \} \) be any finite set of pointed 3-valued views \( (M_i, s_i) \) where all \( (M_i, s_i) \in V \) have only finitely many states. Each \( (M_i, s_i) \) may be an abstraction of a concrete and consistent artefact, e.g. a piece of software. Alternatively, each \( (M_i, s_i) \) may be the description of a hybrid knowledge-representation or database view (where hybrid constraints are needed in XML documents) or the description of a particular stake holder in the sense of requirements engineering.

In the case of software, consistency of all \( (M_i, s_i) \in V \) is usually assured as the artefact is a consistent common refinement by construction, up to casting and representational changes. In the latter cases, checking consistency is a primary concern in crafting a model that honors all views \( (M_i, s_i) \in V \). For example, if each \( (M_i, s_i) \) is an answer to a query from a local database, \( V \) is often not consistent. We identify the relevant decision problems.

**Definition 5.** Let \( C(V) = \{ (N, t) \mid \forall (M, s) \in V : (N, t) \in C(M, s) \} \) be the set of common completions of \( V \). For closed \( \phi \) of hMC, we define parameterized boolean expressions \( C(V), S(V, \phi), \) and \( V(V, \phi) \):

1. Consistency: \( C(V) \) holds iff all views of \( V \) have a common completion, i.e. \( \| C(V) \| \neq \{ \} \).
2. Satisfiability: $S(V, \phi)$ is true iff there is a common completion of $V$ that satisfies $\phi$, i.e. $\{ (N, t) \in C(V) \mid t \models \phi \} \neq \{ \}$.  
3. Validity: $\forall(V, \phi)$ holds iff all common completions of $V$ satisfy $\phi$.  

Since all pointed 3-valued $(M, s)$ have a completion, e.g. obtained from $(M^s, s)$ by “implementing” all hybrid constraints, $C(V)$ holds iff all views of $V$ have a common refinement. Note that $\forall(V, \phi)$ holds for all $\phi$ if $V$ has no common refinement. Thus it is wise to first decide $C(V)$ so as to avoid unintended certifications through $\forall(V, \phi)$. We show that all three decision problems above are reducible to satisfiability checks of hMC over 2-valued views. Following (3) in [25] we construct a closed formula $[M, s]_i$ of hMC for each pointed 3-valued view $(M, s_i)$ such that for all pointed 3-valued views $(N, t)$ we have

$$(N, t) \models [M, s]_i \iff (M, s_i) \prec (N, t).$$

The existence of such characteristic formulas secures the desired reductions.

**Theorem 2.** 1. Each pointed 3-valued view $(M, s_i)$ has a formula $[M, s]_i$ of the hybrid mu-calculus satisfying (4) for all pointed 3-valued views $(N, t)$.  
2. The decision problems $C(V)$, $S(V, \phi)$, and $\forall(V, \phi)$ are reducible to satisfiability checks of hMC over 2-valued views and in EXPTIME.

**Example 6.** For $(M_2, s_2)$ from Figure 6 we express $[M_2, s_2]$ in hMC with $Z_{a_1}, Z_{a_2} \in \text{Var}$ as $[M_2, s_2] = \nu Z_{a_1}.(\langle \alpha \rangle \psi \land [\alpha] \psi \land [\beta] \psi \land q_1) \land \neg \psi$ where $\psi = \nu Z_{a_2}.(\langle \alpha \rangle Z_{a_1} \land [\alpha] Z_{a_2} \land [\beta] Z_{a_2} \land n \land \neg q_1 \land \neg q_2$.

The semantics of Figure 3 is in NP and in co-NP via a reduction to 2-valued checks similar to the one in [5]. Such a reduction is not possible in general for $S(V, \phi)$ and $\forall(V, \phi)$ as they are EXPTIME-complete.

**Theorem 3.** $S(V, \phi)$ and $\forall(V, \phi)$ are EXPTIME-complete in the size of $\phi$.

5 Complexity of common refinement checks

Practical considerations suggest to investigate whether the upper bound of Theorem 2 can be lowered for $C(V)$, which we now do.

**Definition 6.** Let $V = \{ (M_i, s_i) \mid 1 \leq i \leq k \}$, each $M_i$ having state space $S_i$.  
1. We denote the product space $\prod_{i=1}^k S_i$ by $S_V$, write $t$ for tuples $(t_1, t_2, \ldots, t_k) \in S_V$, and use $V_*$ to stress that $s_i$ is the initial state in each $(M_i, s_i)$ of $V$.  
2. A common refinement witness is a relation $W \subseteq S_V$ such that $t \in W$ implies (a) for all $i$ and $q \in AP \cup \text{Nom}$, if $t_i \in L^a(q)$ then $t_j \in L^a(q)$ for all $j \neq i$, (b) for all $i$ and $\alpha \in \text{Act} \cup \text{Act}$, if $(t_i, \alpha, t'_i) \in R^\alpha$, there is $t' \in W$ such that $(t_j, \alpha, t'_j) \in R^\alpha$ for all $j \neq i$, and (c) for all $n \in \text{Nom}$, there is $t \in W$ such that for all $i$, $t_i \in L^a(n)$.
Note that in clause (b) above the *ith* coordinate of *t′* is bound to the given *tᵢ* and that clause (c) is required as each *n* holds in *some* state of *K* for any (*K, k*) ∈ *C(V)*. As the arbitrary union of common refinement witnesses is a common refinement witness, there is a greatest common refinement witness for each *Vₓ*, denoted by *Wₓ*. This relation captures the existence of common refinements.

**Theorem 4.** For any *Vₓ*, the predicate *C(Vₓ)* is equivalent to “*s ∈ Wₓ*.”

Figure 4 shows an algorithm for computing *Wₓ* where we omitted any optimizations for sake of clarity. This algorithm is related to partition refinement algorithms for computing the greatest bisimulation relation (see e.g. [31]), except that *Wₓ* is not an equivalence relation and so no partition or splitting occurs. However if *V* consists of two pointed 2-valued views, the algorithm is a non-optimal version of the familiar splitting algorithm for bisimulation.

```plaintext
No = Ø;
let bad (t, No)) = // fails clause (a) of Definition 6.2:
((some i, j, q | tᵢ in L⁻(q) &\ not tⱼ in L⁻(q))
 } // fails clause (b) of Definition 6.2:
(some (tᵢ,a,x) in R⁻ | all t' in S_V minus No |
 x ≠ tᵢ' i → some j | not (tⱼ,a,tⱼ') in R⁻)
 in
{| while (some t in S_V minus No | (bad (t, No))) |
No = No union {t};
} // else-branch: fails clause (c) of Definition 6.2
if (all n in Nom | some t' in S_V minus No | all i | tᵢ in L⁻(n))
{ Yes = S_V minus No; } else { Yes = {};
}
```

**Fig. 4.** Computing *Wₓ* for a given set of views *Vₓ*, where union and minus denote set-theoretic union and complement, respectively.

**Theorem 5.** The algorithm of Figure 4 terminates after at most | *S_V* | iterations and assigns to *Yes* the set *Wₓ*.

**Example 7.** Let *Act* = {α, β, u} and |*Act*| = {α̃, β̃, ũ}.

1. Let *V* = {(*M₁, s₁*), (*M₂, s₂*)} with *S₁* = {s₁, t₁, u₁}, *S₂* = {s₂, t₂}, *AP* ∪ *Nom* = {}, and transitions and labelling as in Figure 5. The algorithm nondeterministically computes *Yes* to be empty: add (*t₁, s₂*) to *No* as (*t₁, β, s₁*) ∈ *R₁* and there is no (*s₂, β, x*) ∈ *R₂*; add (*u₁, s₂*) to *No* as (*s₂, β, t₂*) ∈ *R₂* and there is no (*u₁, β, x*) ∈ *R₁*; add (*u₁, t₂*) since (*t₂, β, s₂*) ∈ *R₂* and the only match (*u₁, β, t₂*) is such that (*t₁, s₂*) is already in the set *No*; add (*s₁, s₂*) as (*s₁, α, u₁*) ∈ *R₁* but the only match (*s₂, α, t₂*) ∈ *R₂* is such that (*u₁, t₂*) is in *No*; add (*s₁, t₂*) since (*t₂, β, t₂*) ∈ *R₂* and there is no (*s₁, β, x*) ∈ *R₁*; and finally add (*t₁, t₂*) as (*t₁, β, s₁*) ∈ *R₁* but (*s₁, x*) is in *No* for all choices of *x*. 
2. Figure 6 shows $V = \{(M_1, s_1), (M_2, s_2)\}$ with $Nom = \{n\}$ and $W_V = \{(s_1, s_2), (u_1, t_2), (t_1, s_2)\}$; all other pairs are ruled out as they violate clause (a) of Definition 6.2 for $n$.

Fig. 5. A $V = \{(M_1, s_1), (M_2, s_2)\}$ for which no $V_i$ has a common refinement.

Fig. 6. A $V = \{(M_1, s_1), (M_2, s_2)\}$ for which there are common refinements.

6 Summary views

We construct 3-valued “summary” views $V_-$ and $V_+$ that serve as consistent and informative common abstractions and refinements of $V$ (respectively).

**Definition 7.** Let $V$ be given. For $* \in \{-, +\}$ we define a 3-valued view $V_* = (V_*^a, V_*^c)$ with $V_*^m = (W_V, R_{V_*}^m, L_{V_*}^m)$ for $m \in \{a, c\}$ such that for all $q \in AP \cup Nom$, $\alpha \in Act \cup \overline{Act}$

\[
\begin{align*}
    t \in L_{V_*}^a(q) & \iff \exists i: t_i \in L^a(q) \quad t \in L_{V_*}^c(q) & \iff \forall i: t_i \in L^c(q) \\
    (t, \alpha, t') \in R_{V_*}^a & \iff \exists i: (t_i, \alpha, t'_i) \in R^a \quad (t, \alpha, t') \in R_{V_*}^c & \iff \forall i: (t_i, \alpha, t'_i) \in R^c \\
    t \in L_{V_*}^a(q) & \iff \forall i: t_i \in L^a(q) \quad t \in L_{V_*}^c(q) & \iff \exists i: t_i \in L^c(q) \\
    (t, \alpha, t') \in R_{V_*}^a & \iff \forall i: (t_i, \alpha, t'_i) \in R^a \quad (t, \alpha, t') \in R_{V_*}^c & \iff \exists i: (t_i, \alpha, t'_i) \in R^c
\end{align*}
\]
except for \( n \in \text{Nom} \): \( L_{V_+}^n(n) = \{ \} \) if its definition above does not render a singleton; and \( L_{V_-}^n(n) = L_{V_+}^n(n) \) if the latter’s definition above renders a singleton.

These summary views render common refinements and abstractions.

**Proposition 1.** Let \( V = \{ M_i \mid 1 \leq i \leq k \} \) be given. Then \( V_- \) and \( V_+ \) are \( 2 \)-valued views and, for all \( t \in W_V \) and all \( i \), we have \( (V_-, t) \prec (M_i, t_i) \prec (V_+, t) \).

The abstraction \( V_- \) is concrete enough to meaningfully relate to the common views. The common refinement \( V_+ \) aids comprehension. Users may want to explore \( W_V \) to generate alternative summaries, as worked out for \( S_V \) in [35]. Unfortunately, these summary nodes lose too much precision to be of reliable use in model checking. A full version of this paper will approximate the decision problems \( S(V, \phi) \) and \( V(V, \phi) \) with views that mix static and dynamic aspects.

**Example 8.** Figure 7 shows \( V_+ \) and \( V_- \) for the \( V \) of Figure 6.

**Fig. 7.** The summary views \( V_+ \) and \( V_- \) for the \( V \) of Figure 6.

7 Related work

Uchitel & Chechik [35] merge a variant of modal transition systems with overlapping but different sets of events to obtain a minimal common refinement and suggest user participation to explore common behavior if no minimal common refinement exists. Their algorithms check the consistency of two models and construct a least common refinement, provided it exists. Their models are more general in that events may differ in views, but less general than ours in that we handle hybrid constraints and compute the space of all consistent tuples. They stress engineering activities in model elaboration, we use static analysis and identify the complexities of the relevant decision problems. Larsen et al. use projective views for a constrained-based proof methodology on modal transition systems [26]. 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 [11]. 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 [8]. Multiple-valued model checking is reducible to 2-valued model checking [8, 7]. Bruns & Godefroid [6] build a query checker for temporal logic which, for a Kripke structure and a query with a hole as input, returns a formula of propositional logic that, when placed into the query's hole, makes the query true for that Kripke structure. Our models for hybrid views assume that views have the same representation, here algebraic signature and type, of a specification. As Jackson points out, this may not always be appropriate [23]. Nentwich et al. developed the tool xlinkit that analyzes distributed XML documents for possible inconsistencies, based on rules written in first-order logic [29]. Guerra [16] 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 [16] specifications are written in linear-time temporal logic [32] and a non-monotonic semantics for this logic is defined based on default institutions [15], where the semantics of defaults is given by a generalized distance between interpretations. Foundations for view-based model checking, where models are those of first-order logic with transitive closure, are developed in [21]. In [22] assertion-consistency lattices are defined and argued to be the proper generalization of De Morgan lattices for sound abstraction of multiple-valued models and their checks. For modal transition systems and the modal mu-calculus, the decision problems of this paper have already been defined in [19] and the reduction to satisfiability in the modal mu-calculus for common refinement checks has been stated in [18].

8 Future work and leverage

The most pressing research issue from the foundational point of view is the development of approximations to the satisfiability problem $S(\mathcal{V}, \phi)$ using models that are more precise than the summary nodes presented in this paper. This may require a notion of view that has static and dynamic aspects.

The foundations for model checking individual 3-valued models are firmly established, but no tool support for 3-valued abstract-and-refine methodologies seem to exist to date. One important part of future work should therefore focus on building tools for software verification that make use of 3-valued abstract-and-refine model checking [13], for example a 3-valued version of the model checker SLAM [2, 3].

The foundations for model checking multiple 3-valued views are currently emerging and so it is perhaps too early to talk about leverage in earnest. Foundations and proof-of-concept implementations for *multiple-valued* model checking have been developed [8] but the connection of this work to reasoning about collective individual views is not well understood. Proof-of-concept implementations for specifying and reasoning about multiple models collectively are desired.
Although the views proposed in this paper may form a broad and solid basis for applications, one requires dedicated tools and customized property languages that reflect the constraints and desires of an application domain, be it databases, behavioral interaction, etc. Future work of the second author may aim at building such a tool for modal transition systems\cite{27} as views.

Empirical issues in building such explorative tools abound and should drive the development of more advanced versions. For example, in which application domains are guided simulations in putative common refinements more valuable than model checking? What are user-friendly ways of representing diagnostics, which are essentially winning strategies of 2-person games and often not representable as single traces?

9 Conclusions

We studied finite sets of views $\mathcal{V}$ and their common refinements, where each view is a 3-valued hybrid model. Such views are suitable models for answers to database queries, knowledge representation, functional requirements etc. We showed that the decision problems “Is there is common refinement of $\mathcal{V}$ satisfying $\phi$?” (satisfiability) and “Does $\phi$ hold for all common refinements of $\mathcal{V}$?” (validity) are EXPTIME-complete for the hybrid mu-calculus. We gave a PTIME decision procedure for checking whether such a set $\mathcal{V}$ is consistent in that it has a common refinement (satisfiability for $\phi$ being $\text{true}$). This procedure was used to compute the state space of “summary” views that are informative and consistent common refinements (respectively, abstractions). We then stated what future work is required to ensure the leverage of these results to applications.

Acknowledgments

Sebastian Uditel was always ready to give his feedback on this line of work. We acknowledge discussions with Glenn Bruns on the static analysis of sets of answers drawn from a set of queries executed on a database view.

References


