Multi-Valued Abstraction Using Lattice Operations

Stefan Vijzelaar and Wan Fokkink
VU University Amsterdam, The Netherlands
{s.j.j.vijzelaar,w.j.fokkink}@vu.nl

***Abstract*—In model checking, abstractions can cause spurious results, which need to be verified in the concrete system to gain conclusive results. Verification based on a multi-valued logic can distinguish between conclusive and inconclusive results, provides increased precision, and allows for encoding additional information into the model, which gives rise to new applications. To ensure a correct abstraction, one can use a mixed simulation [1] to relate a multi-valued model to its abstraction. In this paper we extend the notion of mixed simulation to include inconsistent values, thereby resolving an asymmetry in the definition and allowing for abstractions with increased precision when inconsistent values are available.**

I. Introduction

The combinatorial blow-up of behaviour when modelling complex systems, i.e. the state space explosion problem, is a central theme in the verification of systems. A solution is to use abstractions [2], but this can introduce spurious results [3]: behaviour in the abstract system that is not present in the concrete system. To detect such spurious results one can combine both over- and under-abstraction: it is guaranteed that behaviour present in both abstractions is also present in the concrete system. Given a suitable abstraction, it is then possible to conclusively verify properties over a much smaller state space than that of the concrete system.

One way to model both over- and under-abstraction is to use a multi-valued logic based on a lattice. For example, a three-valued logic ([4], [5]) can be used to introduce an additional truth value indicating inconclusive results, by using the value *unknown* in addition to the classical *true* and *false*. The classical Boolean operations are then redefined over the lattice formed by these truth values and a suitable truth ordering. In general, any lattice can be used as long as its operators, including negation, generally behave as a Boolean logic.

In addition to the truth ordering, it is possible to define an orthogonal information ordering. The result is a so-called bilattice [6], of which the four-valued Belnap logic [7] is a well-known example. Verification techniques using such four-valued logics have been successful in e.g. verifying logical circuits [8] and software [9]. The information ordering allows modelling of incomplete and conflicting information, which turns out to be a natural way to create abstractions. Bilattices have useful properties, which stem directly from their orthogonal truth and information orderings. Lattice operations over the truth ordering can be used for verifying properties, while lattice operations over the information ordering can be used for abstracting models. The truth ordering helps to redefine the classical Boolean operators, simplifying the reuse of temporal logics in a multi-valued setting ([10], [11]). The information ordering makes it possible to model incomplete and conflicting information and thereby detect inconclusive results. Despite using a multi-valued logic, temporal properties can still be verified using classical model checkers through decomposition [12]. Modelling on the basis of a bilattice creates a useful framework for abstracting and verifying systems.

The focus of this paper is on the correctness of abstraction in a multi-valued setting, specifically when evaluating µ-calculus properties over multi-valued Kripke models. Properties need to properly carry over from the concrete to the abstract model: they should evaluate to the same or a less conclusive value. In other words, we want the value in the abstract model to be equal or lower in the information ordering. We formalise this requirement by expanding on the notion of mixed simulation from [1]. It relates states in the abstract and the concrete model such that if two states are related then one is an abstraction of the other. If a mixed simulation relation exists, it guarantees that evaluating a temporal property at an abstract state gives an equal or less informative answer than evaluating it at the related concrete state.

We present a hierarchy of mixed simulations, each with increasingly weaker abstraction requirements, such that smaller and more precise abstractions of multi-valued Kripke models become possible. Evaluating µ-calculus modalities over a Kripke model involves multisets of truth values. To this end we introduce an accompanying hierarchy of theories on the abstraction of multisets and use it to prove correct abstraction for the mixed simulations presented in this paper. The hierarchy of mixed simulations contains bijective, basic, symmetric, and extended mixed simulation. We will show that the mixed simulation from [1] falls between basic and symmetric mixed simulation in strength of its abstraction requirements. We further weaken the requirements to create an extended mixed simulation, which allows for the increased precision of the abstraction technique described in [13].

Mixed simulation in [1] does not take into account inconsistent values, i.e. values indicating a contradiction. Only consistent partial distributive bilattices are considered, which caused an unnecessary asymmetry in the definition. Mixed simulation applies equally well to partial distributive bilattices that contain inconsistent values. In that setting the asymmetry might cause larger than needed abstractions. We resolve this asymmetry by weakening the definition of mixed simulation.

If additionally the distributive bilattice is required to be complete instead of partial, then we can use inconsistent values and lattice operations in the information order to further increase precision. It is shown in [13] how an abstraction using inconsistent values can be constructed for four-valued Belnap logic. Mixed simulation as defined in [1] however does not allow it. We extend the definition of mixed simulation to account for this construction.

Our motivation for improving the precision of multi-valued abstraction is an interest in using multi-valued lattices to model steerability. By steering an execution one can avoid bugs and reduce the number of states in the Kripke model that need to be explored during model checking. We present an example abstraction using a nine-valued logic where simultaneously steerability information is encoded in the Kripke model.

This paper is structured as follows. In section II we construct multi-valued logics based on a lattice of truth values, introduce bilattices to abstract truth values using the information ordering, and define Kripke models and µ-calculus in a multi-valued setting. In section III we describe abstraction for multisets of truth values, and in section IV apply these notions to multi-valued Kripke models to define increasingly precise abstraction techniques up to extended mixed simulation. Section V gives an example application of a multi-valued lattice, while section VI concludes the paper and refers to future work.

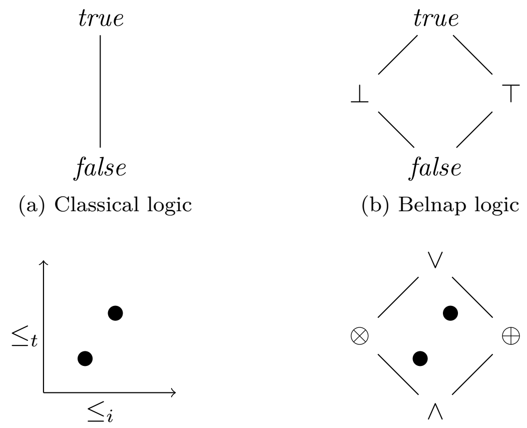
II. Preliminaries

A. Lattices and multi-valued logics

Models and temporal logics typically use a classical two-valued Boolean logic: transitions between states either exist (are *true*) or do not exist (are *false*); atomic propositions either hold for a state (are *true*) or do not hold (are *false*); and by extension temporal properties over a model can be verified (are *true*) or falsified (are *false*). It is customary to only draw *true* transitions in a state space graph, since missing transitions are assumed to be *false*.

Adding additional truth values to a model can increase its expressiveness and lead to more informative answers when verifying a property. One example is to distinguish between may and must transitions in an abstraction, and consequently being able to detect when a property has become inconclusive due to a loss of information. Multi-valued logics, which are logics with more than two truth values, can be defined using lattices.

A lattice $\mathcal{L} = \langle L, \sqsubseteq \rangle$ is a partially ordered ($\sqsubseteq$) set of elements (L), in which any two elements have a least upper bound (supremum, join or $\sqcup$) and a greatest lower bound (infimum, meet or $\sqcap$). A lattice has a join and meet for each non-empty finite subset of elements. Therefore, a non-empty finite lattice is bounded, and has a least element (bottom or $\bot$) and greatest element (top or $\top$).



(a) Classical logic

(b) Belnap logic

(c) Orderings

(d) Operations

Figure 1: Bilattices

A classical two-valued Boolean logic can be described as a lattice consisting of only two elements, with *false* being the infimum and *true* being the supremum; see Fig. 1a. The Boolean conjunction ($\wedge$) and disjunction ($\vee$) operations map respectively to the meet ($\sqcap$) and join ($\sqcup$) of the lattice.

The multi-valued logics we are interested in are quasi-Boolean logics, also called De Morgan logics. Without the complementation requirements of excluded middle ($x \vee \neg x = \mathit{true}$) and noncontradiction ($x \wedge \neg x = \mathit{false}$), they are a generalisation of Boolean logics. In contrast to Boolean logics, for the bounded distributive lattice of a quasi-Boolean logic it is not necessary that each element x has a complement y such that $x \sqcup y = \top$ and $x \sqcap y = \bot$.

Since lattices are bounded and distributive, there is a least element (*false*) and a greatest element (*true*), and meet and join operations distribute over each other. Similar to classical Boolean logic, operations can be defined using lattice operations: conjunction ($\wedge$) and disjunction ($\vee$) map to meet ($\sqcap$) and join ($\sqcup$), while negation ($\neg$) maps to an appropriate involution ($\sim$). In addition to being its own inverse, this involution should adhere to De Morgan's laws. It follows that disjunction and conjunction are distributive, and the law of double negation applies.

B. Bilattices and abstraction

In a bilattice [6] there are two orthogonal orderings over the same set of elements. When used in the context of logics and abstractions, one is generally called the truth ordering and the other the information ordering. The truth ordering, together with a suitable definition for negation, can be used to define a quasi-Boolean logic. The information ordering, together with some correctness criterium, can be used to model abstraction. To distinguish between operations of the two orderings, we use $\wedge$ or $\vee$ to indicate a meet or join over the truth ordering and $\otimes$ or $\oplus$ to indicate a meet or join over the information order.

A bilattice can be used to model both under- and over-abstraction at the same time. The truth ordering of the bilattice used for this purpose in [13] defines a four-valued logic as described by Belnap [7] (Fig. 1b). The additional two truth values $\bot$ and $\top$ should not be interpreted as the top and bottom of the logic: they are used to model the top and bottom of the information ordering. One could interpret $\bot$ as neither true nor false, and $\top$ as both true and false. In other words, the elements of the information ordering can be seen as sets, which can contain an item for truth (t) and an item for falsity (f). Truth values no longer map to a single items of the set $\{t, f\}$, as is the case for classical Boolean logic, but to its subsets. This allows for values which contain none ($\bot$) or both ($\top$) of the elements in $\{t, f\}$. Abstractions can use these additional values to model a loss of information ($\bot$) and increase precision ($\top$).

The examples of (bi)lattices in Fig. 1a and Fig. 1b are presented using Hasse diagrams in which only the transitive reduction of the partial order is represented by lines between elements. An element higher up in the figure is larger in the truth ordering with respect to connected elements further down. Similarly an element more to the right in the figure is larger in the information ordering with respect to connected elements further to the left.

We will extend this convention to depict bilattices in general, without going into the specific structure of the bilattice, by not drawing the entire transitive reduction of its partial orderings. Elements of the lattice are points in two-dimensional space, with each dimension corresponding to an ordering (Fig. 1c). The partial ordering with respect to a specific element is depicted by a cone of bounding diagonal lines, while lattice operations use a diamond shape which should be interpreted as lattice operations on the set of elements within (Fig. 1d).

A bilattice is distributive if the meet and join operators of both its orderings are distributive with respect to each other. This leads to twelve distributive laws: one for each combination of the four operators. Every distributive bilattice is also interlaced: the meet and join operators are monotonous with respect to both orderings, which means that increasing the value of the operands in an ordering never decreases the value of the result in that ordering. This holds even when using meet or join operators of the orthogonal ordering; for example, when increasing the information of operands in a meet operation of the truth ordering, the result will be equal or higher in the information ordering. The exception is the negation operator, which is not monotonous for the order it is defined over, but is monotonous with respect to the orthogonal ordering.

C. Multi-valued Kripke models and µ-calculus

Multi-valued Kripke models are a generalisation of Kripke models and can use values of any quasi-Boolean logic for transitions and atomic propositions, instead of being limited to the usual two Boolean values *true* and *false*. Similarly a µ-calculus formula is evaluated by using the operators as defined by the quasi-Boolean logic. We follow the definitions presented in [1].

Definition 1. A multi-valued Kripke model is a tuple $M = \langle \mathcal{L}, AP, S, s_0, R, \Theta \rangle$, where $\mathcal{L} = \langle L, \leq, \neg \rangle$ is a quasi-Boolean algebra, AP a set of atomic propositions, S a finite set of states, s_0 the initial state, $R : S \times S \to L$ a transition relation mapping to truth values of $\mathcal{L}$, and $\Theta : AP \to (S \to L)$ a labelling function assigning truth values to states for each atomic proposition.

Definition 2. Given a set of atomic propositions AP and a set of propositional variables Var. The µ-calculus in negation normal form is defined as follows:

$$\phi ::= p \mid \neg p \mid Z \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \Diamond \phi \mid \Box \phi \mid \mu Z.\psi \mid \nu Z.\psi$$

With $p \in AP$, and $Z \in \mathit{Var}$. In the above $\mu Z.\psi$ should be interpreted as the least fixed point (lfp) and $\nu Z.\psi$ as the greatest fixed point (gfp) of ψ with respect to Z.

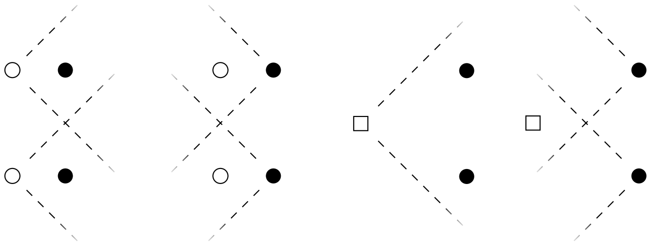
Definition 3. The semantics $\|\phi\|^M_{\mathcal{V}}$ of a µ-calculus formula ϕ for a multi-valued Kripke model M, with respect to an environment $\mathcal{V} : \mathit{Var} \to (S \to L)$ mapping variables to a multi-valued set of states, is defined as follows:

$$\begin{aligned}
\|p\|^M_{\mathcal{V}} &= \Theta(p) \\
\|\neg p\|^M_{\mathcal{V}} &= \neg\Theta(p) \\
\|\phi_1 \vee \phi_2\|^M_{\mathcal{V}} &= \|\phi_1\|^M_{\mathcal{V}} \vee \|\phi_2\|^M_{\mathcal{V}} \\
\|\phi_1 \wedge \phi_2\|^M_{\mathcal{V}} &= \|\phi_1\|^M_{\mathcal{V}} \wedge \|\phi_2\|^M_{\mathcal{V}} \\
\|\Diamond\phi\|^M_{\mathcal{V}} &= \{s \in S \mid \textstyle\bigvee_{t \in S}(R(s,t) \wedge \|\phi\|^M_{\mathcal{V}}(t))\} \\
\|\Box\phi\|^M_{\mathcal{V}} &= \{s \in S \mid \textstyle\bigwedge_{t \in S}(\neg R(s,t) \vee \|\phi\|^M_{\mathcal{V}}(t))\} \\
\|\mu Z.\phi\|^M_{\mathcal{V}} &= \mathrm{lfp}(\|\phi\|^M_{\mathcal{V}}, Z) \\
\|\nu Z.\phi\|^M_{\mathcal{V}} &= \mathrm{gfp}(\|\phi\|^M_{\mathcal{V}}, Z) \\
\|Z\|^M_{\mathcal{V}} &= \mathcal{V}(Z)
\end{aligned}$$

The strong relation between Boolean operations and lattice operations ensures that the verification of temporal properties remains the same for different quasi-Boolean logics. Classical definitions of temporal properties can easily be translated to lattice operations, and are then applicable to the more general class of quasi-Boolean logics instead of just classical Boolean logic.

III. Abstracting multisets

To ensure the correctness of an abstraction technique, we need to define a correctness criterium and show that the technique preserves this criterium. Our goal is to show, given two multi-valued Kripke models M_1 and M_2 with M_2 a correct abstraction of M_1, that evaluating a µ-calculus property ϕ over M_1 is always equally or more informative than evaluating ϕ over M_2. In other words, a property ϕ evaluated at the initial state s^0_2 of M_2 should be equal or less informative than ϕ evaluated at the initial state s^0_1 of



(a) Concrete (b) Abstract (c) Concrete (d) Abstract

Figure 2: Abstracting truth values

M_1: $\|\phi\|^{M_2}(s_2^0) \leq_i \|\phi\|^{M_1}(s_1^0)$. We want to define this for any quasi-Boolean logic based on a distributive bilattice.

To simplify the problem we start with abstracting multisets of truth values; these results can be applied when abstracting multi-valued Kripke models, since multisets of truth values occur when evaluating µ-calculus formulas. Remember that µ-calculus formulas are evaluated using lattice operations: meet and join of the truth ordering are used to build state properties out of atomic propositions, combine state properties with transition values, and quantify over different paths. Irrespective of what a multiset of truth values represents, we can define rules for correctly abstracting these values, such that a lattice operation over the multiset will result in an equally or less informative answer.

A. Bijection between multisets

Let T_1 and T_2 be multisets of truth values for which we want to ensure that T_2 abstracts T_1. One way to ensure a correct abstraction is to provide a bijection $B \subseteq T_1 \times T_2$ which pairs the elements of T_1 and T_2 such that $t_2 \leq_i t_1$ for each pair of t_1 and t_2 in B. Then monotonicity will ensure that whatever distributive lattice operation we evaluate over the multisets, the answer over T_2 will be equally or less informative than the answer over T_1.

Proposition 4. *Given a join or meet operation* $\textcircled{?}$ *of a distributive bilattice* $\mathcal{B}$ *over elements* L*, with (inverted) ordering* $\leq$*. For two multisets* Φ *and* Ψ *with elements from* L*, the formula* $\textcircled{?}\Phi \leq \textcircled{?}\Psi$ *holds if there exists a bijection* $B \subseteq \Phi \times \Psi$ *such that* $\forall(\phi, \psi) \in B : \phi \leq \psi$.

In this paper only finite multisets of truth values are considered, because we only abstract Kripke models with a finite number of states. Binary meet and join operators like $\textcircled{?}$ can therefore be applied to a multiset by using a left associative application over all elements.

In Fig. 2a we see how for each concrete value (●) of T_1 there is a unique abstract value (○) of T_2 which is less or equally informative: each concrete value is in the information cone of a corresponding abstract value, indicating it is equally or more informative than that value. To complete the bijection, we also require the opposite to hold as depicted in Fig. 2b: for each abstract value (○) of T_2 there is a unique concrete value (●) of T_1 which is equally or more informative.

B. Multisets of unequal size

We weaken the requirement for abstraction by allowing an unequal number of values in sets T_1 and T_2; this removes the uniqueness requirement of bijective pairs. It is sufficient if for each value t_1 in T_1 there exists a value t_2 in T_2 such that $t_2 \leq_i t_1$, and for each value t_2 in T_2 there exists a value t_1 in T_1 such that $t_2 \leq_i t_1$. We can use the idempotence of the lattice operation to freely duplicate elements already present in T_1 or T_2 and create the required bijection.

Proposition 5. *Given a join or meet operation* $\textcircled{?}$ *of a distributive bilattice* $\mathcal{B}$ *over elements* L*, with (inverted) ordering* $\leq$*. For two multisets* Φ *and* Ψ *with elements from* L*, the formula* $\textcircled{?}\Phi \leq \textcircled{?}\Psi$ *holds if both:*

1) $\forall \phi \in \Phi : \exists \psi \in \Psi : \phi \leq \psi$
2) $\forall \psi \in \Psi : \exists \phi \in \Phi : \phi \leq \psi$

In Fig. 2c we see for each concrete value (●) of T_2 that there is an abstract value (□) of T_1 which is equally or more informative, but not unique. The opposite is no longer symmetric, as shown in Fig. 2d: for the abstract value (□) of T_1 there are two concrete values (●) of T_2 which are less or equally informative: the value is in the information cone of both concrete values.

Prop. 5 can be used to check whether a set T_2 abstracts a set T_1 of truth values (i.e. T_1 refines T_2) when both T_1 and T_2 are given. It is also useful when creating an abstraction (or refinement) from scratch when only the set T_1 (or T_2) is given. We can visualise creating an abstraction T_2 by using the information cones of concrete values in T_1, as shown in Fig. 2b and Fig. 2d: each cone should contain at least one abstract value (either unique: ○, or not: □) and no abstract values can occur outside of cones. This holds dually for creating a refinement T_1.

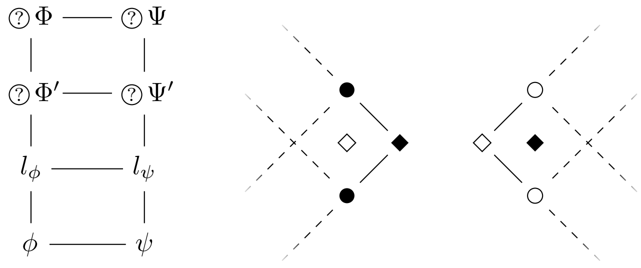
C. Unit elements

We can further weaken the requirements for abstraction if there is a unit element for the operation. In addition to duplicating elements using the idempotence of the lattice operation, we can then freely add the unit element to the multiset without influencing the result.

Assume the operation over the multiset is a disjunction. If we can show for a value t_1 in T_1 that *false* $\leq_i t_1$, then we no longer require a value t_2 in T_2 such that $t_2 \leq_i t_1$. Similarly for those values t_2 in T_2 with $t_2 \leq_i$ *false*.

Proposition 6. *Given a join or meet operation* $\textcircled{?}$ *of a distributive bilattice* $\mathcal{B}$ *over elements* L*, with (inverted) ordering* $\leq$*, and a unit element* $u_?$ *of* $\textcircled{?}$*. For two multisets* Φ *and* Ψ *with elements from* L*, the formula* $\textcircled{?}\Phi \leq \textcircled{?}\Psi$ *holds if both:*

1) $\forall \phi \in \Phi : \phi \leq u_? \vee \exists \psi \in \Psi : \phi \leq \psi$
2) $\forall \psi \in \Psi : u_? \leq \psi \vee \exists \phi \in \Phi : \phi \leq \psi$



(a) Strategies (b) Concrete (c) Abstract

Figure 3: Adding truth values

This proposition is useful if we restrict ourselves to a single lattice operation over the multiset. It allows for smaller abstractions by ignoring values in the set which have no influence on the final result. As we will show later, it is also applicable when evaluating µ-calculus formulas over multi-valued Kripke models, since transition values are effectively combined using only a single operator.

D. Bilattice orderings

The requirements can be further weakened if we take into account the associated orderings of lattice operations. An operation $\textcircled{?}$ over the multiset can be any of the meet and join operations of the bilattice. Since a meet operation can be changed into a join operation by inverting the associated ordering, we can assume without loss of generality that $\textcircled{?}$ is a join operation with respect to an ordering $\leq_?$. We will use the definition of this join operation to weaken the abstraction requirements.

We can add a new truth value l to a multiset T of truth values, as long as it does not influence the result of $\textcircled{?}T$. Since $\textcircled{?}$ calculates the supremum, any value $l \leq_? \textcircled{?}T$ can be added to T. In addition this allows us to change an existing value t in T to a value l, as long as $t \leq_? l \leq_? \textcircled{?}T$, by adding l and calculating $t \,\textcircled{?}\, l$. Replacing t for l in this manner can help to form a bijection.

The new possibilities for abstraction requirements are summarised in Fig. 3a. We explicitly allow for using subsets $\Phi' \subseteq \Phi$ and $\Psi' \subseteq \Psi$, which will prove useful when an abstraction requires us to combine specific subsets of the multiset. A correct abstraction requires direct paths from each $\phi \in \Phi$ to $\textcircled{?}\,\Psi$ and each $\psi \in \Psi$ to $\textcircled{?}\,\Phi$ to form the required bijection. This works because each value ϕ or ψ can be transformed into a value higher in the same column of the diagram, while we can add any value lower than $\textcircled{?}\,\Phi$ or $\textcircled{?}\,\Psi$ to respectively Φ or Ψ. The actual bijection is formed by taking a horizontal step.

For example, we can satisfy the requirements for a value ϕ by transforming it to l_ϕ, pairing it with an l_ψ, and adding this l_ψ to Ψ. This is possible when $\phi \leq_? l_\phi \leq_? \textcircled{?}\,\Phi' \leq_? \textcircled{?}\,\Phi$ and $l_\psi \leq_? \textcircled{?}\,\Psi' \leq_? \textcircled{?}\,\Psi$. Note that we can omit $\psi \leq_? l_\psi$ for this direction of the requirement. Another possibility would be to immediately pair ϕ with a value $\psi \in \Psi$; this corresponds to one of the requirements of Prop. 5.

The diagram can be further simplified. First, because $\textcircled{?}\,\Phi' \leq_? \textcircled{?}\,\Phi$ follows directly from $\Phi' \subseteq \Phi$ and dually for Ψ, we can remove the top row of the diagram. Second, we can choose l_ϕ equal to any ϕ or $\textcircled{?}\,\Phi'$, and l_ψ equal to any ψ or $\textcircled{?}\,\Psi'$; therefore, any requirement pairing ϕ with ψ or $\textcircled{?}\,\Phi'$ with $\textcircled{?}\,\Psi'$ can be described by a pairing of l_ϕ and l_ψ . Finally, given l_ϕ and l_ψ for proving ϕ or ψ, we can always find a new l'_ϕ and l'_ψ with $l'_\phi = \phi$ or $l'_\psi = \psi$, such that we need only one value l to create a proof.

Proposition 7. *Given a join or meet operation* $\textcircled{?}$ *of a distributive bilattice* $\mathcal{B}$ *over elements* L*, with (inverted) orderings* $\leq$ *and* $\leq_?$*, such that* $\textcircled{?}$ *calculates the supremum with respect to* $\leq_?$*. For two multisets* Φ *and* Ψ *with elements from* L*, the formula* $\textcircled{?}\,\Phi \leq \textcircled{?}\,\Psi$ *holds if both:*

1) $\forall \phi \in \Phi : \exists l \in L : \exists \Psi' \subseteq \Psi : \phi \leq l \land l \leq_? \textcircled{?}\,\Psi'$
2) $\forall \psi \in \Psi : \exists l \in L : \exists \Phi' \subseteq \Phi : l \leq \psi \land l \leq_? \textcircled{?}\,\Phi'$

When abstracting truth values we considered the possibility of having a single abstract truth value correspond to multiple concrete truth values. Assume that we want to abstract a multiset of concrete truth values (●) using a single abstract value; if we follow the requirements of item 1 of Prop. 5, this means the abstract truth value (□) has to be equal or less informative than all concrete truth values in the set (see Fig. 2d). Depending on the specific values and operations involved, this requirement can cause a significant loss of information.

However, using Prop. 7 we can increase precision by adding an additional concrete value (◆) to the multiset of concrete values (●) and only requiring the abstract truth value to be equal or less informative than this new value. If we allow both disjunction and conjunction over the multiset, as is the case in this example, then the additional concrete value (◆) needs to be both equal or smaller in the truth ordering than the conjunction, and equal or larger in the truth ordering than the disjunction. The result is shown in Fig. 3b and dually for abstract values in Fig. 3c.

IV. Abstracting multi-valued Kripke models

In the previous section we have looked at abstracting multisets of truth values such that lattice operations on these values lead to an equally or less informative result. When abstracting multi-valued Kripke models we also work with multisets of truth values, but it becomes important to consider what these values represent: since we restrict ourselves to evaluating µ-calculus properties, we only have to consider the operations on truth values which correspond to µ-calculus operators. This allows us to lift the notion of correct abstraction from multisets of truth values in the context of lattice operations, to multi-valued Kripke models in the context of µ-calculus properties.

In µ-calculus, atomic propositions and Boolean operators can be used to construct state properties. As the name suggests, state properties are specific to the state they are evaluated in, since each state has its own assignment of truth values for these atomic propositions. To abstract these atomic propositions, we have to take into account that they might be used in any number of possible µ-calculus state properties. The only way to ensure that the abstraction is correct for all µ-calculus properties is to use monotonicity: every atomic proposition in a state of the abstract system is equally or less informative than in its paired state of the concrete system. This ensures that any state property derived from these propositions is also equally or less informative.

We are left with considering the role of transitions in the abstraction of multi-valued Kripke models, which in µ-calculus are used when evaluating the modal operators ($\Box$ and $\Diamond$). Computing the result of modal operators in a source state s consists of two parts: for each target state t the value of the transition from s to t is combined with the value of a µ-calculus property at t; these intermediate results for each target state are then combined to get the final answer. We can consider the intermediate results as a multiset of truth values to ensure a correct abstraction.

More specifically, the previous paragraph entails that for each source state s we have a set of intermediate results which can be used to calculate a µ-calculus modality. We do not specify how these truth values are exactly constructed, which differs depending on the modality used, but we do want to ensure they are correctly abstracted. For this purpose we can use the abstraction criterium as previously described for multisets of truth values.

A. Approximations

During abstraction, we cannot calculate the intermediate truth values of a Kripke model without doing actual model checking, which is something we want to defer until after the abstraction. The intermediate values in a Kripke model for a source state s are combinations of the transition value to a target state t, and a µ-calculus property evaluated at t; since during abstraction we do not know which properties will be evaluated at state s, we also do not know the value of the µ-calculus property at t or how it should be combined with the transition value from s to t.

Directly comparing intermediate values is therefore not possible; we can however push down the comparison to the transition values and atomic propositions of the underlying Kripke model, as will be shown by the different types of mixed simulation in the following sections. For example, instead of comparing $\phi_1 \vee \psi_1 \leq_i \phi_2 \vee \psi_2$ we can push down the comparison and determine whether both $\phi_1 \leq_i \phi_2$ and $\psi_1 \leq_i \psi_2$, which is sufficient due to monotonicity of $\leq_i$. If done correctly for Kripke models, it will not matter which property is evaluated: the comparison will hold and the intermediate values will form a correct abstraction.

Some of our abstraction requirements use lattice operation and need to compare the result of meet and join operations over intermediate values with other intermediate values. For example, we might want to know the most informative abstract intermediate value which is equally or less informative then a disjunction of concrete intermediate values. We will need a way to push down lattice operations, such that the result of an operation will again take the form of an intermediate value and can be compared as usual.

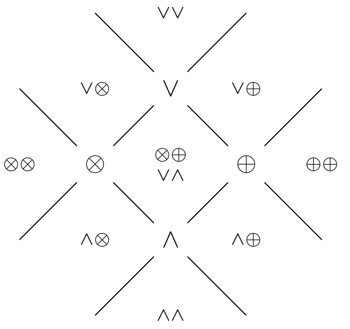


Figure 4: Approximations

Pushing down lattice operations comes at a cost of precision. The result will be an approximation of the actual value and we need to take into account that this imprecision can compromise the abstraction. We solve this by ensuring that the imprecision is in accordance with the direction of the abstraction. For example, if we are approximating a lattice operation over concrete intermediate values, then it is no problem if the approximation becomes less informative. Transitivity of the ordering ensures that any abstract intermediate value equally or less informative than the approximation will also be equally or less informative than the actual value.

To calculate an approximation we use the monotonicity properties of the meet and join operations in the bilattice. Particularly, if intermediate truth values t_1 and t_2 are themselves combinations of truth values ϕ and ψ, meaning that for a meet or join operation $\textcircled{1}$ we have $t_1 = \phi_1 \textcircled{1} \psi_1$ and $t_2 = \phi_2 \textcircled{1} \psi_2$, then we can calculate an approximation $t_3 = \phi_3 \textcircled{1} \psi_3$ of a meet $(t_1 \otimes t_2)$ or join $(t_1 \oplus t_2)$ by defining $\phi_3 = \phi_1 \oplus \phi_2$ and $\psi_3 = \psi_1 \otimes \psi_2$. Due to monotonicity and distributivity it is then guaranteed that $t_1 \otimes t_2 \leq_i t_3 \leq_i t_1 \oplus t_2$. Note that although switching the $\otimes$ and $\oplus$ operators when calculating ϕ_3 and ψ_3 may give a different value t_3, the inequality still holds.

Proposition 8. *The formula* $(\phi_1 \textcircled{?} \psi_1) \textcircled{1} (\phi_2 \textcircled{?} \psi_2) \leq (\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{2} \psi_2)$*, with* $\textcircled{?}$, $\textcircled{1}$, $\textcircled{2}$ *each an arbitrary join or meet operations of the distributive bilattice* $\mathcal{B}$*, and* $\leq$ *an (inverted) ordering of* $\mathcal{B}$*, holds if* $\psi_1 \leq (\psi_1 \textcircled{2} \psi_2)$ *and* $\psi_2 \leq (\psi_1 \textcircled{2} \psi_2)$*.*

Corollary 9. *Given three arbitrary join or meet operators* $\textcircled{?}$, $\textcircled{1}$, $\textcircled{2}$ *of the distributive bilattice* $\mathcal{B}$ *and two orderings* $\leq_1$, $\leq_2$ *such that* $\textcircled{i}$ *calculates the infimum with respect to* $\leq_i$ *for* $i \in \{1, 2\}$*. Then the following inequalities hold:*

1) $(\phi_1 \textcircled{?} \psi_1) \textcircled{1} (\phi_2 \textcircled{?} \psi_2) \leq_2 (\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{2} \psi_2)$
2) $(\phi_1 \textcircled{?} \psi_1) \textcircled{2} (\phi_2 \textcircled{?} \psi_2) \leq_1 (\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{2} \psi_2)$

An overview of all possible approximations is given in Fig. 4. The singletons $\vee$, $\oplus$, $\wedge$, and $\otimes$ indicate the results of directly combining t_1 and t_2 using a single

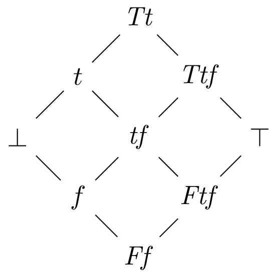


Figure 5: Nine-valued bilattice

operator, while the pairs of operations show the result t_3 of approximating an operation using two operators. We see that pushing down a $\oplus$ operator into two $\oplus$ operators will give an equally or more informative result. Using the $\wedge$ and $\otimes$ operators as an approximation will be less informative than $\oplus$, equally or less informative than $\wedge$, less truthful than $\vee$, and equally or less truthful than $\otimes$. Note that switching the order in pairs of operators can give a different approximation, but the relative position in the figure stays the same.

B. *Bijective mixed simulation*

A simple way to prove that one Kripke model is an abstraction of another is to give a bijection between the intermediate values of their initial states. This requires a recursive approach similar to bisimulation, except that we allow for a loss of information. We use the terminology of [1] and build on their definition of a mixed simulation.

Definition 10. Let $M_i = \left\langle \mathcal{B}, AP, S_i, s_i^0, R_i, \Theta_i \right\rangle$ for $i \in \{1,2\}$ be multi-valued Kripke models. $H \subseteq S_1 \times S_2$ is a bijective mixed simulation from M_1 to M_2 if H is a bijection and $(h_1, h_2) \in H$ implies:

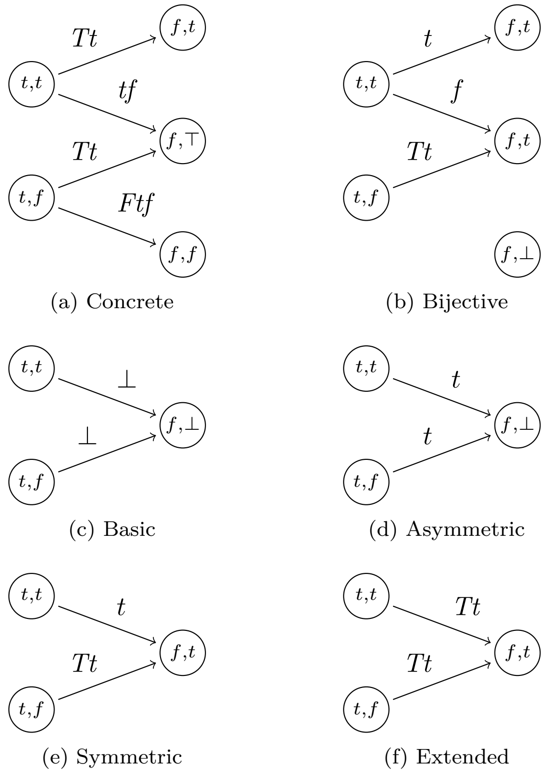
1) $\forall a \in AP : \Theta_2(a)(h_2) \leq_i \Theta_1(a)(h_1)$
2) $\forall t_1 \in S_1 : \forall t_2 \in S_2 : (t_1, t_2) \notin H \vee R_2(h_2, t_2) \leq_i R_1(h_1, t_1)$

If $(s_1^0, s_2^0) \in H$ for some bijective mixed simulation H, then M_2 abstracts M_1.

This definition ensures that all states in M_2 are paired with states in M_1 such that all transitions between states in M_1 are mirrored by less or equally informative transitions for their paired states in M_2, and all atomic propositions for states in M_1 are less or equally informative for their paired state in M_2. Together this ensures that there is a bijection between the intermediate values for all paired states, as required by Prop. 4.

Theorem 11. *Let $H \subseteq S_1 \times S_2$ be a bijective mixed simulation relation from M_1 to M_2, and let ϕ be a closed μ-calculus formula. Then for every $(h_1, h_2) \in H$, $\|\phi\|^{M_2}(h_2) \leq_i \|\phi\|^{M_1}(h_1)$.*

The following examples use the distributive bilattice shown in Fig. 5, of which the notation has been borrowed from [14]. The intuition behind these truth values will be explained in section V. Fig. 6b shows an example abstraction for the concrete model of Fig. 6a. Indicated are: the truth values for all transitions except those which are *false*, which for this specific bilattice corresponds to the value *Ff*; and the values of two propositions p and q, which are visible in the nodes. Each concrete state is paired with its abstract counterpart in the same position, forming a bijection. The only difference between the two models is a reduction in the information order of transition values and atomic propositions.



(a) Concrete

(b) Bijective

(c) Basic

(d) Asymmetric

(e) Symmetric

(f) Extended

Figure 6: Abstractions of a Kripke model

C. *Basic mixed simulation*

We refine Def. 10 by allowing a single abstract state to simulate multiple concrete states and vice-versa. The result is a stronger definition of mixed simulation than the one described in [1], since we do not exempt certain transition values as will be done in the upcoming variants of mixed simulation.

Definition 12. Let $M_i = \left\langle \mathcal{B}, AP, S_i, s_i^0, R_i, \Theta_i \right\rangle$ for $i \in \{1,2\}$ be multi-valued Kripke models. $H \subseteq S_1 \times S_2$ is a basic mixed simulation from M_1 to M_2 if $(h_1, h_2) \in H$ implies:

1) $\forall a \in AP : \Theta_2(a)(h_2) \leq_i \Theta_1(a)(h_1)$
2) $\forall t_1 \in S_1 : \exists t_2 \in S_2 : (t_1, t_2) \in H \wedge R_2(h_2, t_2) \leq_i R_1(h_1, t_1)$
3) $\forall t_2 \in S_2 : \exists t_1 \in S_1 : (t_1, t_2) \in H \wedge R_2(h_2, t_2) \leq_i R_1(h_1, t_1)$

If $(s_1^0, s_2^0) \in H$ for some basic mixed simulation H, then M_2 abstracts M_1.

We require transitions from h_1 to any concrete state t_1 to be mirrored by another less or equally informative transition from h_2 to some abstract t_2, and dually for h_2. By requiring (t_1, t_2) to be in H we ensure that the µ-calculus property is also less or equally informative: either because the atomic propositions are less or equally informative or by recursion of the definition.

Theorem 13. *Let $H \subseteq S_1 \times S_2$ be a basic mixed simulation relation from M_1 to M_2, and let ϕ be a closed μ-calculus formula. Then for every $(h_1, h_2) \in H$, $\|\phi\|^{M_2}(h_2) \leq_i \|\phi\|^{M_1}(h_1)$.*

Fig. 6c shows an abstraction of the concrete model using a basic mixed simulation. The three concrete states with p equalling f have been combined into one abstract state. Since the three concrete states do not agree on the value for q, we have to combine these values to ensure item 1 of Def. 12 holds: we get the most informative value by calculating $t \otimes \top \otimes f = \bot$. For the transitions we have $Tt \otimes tf \otimes Ff = \bot$ and $Ff \otimes Tt \otimes Ftf = \bot$.

D. Symmetric mixed simulation

Now that we have a basic definition for mixed simulation, we can weaken its definition to allow for smaller abstractions. One way to weaken the requirements of Def. 12 is to include the ability to add unit elements to either side of a comparison without influencing the result, as stated in Prop. 6. This means we can ignore an intermediate value if its transition value makes it equal to the unit element. It allows us to weaken items 2 and 3 of the definition.

Definition 14. Let $M_i = \langle \mathcal{B}, AP, S_i, s_i^0, R_i, \Theta_i \rangle$ for $i \in \{1, 2\}$ be multi-valued Kripke models. $H \subseteq S_1 \times S_2$ is a symmetric mixed simulation from M_1 to M_2 if $(h_1, h_2) \in H$ implies:

1) $\forall a \in AP : \Theta_2(a)(h_2) \leq_i \Theta_1(a)(h_1)$
2) $\forall t_1 \in S_1 : u_\vee \leq_i R_1(h_1, t_1) \vee \exists t_2 \in S_2 : (t_1, t_2) \in H \wedge R_2(h_2, t_2) \leq_i R_1(h_1, t_1)$
3) $\forall t_2 \in S_2 : R_2(h_2, t_2) \leq_i u_\vee \vee \exists t_1 \in S_1 : (t_1, t_2) \in H \wedge R_2(h_2, t_2) \leq_i R_1(h_1, t_1)$

If $(s_1^0, s_2^0) \in H$ for some symmetric mixed simulation H, then M_2 abstracts M_1.

These changes are possible because we limit ourselves to µ-calculus properties: for both modalities, a transition value $u_\vee$ ensures that the annihilator element used when calculating the intermediate value is also the unit element when combining intermediate results. For example, assume an intermediate value for a target state t consisting of the transition value *false* and a property ϕ. The intermediate value for the $\diamond$ modality will then be $\mathit{false} \wedge \phi = \mathit{false}$, which has no effect on the subsequent disjunction. The intermediate value for the $\square$ modality will be $\neg \mathit{false} \vee \phi = \mathit{true}$, which has no effect on the subsequent conjunction.

The definition of mixed simulation in [1] is identical to Def. 14, except that for item 2 the clause $\mathit{false} \leq_i R_1(h_1, t_1)$ is replaced with $\mathit{false} = R_1(h_1, t_1)$. We were motivated to correct this asymmetry because we want the definition to work better in the presence of inconsistent values. Note that a symmetric mixed simulation has weaker requirements than both basic mixed simulation and mixed simulation from [1].

Theorem 15. *Let $H \subseteq S_1 \times S_2$ be a symmetric mixed simulation relation from M_1 to M_2, and let ϕ be a closed μ-calculus formula. Then for every $(h_1, h_2) \in H$, $\|\phi\|^{M_2}(h_2) \leq_i \|\phi\|^{M_1}(h_1)$.*

We can calculate the transition values for the first transition of the symmetric mixed simulation (see Fig. 6e) by ignoring the *Ff* transition: $Tt \otimes tf = t$. For the second transition we can ignore both the *Ff* and *Ftf* transition, leaving only the value *Tt*. In addition the value of q in the combined state can be increased to $t \otimes \top = t$, because the lowest of the three rightmost concrete states has become reachable only by transitions larger in the information order than *false* and can therefore be ignored.

Would we have used the asymmetric definition of mixed simulation of [1], the result of which is shown in Fig. 6d, then we could not have ignored the *Ftf* transition. The value for q would have remained $\bot$ and the transitions would respectively be valued as $Tt \otimes tf = t$ and $Tt \otimes Ftf = t$. Note that symmetric mixed simulation is more informative than asymmetric mixed simulation, which is more informative than basic mixed simulation.

E. Extended mixed simulation

We use Prop. 8 to further weaken the symmetric mixed simulation of Def. 14, by allowing states in S_2 which are not related by the simulation to any singular state in S_1. Such states relate to a subset of states in S_1, which are then combined to form an additional intermediate value. Note that subsets are not represented by any actual state in the concrete Kripke model. Therefore their values need to be approximated when required for comparison.

Definition 16. Let $M_i = \langle \mathcal{B}, AP, S_i, s_i^0, R_i, \Theta_i \rangle$ for $i \in \{1, 2\}$ be multi-valued Kripke models. $H \subseteq \mathcal{P}(S_1) \times \mathcal{P}(S_2)$ is an extended mixed simulation from M_1 to M_2 if $(H_1, H_2) \in H$ implies:

1) $\forall a \in AP : \oplus_{h_2 \in H_2} \Theta_2(a)(h_2) \leq_i \otimes_{h_1 \in H_1} \Theta_1(a)(h_1)$
2) $\forall t_1 \in S_1 : \exists T_2 \subseteq S_2 : (\{t_1\}, T_2) \in H \wedge \oplus_{h_2 \in H_2}(\otimes_{t_2 \in T_2} R_2(h_2, t_2) \otimes u_\vee) \leq_i \otimes_{h_1 \in H_1} R_1(h_1, t_1)$
3) $\forall t_2 \in S_2 : \exists T_1 \subseteq S_1 : (T_1, \{t_2\}) \in H \wedge \oplus_{h_2 \in H_2} R_2(h_2, t_2) \leq_i \otimes_{h_1 \in H_1}(\oplus_{t_1 \in T_1} R_1(h_1, t_1) \oplus u_\vee)$

If $(\{s_0^1\}, \{s_0^2\}) \in H$ for some extended mixed simulation H, then M_2 abstracts M_1.

Prop. 7 allows for additional values $l \in L$ which are lower or higher in the truth ordering than a meet or join over a subset of intermediate values, depending on the operation we use. In µ-calculus properties both disjunction and conjunction over the intermediate values are possible. Therefore we need a value l which in the truth ordering is both smaller than the disjunction ($\vee$) and larger than the conjunction ($\wedge$) over a set of intermediate values.

In Fig. 1d we can see that the least informative value l for which this holds is calculated by taking the $\otimes$ of the same subset. The most informative value l for which this holds is calculated by taking the $\oplus$ operation. Both can be approximated using the combination of $\oplus$ and $\otimes$, but we have to be careful which operation to use for the µ-calculus properties. This operation will be pushed further down to the atomic propositions, causing imprecision.

For approximating abstract intermediate values we can only safely increase the information of a value; therefore we $\oplus$ the µ-calculus properties and by extension the atomic propositions. When approximating concrete intermediate values, we can only safely decrease the information of a value; therefore we $\otimes$ the µ-calculus properties. This leaves $\otimes$ and $\oplus$ for respectively abstract transitions and concrete transitions when iterating over target states.

Note that we can safely add the value $u_\vee$ to a set of transitions, since *false* transitions do not influence µ-calculus properties as explained at Def. 14. Comparing transitions to $u_\vee$ like in the previous definition is now a matter of choosing T_1 or T_2 to be the empty set. It vacuously holds that $(H_1, H_2) \in H$ when either H_1 or H_2 is the empty set.

Theorem 17. *Let $H \subseteq \mathcal{P}(S_1) \times \mathcal{P}(S_2)$ be an extended mixed simulation relation from M_1 to M_2, and let ϕ be a closed µ-calculus formula. Then for every $(H_1, H_2) \in H$, $\oplus_{h_2 \in H_2}(\|\phi\|^{M_2}(h_2)) \leq_i \otimes_{h_1 \in H_1}(\|\phi\|^{M_1}(h_1))$.*

The result of an extended mixed simulation is shown in Fig. 6f. We can increase precision over the symmetric simulation of Fig. 6e because the most informative value for the first transition is equal to $Tt \oplus tf = Ttf$. This would satisfy item 3 of Def. 16, but by choosing Tt for the transition we ensure that the opposite direction described by item 3 is also satisfied.

V. Steerability example

In the previous section we have shown different requirements for a correct abstraction; each set of requirements weaker than the previous. Weaker requirements allow for increasing precision and reducing the size of the abstraction. The result can be seen in the sequence of abstractions of Fig. 6: each new abstraction uses more informative values for its transitions and atomic propositions, while still being a correct abstraction of the concrete system.

To give a better intuition of this increased precision, we will take a closer look at the nine-valued bilattice of Fig. 5. The motivation for developing the theory of this paper is to investigate multi-valued abstraction in the context of steerability: the possibility of guiding the execution of a program and thereby avoid bugs. Values of the nine-valued bilattice can be interpreted in this context as values indicating the steerability of transitions in a Kripke model.

In a Kripke model using classical Boolean logic, a transition from a state s to a state t with the value *true* indicates that the modelled program can progress from state s to state t. Similarly, a transition with the value *false* indicates that this is not possible. An execution of the Kripke model is a path through the graph of states and transitions. At each state a successor is picked non-deterministically from the set of *true* transitions. The Kripke model describes all possible executions caused by this non-determinacy.

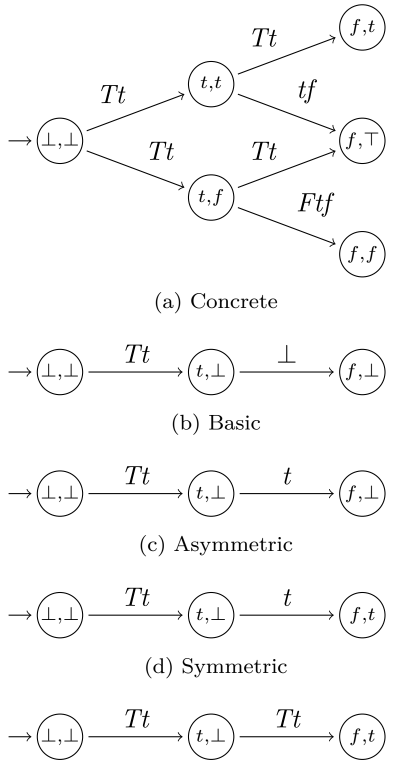
The bilattice of Fig. 5 can be used to encode steerability information in a model. Values of the bilattice are effectively subsets of $\{t, f, T, F\}$ with $\bot$ being the empty set, and $\top$ being the complete set. We use the convention that lowercase letters indicate steering and uppercase letters indicate defaults. Note that in this lattice Tt acts as *true*, being the largest element in the truth ordering, while Ff acts as *false*, being the smallest. Negation is defined as exchanging T with F and t with f in the subset. While the subset construction is helpful to understand the semantics behind the values, note that at the level of the bilattice we are oblivious to this internal structure and use the subsets as indivisible values.

The intuition of the individual values is that T indicates a transition that is enabled by default: during execution it can be non-deterministically chosen to further the execution. A value t indicates a transition that can be enabled when controlling the execution: if we want this transition to be considered, we will have to influence the execution. Similarly F is a transition that is disabled by default, while f can be disabled when controlled.

We could use all possible subsets of these base values to form a bilattice, but it turns out we can reduce the number of values by adding a restriction: if a subset contains an uppercase value, then it also needs to contain the corresponding lowercase value. For example, we do not allow the value T, but do allow the value Tt. The intuition behind this restriction is that a transition that is enabled by default can be trivially enabled when controlled, simply by not exerting any influence.

To indicate a steerable transition we use the values *tf*, *Ttf*, and *Ftf*. They respectively indicate a transition that: can be enabled or disabled when controlled; is enabled by default, but can be disabled when controlled; and is disabled by default, but can be enabled when controlled. Using these values in a multi-valued Kripke model enables us to detect how a property is influenced by the ability to steer an execution. A property with the value *tf* can be enforced or broken using steering, while a value *Ttf* holds by default, but can be broken using steering. A value *Tt* indicates a property which holds by default, similar to the value *true* when verifying a property using classical Boolean logic; the value *Ff* indicates a property which is broken by default, similar to *false*.

We can apply this interpretation of the nine-valued bilattice when evaluating properties over the Kripke model in Fig. 7a. Note that this example corresponds to the



(a) Concrete

(b) Basic

(c) Asymmetric

(d) Symmetric

(e) Extended

Figure 7: Steerability abstraction

concrete model in Fig. 6a, except that we have added an initial state at the front. Remember that each state contains two atomic propositions, p and q, the values of which are indicated in each node of the graph. Evaluating the property $\Box(p = t \land \Diamond(p = f))$ in this system yields Tt, which indicates that the property holds by default. (Note that equality in this property should be interpreted as syntactic equality and will resolve to either *true* or *false*.) We get this result by first evaluating $\Diamond(p = f)$ in the successors of the initial state, which gives $(Tt \land Tt) \lor (tf \lor Tt) = Tt$ and $(Tt \land Tt) \lor (Ftf \lor Tt) = Tt$; then we use these values to calculate the $\Box$ modality and get the final result $(\neg Tt \lor (Tt \land Tt)) \land (\neg Tt \lor (Tt \land Tt)) = Tt$.

To create an abstraction of this model and reduce the number of states, we might decide to combine states based on their value for p. All states where $p = t$ are combined, and all states where $p = f$ are combined, forming two new abstract states. The first step of this abstraction is shown in Fig. 6 where all states with $p = f$ are combined, while Fig. 7 shows the result of combining states with $p = t$.

Evaluating the property $\Box(p = t \land \Diamond(p = f))$ in these abstract models gives different results based on the abstraction requirements used. In a basic mixed simulation we get $\neg Tt \lor (Tt \land (\bot \land Tt) = \bot$, indicating we have no information to determine whether the property holds. In the asymmetric and symmetric mixed simulation we get $\neg Tt \lor (Tt \land (t \land Tt) = t$, indicating that we can at least steer the execution to ensure the property will hold. Finally, in the extended simulation we get $\neg Tt \lor (Tt \land (Tt \land Tt) = Tt$, indicating that even without steering the property will hold by default. With each improvement of the abstraction requirements we get a more informative answer when evaluating the µ-calculus property.

VI. Conclusion and future work

We have shown how the definition of mixed simulation can be modified into an extended mixed simulation. This new definition allows for a more general class of abstraction techniques which can offer better precision when inconsistent values are available in the logic. It unifies the notion of mixed simulation [1] with the abstraction technique used in [13].

Since the theory presented in this paper is generic in that it works for any distributive bilattice, we aim to apply this theory in developing new abstraction techniques, based on new multi-valued logics. More specifically, we want to apply this knowledge in techniques which encode additional information in the abstract model, such that the state space explosion problem can be dealt with more effectively. Currently we are investigating the encoding of steerability information in a multi-valued logic for use during execution steering and runtime verification.

References

[1] Y. Meller, O. Grumberg, and S. Shoham, "A framework for compositional verification of multi-valued systems via abstraction-refinement," in *ATVA*, vol. 5799 of *LNCS*, pp. 271–288, Springer, 2009.

[2] O. Grumberg, "Abstraction and refinement in model checking," in *FMCO*, vol. 4111 of *LNCS*, pp. 219–242, Springer, 2005.

[3] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement," in *CAV*, vol. 1855 of *LNCS*, pp. 154–169, Springer, 2000.

[4] G. Bruns and P. Godefroid, "Model checking partial state spaces with 3-valued temporal logics," in *CAV*, vol. 1633 of *LNCS*, pp. 274–287, Springer, 1999.

[5] M. Huth, R. Jagadeesan, and D. Schmidt, "Modal transition systems: A foundation for three-valued program analysis," in *ESOP*, vol. 2028 of *LNCS*, pp. 155–169, Springer, 2001.

[6] M. Fitting, "Bilattices and the theory of truth," *Journal of Philosophical Logic*, vol. 18, pp. 225–256, 1989.

[7] N. Belnap, *Modern Uses of Multiple-Valued Logics*, pp. 30–56. Reidel, 1977.

[8] C. Seger and R. Bryant, "Formal verification by symbolic evaluation of partially-ordered trajectories.," *Formal Methods in System Design*, vol. 6, no. 2, pp. 147–189, 1995.

[9] A. Gurfinkel, O. Wei, and M. Chechik, "Yasm: A software model-checker for verification and refutation," in *CAV*, vol. 4144 of *LNCS*, pp. 170–174, Springer, 2006.

[10] B. Konikowska and W. Penczek, "Reducing model checking from multi-valued CTL* to CTL*," in *CONCUR*, vol. 2421 of *LNCS*, pp. 226–239, Springer, 2002.

[11] M. Chechik, B. Devereux, S. Easterbrook, and A. Gurfinkel, "Multi-valued symbolic model-checking," *ACM TOSEM*, vol. 12, no. 4, pp. 371–408, 2003.

[12] A. Gurfinkel and M. Chechik, "Multi-valued model checking via classical model checking," in *CONCUR*, vol. 2761 of *LNCS*, pp. 263–277, Springer, 2003.

[13] A. Gurfinkel and M. Chechik, "Why waste a perfectly good abstraction," in *TACAS*, vol. 3920 of *LNCS*, pp. 212–226, Springer, 2006.

[14] Y. Shramko, J. M. Dunn, and T. Takenaka, "The trilattice of constructive truth values," *J. Log. Comput.*, vol. 11, no. 6, pp. 761–788, 2001.