Multi-Valued Simulation and Abstraction using Lattice Operations

STEFAN VIJZELAAR and WAN FOKKINK, Vrije Universiteit Amsterdam

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 [Meller et al. 2009] 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. In addition we present a set of abstraction rules, compatible with the extended notion, for constructing abstract models.

1. 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 [Dams et al. 1997; Grumberg 2005], but this can introduce spurious results [Clarke et al. 2000]: 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 ([Bruns and Godefroid 1999; Huth et al. 2001]) 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 [Fitting 1989], of which the four-valued Belnap logic [Belnap 1977] is a well-known example. Verification techniques using such four-valued logics have been successful in e.g. verifying logical circuits [Seger and Bryant 1995] and software [Gurfinkel et al. 2006]. 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 ([Konikowska and Penczek 2002; Chechik et al. 2003]). The information ordering makes it possible to model incomplete information and thereby detect inconclusive results. Despite using a multi-valued logic, temporal properties can still be verified using classical model checkers through decomposition [Gurfinkel and Chechik 2003]. 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 [Meller et al. 2009]. Mixed simulation is similar to bisimulation, but allows for loss of information. 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 a less or equally 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 the mixed simulations presented in this paper. The hierarchy of mixed simulations contains basic, symmetric, and extended mixed simulation. We will show that the mixed simulation from [Meller et al. 2009] falls between basic and symmetric mixed simulation in strength of its abstraction requirements. We further weaken the requirements of symmetric mixed simulation to create an extended mixed simulation, which allows for the increased precision of the abstraction technique described in [Gurfinkel and Chechik 2006] and extends this technique to any multi-valued logic.

Mixed simulation in [Meller et al. 2009] does not take into account inconsistent values, i.e. values indicating a contradiction. Only consistent partial distributive bilattices are considered, which causes 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 ordering to further increase precision. It is shown in [Gurfinkel and Chechik 2006] how an abstraction using inconsistent values can be constructed for four-valued Belnap logic. Mixed simulation as defined in [Meller et al. 2009] however does not allow it. We extend the definition of mixed simulation to account for this construction.

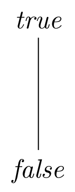
Our motivation for improving the precision of multi-valued abstraction is an interest in using multi-valued lattices to model steerability. By steering an execution one can avoid bugs and reduce the number of states in the Kripke model that need to be explored during model checking. We present an example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

A preliminary paper appeared as [Vijzelaar and Fokkink 2015]. In comparison to that paper, we add proofs to the main body, and go into more detail on the concept of intermediate truth values and their approximations. The abstraction method of [Vijzelaar et al. 2014] is integrated in section 7 and extended based on the theory of extended mixed simulation. This results in a set of abstraction rules for constructing an abstract Kripke model from a given more concrete model, such that the two models are in an extended mixed simulation.

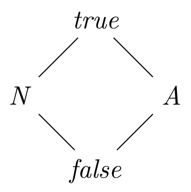
We start with preliminaries in section 2 and present a hierarchy of abstraction requirements for multisets in section 3. We explain how states of Kripke models can be treated as multisets in section 4, such that we can create a matching hierarchy of abstraction requirements for Kripke models in section 5. In section 6 we go into the details of steering logic to present an example for each type of mixed simulation, and in section 7 we describe abstraction rules for correctly constructing an abstract model.

2. PRELIMINARIES

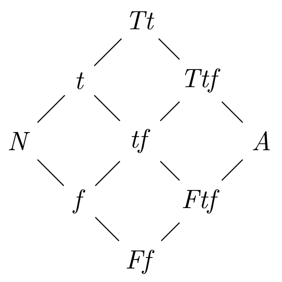
In the following subsections we define lattices and show how they can be used to create quasi-Boolean logics. Kripke models, which normally use classical Boolean logic, are given a multi-valued definition using quasi-Boolean logics. Similarly, temporal properties expressed in μ-calculus are lifted to a multi-valued setting. Finally we present bilattices, which add an additional ordering to lattices, and show how this ordering can be used to define abstraction.



(a) Classical logic



(b) Belnap logic



(c) Steering logic

Fig. 1: Distributive lattices

2.1. Lattices

Multi-valued logics can be defined using lattices. The truth values of the logic are the elements of the lattice, and the logical connectives correspond to meet and join operations.

Definition 2.1. A lattice is a tuple $\mathcal{L} = \langle L, \leq \rangle$, with L a set of elements, and $\leq$ a partial ordering on elements in L. Any two elements of L 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 $\top$). Meet and join operations are associative, commutative and idempotent. In a distributive lattice we have the property that meet and join distribute over each other.

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. In multi-valued logics we can use lattices which have additional elements beyond *true* and *false*.

2.2. Quasi-Boolean logic

The multi-valued logics we are interested in are quasi-Boolean logics [Bialynicki-Birula and Rasiowa 1957], 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. The lattice of a quasi-Boolean logic is not required to have a complement y for each element x such that $x \sqcup y = \top$ and $x \sqcap y = \bot$.

Definition 2.2. A quasi-Boolean logic is a tuple $\mathcal{Q} = \langle L, \leq, \neg \rangle$, with $\langle L, \leq \rangle$ a finite distributive lattice, and $\neg$ an involution following De Morgan's laws with respect to meet and join operations over the lattice.

The lattice $\langle L, \leq \rangle$ of a quasi-Boolean logic is bounded; we define the least element as *false*, and the greatest element as *true*. Lattice operations map to the Boolean connectives: meet is used as a conjunction ($\wedge$), and join is used as a disjunction ($\vee$). Negation ($\neg$) requires an appropriate involution (i.e., being its own inverse) which in addition should adhere to De Morgan's laws: $\neg(x \wedge y) = \neg x \vee \neg y$ and $\neg(x \vee y) = \neg x \wedge \neg y$. It follows that disjunction and conjunction are distributive, and the law of double negation applies.

A typical example of a distributive lattice is the one used for Belnap logic [Belnap 1977], as depicted in Fig. 1b. This logic can be used to encode classical Boolean abstraction as explained in section 2.6. The steering logic shown in Fig. 1c is based on the trilattice of constructive truth values of [Shramko et al. 2001]. It encodes steering information in the logic and will be discussed in more detail in section 6. Both lattices define negation by inverting their partial order: $\neg true = false$, $\neg N = N$, $\neg Ftf = Ttf$, etc.

Note that the lattices in Fig. 1 are Hasse diagrams in which only the transitive reduction of the partial ordering is represented by lines between elements. An element higher up in the figure is larger in the partial ordering with respect to connected elements further down.

2.3. Multi-valued Kripke models and µ-calculus

A Kripke model [Kripke 1963] is used in model checking to formally describe a system under verification. It is a transition system containing all possible behaviours of the system being verified: nodes describe the possible states of a system and transitions indicate how the system can change between these states. By specifying wanted or unwanted behaviour using a temporal logic, such as µ-calculus [Kozen 1983], it is possible to verify whether this behaviour exists in a given Kripke model.

Multi-valued Kripke models are a generalisation of Kripke models and can use values of any multi-valued 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 multi-valued logic. In the following we use quasi-Boolean logics as our multi-valued logic of choice.

Definition 2.3. A multi-valued Kripke model is a tuple $M = \langle \mathcal{Q}, AP, S, s^0, R, \Theta \rangle$, where $\mathcal{Q} = \langle L, \leq, \neg \rangle$ is a quasi-Boolean logic, 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 $\mathcal{Q}$, and $\Theta : AP \to (S \to L)$ a labelling function assigning truth values to states for each atomic proposition.

Definition 2.4. Given a set of atomic propositions AP and a set of propositional variables Var. The µ-calculus (in so-called 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 $Z \in Var$.

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, as we will see in the semantics defined below. An environment $\mathcal{V}$ is used to assign values to free propositional variables. A formula is closed if there are no free propositional variables: all occurrences of propositional variables are bound by a fixed point operator.

Definition 2.5. The semantics $\|\phi\|^M_{\mathcal{V}} : S \to L$ of a µ-calculus formula ϕ assigns a truth value to each state of a multi-valued Kripke model M, with respect to an environment $\mathcal{V} : Var \to (S \to L)$ mapping variables to a multi-valued set of states, is defined as follows:

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

We omit $\mathcal{V}$ if the µ-calculus formula ϕ is closed and write $\|\phi\|^M$ instead of $\|\phi\|^M_{\mathcal{V}}$.

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.

2.4. Bilattices

In a bilattice [Fitting 1989] 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 $\leq_t$ and the other the information ordering $\leq_i$ (see e.g. [Meller et al. 2009]). The truth ordering, together with a suitable definition for negation, can be used to define a quasi-Boolean logic. The information ordering can be used to model abstraction. To distinguish between lattice 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 ordering.

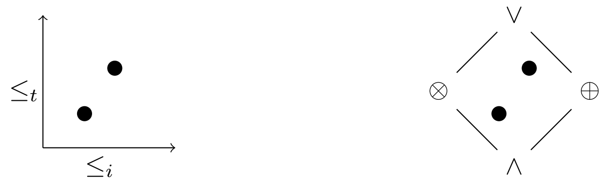
Definition 2.6. A bilattice is a tuple $\mathcal{B} = \langle L, \leq_1, \leq_2 \rangle$, with L a set of elements, and $\leq_1$, $\leq_2$ partial orderings on elements in L. Both $\langle L, \leq_1 \rangle$ and $\langle L, \leq_2 \rangle$ form a lattice.

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 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 ordering it is defined over, but can be monotonous with respect to the orthogonal ordering.

The lattices in Fig. 1b and Fig. 1c are bilattices because they have an additional orthogonal partial ordering. Similar to the truth ordering in the vertical direction, they have an information ordering in the horizontal direction. An element more to the right is larger in this ordering with respect to connected elements further to the left. A truth value t of one of these lattices should be interpreted as any value equal or more informative than t. It is allowed to lower a value t in the information order, but this will cost precision. We can use this loss of information to model abstraction.

In this context we can give an intuition for the $\otimes$ and $\oplus$ operators. The $\otimes$ operator is a consensus operator: if a value can be $p$or q, then the consensus is that it is at least $p \otimes q$. The $\oplus$ operator can be understood as a gullibility operator: if a value is both p and q, then it must be at least equal or more informative than $p \oplus q$. For example in the Belnap logic of Fig. 1b we have $\mathit{true} \otimes \mathit{false} = N$: if something can be true or false then we have no information. This is indicated by the value N, which is the lowest value in the information order. In the steering logic of Fig. 1c we distinguish between true by default, indicated by Tt, and true by steering, indicated by t. In this logic we have $Tt \oplus tf = Ttf$: if something is both true by default and can be made true or false by steering, then it is true by default and can be made false by steering. This is indicated by the value Ttf. We will take a closer look at the semantics of steering logic in section 6.

In some of our figures, we will depart from the conventions of Hasse diagrams and instead depict bilattices in general, without going into their specific structure. In those cases, the transitive reduction of the partial orderings is no longer shown: elements of the lattice are drawn as points in two-dimensional space, with each dimension corresponding to an ordering (Fig. 2a). Unconnected elements may or may not be comparable. The partial ordering with respect to a specific element is depicted by a cone of bounding diagonal lines, such that elements falling within the cone are comparable and possibly equal to this element. Lattice operations form a diamond shape which should be interpreted as the result of the lattice operation on the multiset of elements within (Fig. 2b).



(a) Orderings (b) Operations

Fig. 2: Bilattices

2.5. Lattice operations and (inverted) orderings

To simplify some of the definitions, we use the notions of "lattice operations" and "(inverted) orderings". These are used to remove the distinction between meet and join operations and allows us to use a generic operator $\textcircled{?}$ to indicate any lattice operation. An operation $\textcircled{?}$ over the multiset can be any of the meet or join operations of the lattice.

Definition 2.7. An (inverted) ordering $\leq_o$ of a lattice $\mathcal{L} = \langle L, \leq \rangle$, is either the ordering $\leq$ or its inverse. The (inverted) ordering $\leq_?$ with respect to some operator $\textcircled{?}$, is the (inverted) ordering such that $\textcircled{?}$ calculates the supremum with respect to $\leq_?$.

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 (inverted) ordering $\leq_?$. In addition, this allows us to define a unit element $u_?$ and an annihilator element $a_?$ for the operator $\textcircled{?}$.

Definition 2.8. The unit element $u_?$ of a lattice operation $\textcircled{?}$ with (inverted) ordering $\leq_?$ over a lattice $\mathcal{L} = \langle L, \leq \rangle$, is that element $u_? \in L$ such that $\forall l \in L : u_? \leq_? l$.

Adding the unit element $u_?$ to a multiset has no influence when calculating $\textcircled{?}$ over the multiset: the unit element can be safely ignored. Operations over empty multisets reduce to their unit element.

Definition 2.9. The annihilator element $a_?$ of a lattice operation $\textcircled{?}$ with (inverted) ordering $\leq_?$ over a lattice $\mathcal{L} = \langle L, \leq \rangle$, is that element $a_? \in L$ such that $\forall l \in L : l \leq_? a_?$.

Adding the annihilator element $a_?$ to a multiset ensures that $\textcircled{?}$ over the multiset will be equal to $a_?$: all other elements of the multiset can be ignored. Note that every bounded lattice has both a unit and an annihilator element.

2.6. Abstraction

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 [Gurfinkel and Chechik 2006] defines a four-valued logic as described by [Belnap 1977] (Fig. 1b). One can interpret N as neither *true* nor *false*, and A as both *true* and *false*. To model abstraction in Belnap logic, we use N or *true* for may transitions, and A or *true* for must transitions.

In general, the elements of the information ordering can be seen as sets, which in the case of Belnap logic 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 $\{t, f\}$, but to its subsets. The singleton $\{t\}$ is equal to *true*, and the singleton $\{f\}$ is equal to *false*. This construction also allows for two additional truth values which contain *none* (N) or *all* (A) of the elements in $\{t, f\}$. Abstractions can use these additional values to model a loss of information (N) or increase precision (A). This interpretation of truth values as subsets can be extended to any quasi-Boolean logic based on a distributive bilattice, since it is always possible to find a lattice of sets that is isomorphic to a distributive lattice.

With a suitable information ordering, we can define abstraction for the multi-valued setting. Classical under- and over-abstraction can be seen as constructing two classical Boolean models from an abstract model using Belnap logic: one model for truth values larger than or equal in truth to N, and one model for truth values larger than or equal in truth to A. However, instead of using separate models, it is possible to directly use the abstract multi-valued Kripke model. This requires a multi-valued definition of abstraction between Kripke models.

Definition 2.10. Let $M_n = \left\langle \mathcal{Q}, AP, S_n, s_n^0, R_n, \Theta_n \right\rangle$ for $n \in \{1, 2\}$ be multi-valued Kripke models. The model M_2 abstracts M_1 if for any closed µ-calculus formula ϕ:

$$\|\phi\|^{M_2}(s_2^0) \leq_i \|\phi\|^{M_1}(s_1^0)$$

We define abstraction of multi-valued Kripke models with respect to µ-calculus properties. Since we are interested in abstraction of existing models, we generally call M_2 the abstract model, and M_1 the concrete model. (Note that it is equally fine to call M_2 the concrete model, and M_1 the refined model, since refinement is the inverse of abstraction. Our theory makes no distinction between the two.) The above definition requires that a property evaluated over the abstract model is less or equally informative than the same property evaluated over the concrete model. This relation between two Kripke models can be ensured by imposing specific requirements on the models.

To prove correctness of abstraction requirements on Kripke models, it is useful to also define the correct abstraction of multisets with respect to the lattice operations. This is because the evaluation of µ-calculus properties in a Kripke model can be described as lattice operations over multisets. 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.

Definition 2.11. A lattice operation $\textcircled{?}$ is applied recursively to a multiset $\Phi = \{\phi_1, ..., \phi_n\}$ of n elements, notation $\textcircled{?}\,\Phi$, as follows: $\Phi_1 = \phi_1$, $\Phi_i = \Phi_{i-1} \textcircled{?} \phi_i$ for $1 < i < n$, and $\textcircled{?}\,\Phi = \Phi_n$.

Remark 2.12. Since Φ is a multiset and a lattice operation $\textcircled{?}$ is associative and commutative, the ordering of the operands can be chosen arbitrarily. We write $\textcircled{?}\,\Phi = \phi_1 \textcircled{?} \ldots \textcircled{?} \phi_n$.

Definition 2.13. Let Φ and Ψ be multi-valued multisets. The multiset Φ abstracts Ψ if for any lattice operation $\textcircled{?}$:

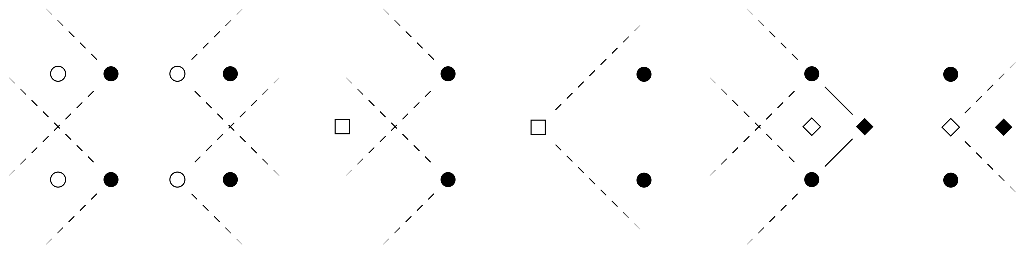
$$\textcircled{?}\,\Phi \leq_i \textcircled{?}\,\Psi$$

In the following sections we will look in more detail at abstraction of multisets and Kripke models. We will define a hierarchy of abstraction requirements on both multisets and Kripke models: each set of requirements on the abstraction of multisets will be connected to a set of requirements on the abstraction of Kripke models.

3. ABSTRACTING MULTI-VALUED MULTISETS

To simplify abstracting multi-valued Kripke models, 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 the modalities of µ-calculus formulas.

In this section we present a hierarchy of abstraction requirements for multisets. Each set of requirements is weaker than the next and thereby allows more precise abstractions. We start with a bijection between multisets and weaken its definition: first by allowing multisets of unequal size, then by adding unit elements, and finally by considering the lattice orderings.



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

Fig. 3: Abstracting truth values

These abstraction requirements are defined using multisets so they can be applied when truth values are implicit and could be duplicate.

3.1. Bijection between multisets

Let Φ and Ψ be multisets of truth values for which we want to ensure that Φ abstracts Ψ with respect to some lattice operation $\textcircled{?}$. One way to ensure a correct abstraction is to provide a relation $R \subseteq \Phi \times \Psi$ which uniquely pairs the elements of Φ and Ψ such that $\phi \leq_i \psi$ for each pair of ϕ and ψ in R. Then monotonicity with respect to the information ordering will ensure that, regardless of the chosen lattice operation $\textcircled{?}$, the answer over Φ will be less or equally informative than the answer over Ψ. We generalise this approach to any (inverted) ordering $\leq_o$ of a bilattice.

Proposition 3.1. *Given a lattice operation $\textcircled{?}$ of a distributive bilattice $\mathcal{B}$ over elements L, with (inverted) ordering $\leq_o$. For two multisets Φ and Ψ with elements from L, the formula $\textcircled{?}\Phi \leq_o \textcircled{?}\Psi$ holds if there exists a relation $R \subseteq \Phi \times \Psi$ such that:*

(*1*) *R is a bijective function*
(*2*) $\forall(\phi, \psi) \in R : \phi \leq_o \psi$

Proof. R ensures that Φ and Ψ have the same number of elements n. These can be reordered, such that $\textcircled{?}\Phi = \phi_1 \textcircled{?} \dots \textcircled{?} \phi_n$, $\textcircled{?}\Psi = \psi_1 \textcircled{?} \dots \textcircled{?} \psi_n$, and $\phi_i \leq_o \psi_i$ for all $1 \leq i \leq n$. The formula follows from the monotonicity of $\textcircled{?}$ with respect to $\leq_o$. $\square$

In Fig. 3a we see how for each abstract value (○) of Φ there is a unique concrete value (●) of Ψ that is equally or more informative: each abstract value is in the information cone of a corresponding concrete value, indicating it is less or equally informative than that value. To complete the bijection, we also require the opposite to hold, as depicted in Fig. 3b: for each concrete value (●) of Ψ there is a unique abstract value (○) of Φ that is equally or more informative.

3.2. Multisets of unequal size

We weaken the requirement for abstraction by allowing an unequal number of values in the multisets Φ and Ψ; this removes the uniqueness requirement of pairs. It is sufficient if for each value ϕ in Φ there exists a value ψ in Ψ such that $\phi \leq_i \psi$, and for each value ψ in Ψ there exists a value ϕ in Φ such that $\phi \leq_i \psi$. We use idempotency of the lattice operations to duplicate elements already present in Φ or Ψ and create the required bijection without influencing the result of the lattice operations.

Proposition 3.2. *Given a lattice operation $\textcircled{?}$ of a distributive bilattice $\mathcal{B}$ over elements L, with (inverted) ordering $\leq_o$. For two multisets Φ and Ψ with elements from L, the formula $\textcircled{?}\Phi \leq_o \textcircled{?}\Psi$ holds if both:*

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

PROOF. Start with $\Phi' = \Phi$ and $\Psi' = \Psi$. For each $\phi \in \Phi$ add the element ψ of item 1 to Ψ', and for each $\psi \in \Psi$ add the element ϕ of item 2 to Φ'. Pairing these elements gives a relation $R \subseteq \Phi' \times \Psi'$ with $\forall(\phi', \psi') \in R : \phi' \leq_o \psi'$. Prop. 3.1 gives $\textcircled{?}\,\Phi' \leq_o \textcircled{?}\,\Psi'$, and by idempotency of $\textcircled{?}$ the formula holds. □

In Fig. 3c we see for the abstract value (□) of Φ that two concrete values (●) of Ψ are equally or more informative: the value is in the information cone of both concrete values. The opposite is no longer symmetric, as shown in Fig. 3d: for each concrete value (●) of T_2 there is an abstract value (□) of T_1 that is equally or more informative, but it is not unique. Note that the abstract value is the $\otimes$ of the concrete multiset.

3.3. Unit elements

We can further weaken the requirements for abstraction if we only consider a single lattice operation over the multisets Φ and Ψ. For this specific lattice operation the unit element can be added to the multiset without influencing the result. Assume the operation over the multiset is a disjunction $\vee$. If we can show for a value ψ in Ψ that *false* $\leq_i \psi$, then we no longer require a value ϕ in Φ such that $\phi \leq_i \psi$: we simply add *false* to Φ. Similarly for those values ϕ in Φ with $\phi \leq_i$ *false*, we add *false* to Ψ.

PROPOSITION 3.3. *Given a lattice operation $\textcircled{?}$ of a distributive bilattice $\mathcal{B}$ over elements L, with (inverted) ordering $\leq_o$, and a unit element $u_?$ of $\textcircled{?}$. For two multisets Φ and Ψ with elements from L, the formula $\textcircled{?}\,\Phi \leq_o \textcircled{?}\,\Psi$ holds if both:*

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

PROOF. Let $\Phi' = \Phi \cup u_?$ and $\Psi' = \Psi \cup u_?$, then $\textcircled{?}\,\Phi' \leq_o \textcircled{?}\,\Psi'$ by Prop. 3.2. (Use $u_? \leq_o u_?$ for the new elements.) Since $u_?$ is the unit element of $\textcircled{?}$, we have $\textcircled{?}\,\Phi = \textcircled{?}\,\Phi'$, $\textcircled{?}\,\Psi = \textcircled{?}\,\Psi'$, and the formula holds. □

This proposition does however limit the correctness of our abstraction to the corresponding lattice operation of the unit element. Adding the value *true* to a multiset, because it is the unit element for conjunction, invalidates calculating the disjunction over the same multiset, because *true* is the annihilator element for disjunction.

3.4. Bilattice orderings

The requirements on the abstraction can be further weakened if we take into account the associated ordering of the lattice operation. We can add a new truth value l to a multiset Φ of truth values, as long as it does not influence the result of $\textcircled{?}\,\Phi$. Since $\textcircled{?}$ calculates the supremum with respect to its (inverted) ordering $\leq_i$, any value $l \leq_? \textcircled{?}\,\Phi$ can be added.

PROPOSITION 3.4. *Given a lattice operation $\textcircled{?}$ of a distributive bilattice $\mathcal{B}$ over elements L, with (inverted) orderings $\leq_o$ and $\leq_?$. For two multisets Φ and Ψ with elements from L, the formula $\textcircled{?}\,\Phi \leq_o \textcircled{?}\,\Psi$ holds if both:*

(1) $\forall \phi \in \Phi : \exists l \in L : \phi \leq_o l \wedge l \leq_? \textcircled{?}\,\Psi$
(2) $\forall \psi \in \Psi : \exists l \in L : l \leq_o \psi \wedge l \leq_? \textcircled{?}\,\Phi$

PROOF. Start with $\Phi' = \Phi$ and $\Psi' = \Psi$. For each $\phi \in \Phi$ add the element l of item 1 to Ψ', and for each $\psi \in \Psi$ add the element l of item 2 to Φ'. Pairing these elements gives a relation $R \subseteq \Phi' \times \Psi'$ with $\forall(\phi', \psi') \in R : \phi' \leq_o \psi'$, and by Prop. 3.1 we get $\textcircled{?}\,\Phi' \leq_o \textcircled{?}\,\Psi'$. For each l added to Φ or Ψ, respectively $l \leq_? \textcircled{?}\,\Phi$ or $l \leq_? \textcircled{?}\,\Psi$; by definition of $\textcircled{?}$ we can conclude that $\textcircled{?}\,\Phi = \textcircled{?}\,\Phi'$ and $\textcircled{?}\,\Psi = \textcircled{?}\,\Psi'$. Therefore, $\textcircled{?}\,\Phi = \textcircled{?}\,\Phi' \leq_o \textcircled{?}\,\Psi' = \textcircled{?}\,\Psi$. □

Abstracting a multiset of concrete values (●) using a single abstract value (□), as shown in Fig. 3c, can cause a significant loss of information, since the abstract value has to be less or equally informative than all concrete values in the multiset. We could improve precision by adding abstract values (○) for each individual concrete value, as shown in Fig. 3a, but this would increase the size of the abstract multiset. A better option is to increase the precision of the abstract value (◇), by considering an additional concrete value (◆).

In Fig. 3e we consider both conjunctions and disjunctions over the multiset of concrete values. The most informative concrete value that can be added under these conditions is the $\oplus$ of the concrete multiset (◆). The abstract value (◇) no longer needs to be in the information cone of all concrete values, but only in the cone of this additional value. Note that the reverse case of Fig. 3f is no longer correct by default. This issue will be explored in more detail in section 7.

4. ABSTRACTING INTERMEDIATE TRUTH VALUES

In the previous section we looked at abstracting multisets of truth values such that lattice operations on these values lead to a less or equally 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. State properties are specific to the state they are evaluated in, since each state has its own assignment of truth values to atomic propositions. To abstract these atomic propositions, we have to take into account that they can be used by any 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 model is less or equally informative than in its paired state of the concrete model. This ensures that any state property derived from these propositions is also less or equally informative.

We are then 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 $\Diamond$). By defining so-called intermediate truth values for the modal operators, these modalities can be described as lattice operations over multisets, and as such the abstraction rules for multisets can be applied.

4.1. Comparing intermediate truth values

Evaluating the $\Box$ and $\Diamond$ modalities of µ-calculus for a source state s requires two parts: for each possible target state t the value of the transition from s to t is combined with the value of a µ-calculus property at t, and these intermediate results are combined to get the final result. Writing the definitions of the model operators given in Def. 2.5 as functions gives $\|\Box\phi\|(s) = \bigwedge_{t\in S}(\neg R(s,t) \vee \|\phi\|(t))$ for the box modality and $\|\Diamond\phi\|(s) = \bigvee_{t\in S}(R(s,t) \wedge \|\phi\|(t))$ for the diamond modality. In these definitions we call $R(s,t)$ the *transition*, and $\|\phi\|(t)$ the *µ-expansion*. Depending on the modality, a disjunction or conjunction combines these values into an *intermediate truth value*, negating the transition when necessary. There is an intermediate truth value for each target state t, and the modalities require a final lattice operation over a multiset of their intermediate truth values to get the final result.

Definition 4.1. An *intermediate truth value* of $\|\Box\phi\|(s)$ is a value $\neg R(s,t) \vee \|\phi\|(t)$ with $t \in S$. An *intermediate truth value* of $\|\Diamond\phi\|(s)$ is a value $R(s,t) \wedge \|\phi\|(t)$ with $t \in S$. A multiset of intermediate truth values for a specific $s \in S$ is created by iterating over all $t \in S$.

During abstraction, we cannot calculate the intermediate truth values in a concrete Kripke model without doing actual model checking, which is something we want to defer until after the abstraction. This is why the internal structure of an intermediate truth value remains important: we want to ensure that the intermediate truth values follow the rules of abstraction for multisets without knowing which modalities will need to be evaluated in a specific state. Only the values of transitions and atomic propositions are known. This can be done using monotonicity of conjunction, disjunction and negation with respect to the information ordering.

Direct comparison of intermediate truth values is not possible, but we can instead compare the transition and µ-expansion, as expressed in Prop. 4.2. If both the transition and µ-expansion of an abstract intermediate value are less or equally informative than those of a concrete intermediate value, then the same holds for the intermediate values as a whole. The first part is easy to verify, since transition values are known; the second part is more difficult, since we do not know what property is evaluated by the µ-expansion. The abstraction should hold for any µ-calculus property.

PROPOSITION 4.2. *To prove the inequalities* $\neg R(s_2, t_2) \vee \|\phi\|(t_2) \leq_i \neg R(s_1, t_1) \vee \|\phi\|(t_1)$ *and* $R(s_2, t_2) \wedge \|\phi\|(t_2) \leq_i R(s_1, t_1) \wedge \|\phi\|(t_1)$ *between intermediate truth values, it is sufficient to prove* $R(s_2, t_2) \leq_i R(s_1, t_1)$ *and* $\|\phi\|(t_2) \leq_i \|\phi\|(t_1)$*.*

PROOF. Follows directly from monotonicity of the operators with respect to $\leq_i$. □

If we assume as a base case that the µ-expansion is a state property, then we only have to consider the atomic propositions of the target state. State properties can contain conjunctions, disjunctions and negations; but if all atomic propositions of the abstract target are less or equally informative than those of the concrete target, then the same will hold for state properties over these atomic propositions. If the µ-expansion is not a state property then we can prove it less or equally informative by induction on the target states, whose intermediate truth values should in turn also be correct abstractions.

PROPOSITION 4.3. *To prove an inequality* $\|\phi\|(t_2) \leq_i \|\phi\|(t_1)$ *between µ-expansions, it is sufficient to prove* $\forall a \in AP : \Theta_2(a)(t_2) \leq_i \Theta_1(a)(t_1)$ *if by induction we can assume* $\|\Box\psi\|(t_2) \leq_i \|\Box\psi\|(t_1)$ *and* $\|\Diamond\psi\|(t_2) \leq_i \|\Diamond\psi\|(t_1)$ *for any µ-calculus formula* ψ*.*

PROOF. Follows directly from monotonicity of the operators with respect to $\leq_i$ if we can assume the induction step. □

Note that by separately comparing transitions and atomic propositions of the Kripke model, we actually lose precision. Intermediate values can be less or equally informative, without this being required to hold for both the transition and µ-expansion. The inequalities from monotonicity are sufficient but not necessary. The same holds for atomic propositions in the µ-expansion. It is however the best option given the circumstances, since we do not know what µ-calculus properties will be evaluated in each state. That would require knowledge of the µ-calculus property evaluated in the initial state of the concrete Kripke model, and a form of model checking to subsequently determine the µ-expansions evaluated at each state.

4.2. Approximating intermediate truth values

In the previous subsection we compared two intermediate truth values by comparing the transitions and atomic propositions. In some cases, for example in Prop. 3.4, we want to compare a single intermediate truth value with a multiset of intermediate truth values: the multiset is combined using a lattice operation before comparing it with the single value. This prevents us from simply comparing the transitions and µ-expansions, because the combined multiset no longer has the form of an intermediate truth value. To solve this problem, we can approximate the lattice operation by pushing it down in the parse tree of the multiset.

Assume we have three intermediate truth values: $t_1 = R_1 \wedge \phi_1$, $t_2 = R_2 \wedge \phi_2$, and $t_3 = R_3 \wedge \phi_3$. To ensure that $t_3 \leq_i t_1 \textcircled{?} t_2$, using monotonicity, requires $t_1 \textcircled{?} t_2$ to be of the form $R_? \wedge \phi_?$. Only then can we verify that $R_3 \leq_i R_?$ and $\phi_3 \leq_i \phi_?$.

PROPOSITION 4.4. *Pushing down the* $\textcircled{?}$ *operator in the parse tree of* $(R_1 \wedge \phi_1) \textcircled{?} (R_2 \wedge \phi_2)$ *to get the approximation* $(R_1 \textcircled{?} R_2) \wedge (\phi_1 \textcircled{?} \phi_2)$ *does not necessarily preserve its truth value.*

PROOF. Using classical Boolean logic, disjunction as the $\textcircled{?}$ operator, $R_1 = \phi_2 = \mathit{true}$, and $R_2 = \phi_1 = \mathit{false}$, gives $(R_1 \wedge \phi_1) \vee (R_2 \wedge \phi_2) = \mathit{false}$ while $(R_1 \vee R_2) \wedge (\phi_1 \vee \phi_2) = \mathit{true}$. □

This imprecision of the approximation can compromise the abstraction. We prevent 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 less or equally informative than the approximation will also be less or equally informative than the actual value of the operation.

Control over the approximation can be further improved by using an additional lattice operation. The value $(\phi_1 \textcircled{?} \psi_1) \textcircled{1} (\phi_2 \textcircled{?} \psi_2)$ can be approximated by pushing down the $\textcircled{1}$ operator, giving $(\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{1} \psi_2)$, but this result will always be higher or equal in the (inverted) ordering $\leq_1$. This poses a problem when $\textcircled{1}$ is the $\oplus$ operator, and we want a less informative approximation because of abstraction. Adding an additional lattice operator $\textcircled{2}$, resulting in the approximation $(\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{2} \psi_2)$, gives more control.

LEMMA 4.5. *The formula* $(\phi_1 \textcircled{?} \psi_1) \textcircled{1} (\phi_2 \textcircled{?} \psi_2) \leq_o (\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{2} \psi_2)$, *with* $\textcircled{?}$, $\textcircled{1}$, $\textcircled{2}$ *each an arbitrary lattice operations of the distributive bilattice* $\mathcal{B}$, *and* $\leq_o$ *an (inverted) ordering of* $\mathcal{B}$, *holds if* $\psi_1 \leq_o (\psi_1 \textcircled{2} \psi_2)$ *and* $\psi_2 \leq_o (\psi_1 \textcircled{2} \psi_2)$.

PROOF. Due to $\mathcal{B}$ being distributive, $\textcircled{?}$ distributes over $\textcircled{1}$ and we have $(\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{2} \psi_2) = (\phi_1 \textcircled{?} (\psi_1 \textcircled{2} \psi_2)) \textcircled{1} (\phi_2 \textcircled{?} (\psi_1 \textcircled{2} \psi_2))$. Monotonicity of $\textcircled{?}$ and $\textcircled{1}$ with respect to $\leq_o$ ensures that the formula holds if the requirements on ψ_1 and ψ_2 are met. □

PROPOSITION 4.6. *Given three arbitrary lattice operators* $\textcircled{?}$, $\textcircled{1}$, $\textcircled{2}$ *of the distributive bilattice* $\mathcal{B}$, *with (inverted) orderings* $\leq_1$, $\leq_2$. *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) $(\phi_1 \textcircled{?} \psi_1) \textcircled{2} (\phi_2 \textcircled{?} \psi_2) \leq_1 (\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{2} \psi_2)$

PROOF. Follows directly from the lattice operations by applying Lemma 4.5. □

An overview of all possible approximations is given in Fig. 4. The singleton operators indicate the results of directly combining $\phi_1 \textcircled{?} \psi_1$ and $\phi_2 \textcircled{?} \psi_2$ using that operator, while the pairs of operators show the approximation $(\phi_1 \textcircled{1} \phi_2) \textcircled{?} (\psi_1 \textcircled{2} \psi_2)$. We see that pushing down a $\oplus$ operator into two $\oplus$ operators will give an equally or more informative answer. An approximation using the $\wedge$ and $\otimes$ operators will be: less or equally informative than $\oplus$ or $\wedge$, and less or equally truthful than $\vee$ or $\otimes$. Note that switching the order of a pair of operators does not change their position in the figure.

5. ABSTRACTING MULTI-VALUED KRIPKE MODELS

In this section we apply the abstraction rules for multisets to intermediate truth values of multi-valued Kripke models. The resulting abstraction requirements can guarantee a correct abstraction relation between two Kripke models. For each set of abstraction rules we give an example in Fig. 5 using the distributive bilattice shown in Fig. 1c. These models will be used as a running example in the next two sections. The intuition behind the truth values will be explained in section 6.

Since we mirror the hierarchy of abstraction requirements on multisets presented in section 3, each set of requirements on Kripke models is weaker than the next, and allows abstractions

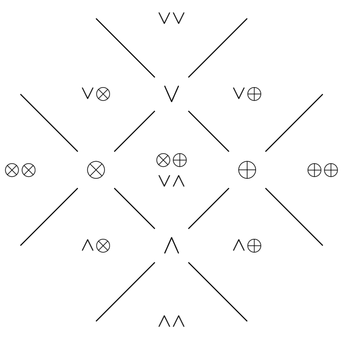


Fig. 4: Approximations

that are increasingly more precise. We present these requirements as different types of mixed simulations, after the simulation presented in [Meller et al. 2009]: basic, symmetric and extended mixed simulation. We show that the (asymmetric) mixed simulation of [Meller et al. 2009] falls between our basic and symmetric definitions. In addition our extended mixed simulation agrees with the abstraction method using inconsistent values as presented in [Gurfinkel and Chechik 2006].

5.1. Basic mixed simulation

We start by allowing a single abstract state to simulate multiple concrete states and vice versa. This is what one would expect from a standard bisimulation between Kripke models. The result is a stronger definition of mixed simulation than the one described in [Meller et al. 2009], since we do not exempt certain transition values, as is done in [Meller et al. 2009] and the upcoming variants of mixed simulation.

Definition 5.1. Let $M_n = \langle \mathcal{B}, AP, S_n, s_n^0, R_n, \Theta_n \rangle$ for $n \in \{1, 2\}$ be multi-valued Kripke models. $H \subseteq S_1 \times S_2$ is a basic mixed simulation from M_1 to M_2 if $\forall (h_1, h_2) \in H$:

(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) $\forall t_2 \in S_2 : \exists t_1 \in S_1 : (t_1, t_2) \in H \wedge R_2(h_2, t_2) \leq_i R_1(h_1, t_1)$

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. This holds dually for h_2. By requiring (t_1, t_2) to be in H we ensure that any µ-calculus property at t_2 is also less or equally informative than at t_1: either because the atomic propositions are less or equally informative or by recursion of the definition. Monotonicity ensures that the intermediate truth value, which is the combination of the transition and µ-calculus property, is also less or equally informative.

THEOREM 5.2. *Let $H \subseteq S_1 \times S_2$ be a basic mixed simulation from M_1 to M_2, and ϕ a closed μ-calculus formula. Then $\forall (H_1, H_2) \in H : \|\phi\|^{M_2}(h_2) \leq_i \|\phi\|^{M_1}(h_1)$.*

PROOF. We apply induction over the structure of the µ-calculus formula, assuming that fixed point calculations take a finite number of iterations:

— $\phi \in AP$: The inequality holds by item 1 of Def. 5.1.
— $\phi = \neg\psi$, $\phi = \psi \wedge \psi'$, or $\phi = \psi \vee \psi'$: The inequality holds by monotonicity of the operators with respect to $\leq_i$ and the induction hypothesis for ψ and ψ'.

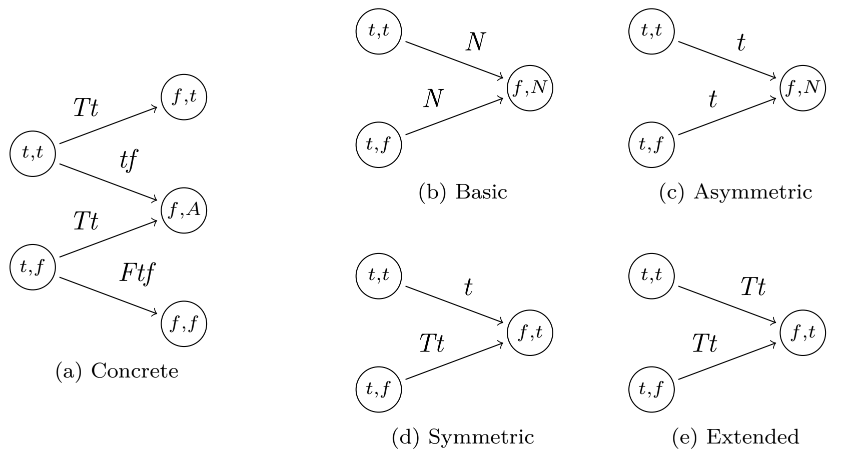


Fig. 5: Abstractions of a Kripke model

— $\phi = \Diamond\psi$: By definition of the $\Diamond$ modality, we need to prove:
$\bigvee_{t_2 \in S_2}(R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2)) \leq_i \bigvee_{t_1 \in S_1}(R_1(h_1,t_1) \wedge \|\psi\|^{M_1}(t_1))$
By Prop. 3.2, it is sufficient to show:
  — $\forall t_1 \in S_1 : \exists t_2 \in S_2 : (R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2)) \leq_i (R_1(h_1,t_1) \wedge \|\psi\|^{M_1}(t_1))$
  This holds by item 2 of Def. 5.1, the induction hypothesis for ψ because $(t_1,t_2) \in H$, and monotonicity of $\wedge$ with respect to $\leq_i$.
  — $\forall t_2 \in S_2 : \exists t_1 \in S_1 : (R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2)) \leq_i (R_1(h_1,t_1) \wedge \|\psi\|^{M_1}(t_1))$
  This holds by item 3 of Def. 5.1, the induction hypothesis for ψ because $(t_1,t_2) \in H$, and monotonicity of $\wedge$ with respect to $\leq_i$.

— $\phi = \Box\psi$: By definition of the $\Box$ modality, we need to prove:
$\bigwedge_{t_2 \in S_2}(\neg R_2(h_2,t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i \bigwedge_{t_1 \in S_1}(\neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1))$
By Prop. 3.2, it is sufficient to show:
  — $\forall t_1 \in S_1 : \exists t_2 \in S_2 : (\neg R_2(h_2,t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i (\neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1))$
  This holds by item 2 of Def. 5.1, the induction hypothesis for ψ because $(t_1,t_2) \in H$, and monotonicity of $\neg$ and $\vee$ with respect to $\leq_i$.
  — $\forall t_2 \in S_2 : \exists t_1 \in S_1 : (\neg R_2(h_2,t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i (\neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1))$
  This holds by item 3 of Def. 5.1, the induction hypothesis for ψ because $(t_1,t_2) \in H$, and monotonicity of $\neg$ and $\vee$ with respect to $\leq_i$. □

COROLLARY 5.3. *If $(s_1^0, s_2^0) \in H$ for some basic mixed simulation H, then M_2 abstracts M_1.*

Fig. 5b shows an abstraction of the concrete model in Fig. 5a using a basic mixed simulation. Its truth values are taken from the steering logic of Fig. 1c. 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. 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. 5.1 holds: we get the most informative value by calculating $t \otimes A \otimes f = N$. For the transitions we have $Tt \otimes tf \otimes Ff = N$ and $Ff \otimes Tt \otimes Ftf = N$.

5.2. Symmetric mixed simulation

Now that we have a basic definition for mixed simulation, we can weaken it to allow for better abstractions. One way to weaken the requirements of Def. 5.1 is to include the ability to add unit elements to either side of a comparison without influencing the result, as stated in Prop. 3.3. 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 5.4. Let $M_n = \langle \mathcal{B}, AP, S_n, s_n^0, R_n, \Theta_n \rangle$ for $n \in \{1, 2\}$ be multi-valued Kripke models. $H \subseteq S_1 \times S_2$ is a symmetric mixed simulation from M_1 to M_2 if $\forall (h_1, h_2) \in H$:

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

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 $\Box$ modality will be $\neg\mathit{false} \vee \phi = \mathit{true}$, which has no effect on the subsequent conjunction.

The definition of mixed simulation in [Meller et al. 2009] is identical to Def. 5.4, 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 [Meller et al. 2009].

THEOREM 5.5. *Let $H \subseteq S_1 \times S_2$ be a symmetric mixed simulation from M_1 to M_2, and ϕ a closed µ-calculus formula. Then $\forall (H_1, H_2) \in H : \|\phi\|^{M_2}(h_2) \leq_i \|\phi\|^{M_1}(h_1)$.*

PROOF. We extend on the proof of Thm. 5.2. The changes to items 2 and 3 between Def. 5.1 and Def. 5.4 can be considered separately. The change to item 2 only influences the proofs for the modalities in the µ-calculus formula in those cases when $\mathit{false} \leq_i R_1(h_1, t_1)$:

— $\phi = \Diamond\psi$: Due to the changed definition we need to revisit the case where $\forall t_1 \in S_1 : \exists t_2 \in S_2 : (R_2(h_2, t_2) \wedge \|\psi\|^{M_2}(t_2)) \leq_i (R_1(h, t_1) \wedge \|\psi\|^{M_1}(t_1))$.
By Prop. 3.3 and the definition of the $\Diamond$ modality, it is sufficient to show $\mathit{false} \leq_i (R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1))$, since *false* is the unit element of $\vee$. This holds due to monotonicity of $\wedge$ with respect to $\leq_i$, and rewriting *false* as $(\mathit{false} \wedge \|\psi\|^{M_1}(t_1))$.
— $\phi = \Box\psi$: Due to the changed definition we need to revisit the case where $\forall t_1 \in S_1 : \exists t_2 \in S_2 : (\neg R_2(h_2, t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i (\neg R_1(h_1, t_1) \vee \|\psi\|^{M_1}(t_1))$.
By Prop. 3.3 and the definition of the $\Box$ modality, it is sufficient to show $\mathit{true} \leq_i (\neg R_1(h_1, t_1) \vee \|\psi\|^{M_1}(t_1))$, since *true* is the unit element of $\wedge$. This holds due to monotonicity of $\neg$ and $\vee$ with respect to $\leq_i$, and rewriting *true* as $(\neg\mathit{false} \vee \|\psi\|^{M_1}(t_1))$.

The change to item 3 only influences the proofs for the modalities in the µ-calculus formula in those cases when $R_2(h_2, t_2) \leq_i \mathit{false}$:

— $\phi = \Diamond\psi$: Due to the changed definition we need to revisit the case where $\forall t_2 \in S_2 : \exists t_1 \in S_1 : (R_2(h_2, t_2) \wedge \|\psi\|^{M_2}(t_2)) \leq_i (R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1))$.
By Prop. 3.3 and the definition of the $\Diamond$ modality, it is sufficient to show $(R_2(h_2, t_2) \wedge$

$\|\psi\|^{M_2}(t_2)) \leq_i$ *false*, since *false* is the unit element of $\vee$. This holds due to monotonicity of $\wedge$ with respect to $\leq_i$, and rewriting *false* as $(\mathit{false} \wedge \|\psi\|^{M_2}(t_2))$.

— $\phi = \Box\psi$: Due to the changed definition we need to revisit the case where $\forall t_2 \in S_2 : \exists t_1 \in S_1 : (\neg R_2(h_2, t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i (\neg R_1(h_1, t_1) \vee \|\psi\|^{M_1}(t_1))$.
By Prop. 3.3 and the definition of the $\Box$ modality, it is sufficient to show $(\neg R_2(h_2, t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i$ *true*, since *true* is the unit element of $\wedge$. This holds due to monotonicity of $\neg$ and $\vee$ with respect to $\leq_i$, and rewriting *true* as $(\neg \mathit{false} \vee \|\psi\|^{M_2}(t_2))$. $\Box$

COROLLARY 5.6. *If $(s_1^0, s_2^0) \in H$ for some symmetric mixed simulation H, then M_2 abstracts M_1.*

We can calculate the transition values for the first transition of the symmetric mixed simulation (see Fig. 5d) 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 $t \otimes A = t$, because the lowest of the three rightmost concrete states has become reachable only by transitions larger in the information ordering than *false* and can therefore be ignored.

Would we have used the asymmetric definition of mixed simulation of [Meller et al. 2009], the result of which is shown in Fig. 5c, then we could not have ignored the *Ftf* transition. The value for q would have remained N and the transitions would respectively be valued as $Tt \otimes tf = t$ and $Tt \otimes Ftf = t$. Note that the symmetric mixed simulation is more informative than the asymmetric mixed simulation, which is more informative than the basic mixed simulation.

5.3. Extended mixed simulation

We use Prop. 4.5 to further weaken the symmetric mixed simulation of Def. 5.4, by allowing states in S_2 which are not related by the simulation to any singular state in S_1. Such a state relates to a subset of states in S_1, the members of which can be combined into a new state for the set S_1. Note that subsets are not represented by any actual state in the Kripke model: we only approximate their intermediate values for comparison with the intermediate values of states from S_2.

Definition 5.7. Let $M_n = \langle \mathcal{B}, AP, S_n, s_n^0, R_n, \Theta_n \rangle$ for $n \in \{1, 2\}$ 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 $\forall (H_1, H_2) \in H$:

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

where any capitalised subscript C_n of an operator, should be read as $c_n \in C_n$.

Prop. 3.4 allows for additional values $l \in L$ which are lower or higher in the truth ordering than respectively 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 multiset of intermediate values.

In Fig. 4 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-expansion. This operation will be pushed further down to the individual atomic propositions, causing imprecision.

For approximating abstract intermediate values we can only safely increase the information of a value; therefore we $\oplus$ the µ-expansion 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-expansion. This leaves $\otimes$ and $\oplus$ for respectively abstract transitions and concrete transitions.

Note that we can safely add the value *false* to a multiset of transitions, since *false* transitions do not influence µ-calculus properties, as explained at Def. 5.4. Comparing transitions to *false* 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 empty.

THEOREM 5.8. *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 ϕ a closed µ-calculus formula. Then $\forall (H_1, H_2) \in H : \oplus_{h_2 \in H_2}(\|\phi\|^{M_2}(h_2)) \leq_i \otimes_{h_1 \in H_1}(\|\phi\|^{M_1}(h_1))$.*

PROOF. We apply induction over the structure of the µ-calculus formula, assuming that fixed point calculations take a finite number of iterations:

— $\phi \in AP$: The inequality holds by item 1 of Definition 5.7.
— $\phi = \neg\psi$: By monotonicity of the $\neg$ operator with respect to $\leq_i$ we have
$\oplus_{h_2 \in H_2}(\|\neg\psi\|^{M_2}(h_2)) = \neg \oplus_{h_2 \in H_2}(\|\psi\|^{M_2}(h_2))$ and
$\neg \otimes_{h_1 \in H_1}(\|\psi\|^{M_1}(h_1)) = \otimes_{h_1 \in H_1}(\|\neg\psi\|^{M_1}(h_1))$. This proves the inequality when we include the induction hypothesis for ψ.
— $\phi = \psi \wedge \psi'$, or $\phi = \psi \vee \psi'$: By Proposition 4.5 we have
$\oplus_{h_2 \in H_2}(\|\psi \textcircled{?} \psi'\|^{M_2}(h_2)) \leq_i (\oplus_{h_2 \in H_2}(\|\psi\|^{M_2}(h_2))) \textcircled{?} (\oplus_{h_2 \in H_2}(\|\psi'\|^{M_2}(h_2)))$ and
$(\otimes_{h_1 \in H_1}(\|\psi\|^{M_1}(h_1))) \textcircled{?} (\otimes_{h_1 \in H_1}(\|\psi'\|^{M_1}(h_1))) \leq_i \otimes_{h_1 \in H_1}(\|\phi \textcircled{?} \phi'\|^{M_1}(h_1))$, where $\textcircled{?}$ can be substituted with either $\wedge$ or $\vee$. Monotonicity of these operators with respect to $\leq_i$ and the induction hypothesis for ψ and ψ' ensure the inequality holds.
— $\phi = \Diamond\psi$: By definition of the $\Diamond$ modality and the $\oplus$ and $\otimes$ operators, we need to prove:
$\bigvee_{t_2 \in S_2}(R_2(h_2, t_2) \wedge \|\psi\|^{M_2}(t_2)) \leq_i \bigvee_{t_1 \in S_1}(R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1))$ for all $(h_2, h_1) \in (H_2 \times H_1)$
By Proposition 3.4, it is sufficient to prove the following two inequalities:
 — $\forall t_1 \in S_1 : \exists l \in L : \exists T_2' \subseteq S_2 : l \leq_i (R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1)) \wedge l \leq_t \bigvee_{t_2' \in T_2'}(R_2(h_2, t_2') \wedge \|\psi\|^{M_2}(t_2'))$ for all $(h_2, h_1) \in (H_2 \times H_1)$
 Moving in the quantification over $(H_2 \times H_1)$ results in a stronger requirement:
 $\forall t_1 \in S_1 : \exists l \in L : \exists T_2' \subseteq S_2 : l \leq_i \otimes_{h_1 \in H_1}(R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1)) \wedge l \leq_t \oplus_{h_2 \in H_2} \bigvee_{t_2' \in T_2'}(R_2(h_2, t_2') \wedge \|\psi\|^{M_2}(t_2'))$
 Including Proposition 4.5 allows for two substitutions:
 $\otimes_{h_1 \in H_1}(R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1)) = \otimes_{h_1 \in H_1} R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1)$
 $\otimes_{t_2' \in T_2'} R_2(h_2, t_2') \wedge \oplus_{t_2' \in T_2'} \|\psi\|^{M_2}(t_2') \leq_t \bigvee_{t_2' \in T_2'}(R_2(h_2, t_2') \wedge \|\psi\|^{M_2}(t_2'))$
 By item 2 we have two possibilities. If *false* $\leq_i \otimes_{h_1 \in H_1} R_1(h_1, t_1)$ then we can choose $l =$ *false* and both inequalities hold:
 false $\leq_i \otimes_{h_1 \in H_1} R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1)$
 false $\leq_t \oplus_{h_2 \in H_2} \otimes_{t_2' \in T_2'} R_2(h_2, t_2') \wedge \oplus_{t_2' \in T_2'} \|\psi\|^{M_2}(t_2')$
 Otherwise we know $(\{t_1\}, T_2) \in H$ and
 $\oplus_{h_2 \in H_2} \otimes_{t_2 \in T_2} R_2(h_2, t_2) \leq_i \otimes_{h_1 \in H_1} R_1(h_1, t_1)$, meaning we can choose
 $l = \oplus_{h_2 \in H_2} \otimes_{t_2' \in S_2'} R_2(h_2, t_2') \wedge \oplus_{t_2' \in S_2'} \|\psi\|^{M_2}(t_2')$ and $T_2' = T_2$, ensuring both inequalities hold:
 $\oplus_{h_2 \in H_2} \otimes_{t_2' \in T_2'} R_2(h_2, t_2') \wedge \oplus_{t_2' \in T_2'} \|\psi\|^{M_2}(t_2') \leq_i \otimes_{h_1 \in H_1} R_1(h_1, t_1) \wedge \|\psi\|^{M_1}(t_1)$
 $l \leq_t l$

— $\forall t_2 \in S_2 : \exists l \in L : \exists T_1' \subseteq S_1 : (R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2)) \leq_i l \wedge l \leq_t \bigvee_{t_1' \in T_1'}(R_1(h_1,t_1') \wedge \|\psi\|^{M_1}(t_1'))$ for all $(h_2,h_1) \in (H_2 \times H_1)$
  Moving in the quantification over $(H_2 \times H_1)$ results in a stronger requirement:
  $\forall t_2 \in S_2 : \exists l \in L : \exists T_1' \subseteq S_1 : \oplus_{h_2 \in H_2}(R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2)) \leq_i l \wedge l \leq_t \otimes_{h_1 \in H_1} \bigvee_{t_1' \in T_1'}(R_1(h_1,t_1') \wedge \|\psi\|^{M_1}(t_1'))$
  Including Proposition 4.5 allows for two substitutions:
  $\oplus_{h_2 \in H_2}(R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2)) = \oplus_{h_2 \in H_2} R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2)$
  $\oplus_{t_1' \in T_1'} R_1(h_1,t_1') \wedge \otimes_{t_1' \in T_1'} \|\psi\|^{M_1}(t_1') \leq_t \bigvee_{t_1' \in T_1'}(R_1(h_1,t_1') \wedge \|\psi\|^{M_1}(t_1'))$
  By item 3 we have two possibilities. If $\oplus_{h_2 \in H_2} R_2(h_2,t_2) \leq_i \mathit{false}$ then we can choose $l = \mathit{false}$ and both inequalities hold:
  $\oplus_{h_2 \in H_2} R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2) \leq_i \mathit{false}$
  $\mathit{false} \leq_t \otimes_{h_1 \in H_1} \oplus_{t_1' \in S_1'} R_1(h_1,t_1') \wedge \otimes_{t_1' \in S_1'} \|\psi\|^{M_1}(t_1')$
  Otherwise we know $(T_1, \{t_2\}) \in H$ and
  $\oplus_{h_2 \in H_2} R_2(h_2,t_2) \leq_i \otimes_{h_1 \in H_1} \oplus_{t_1 \in T_1} R_1(h_1,t_1)$, meaning we can choose $l = \otimes_{h_1 \in H_1} \oplus_{t_1' \in S_1'} R_1(h_1,t_1') \wedge \otimes_{t_1' \in S_1'} \|\psi\|^{M_1}(t_1')$ and $T_1' = T_1$, ensuring both inequalities hold:
  $\oplus_{h_2 \in H_2} R_2(h_2,t_2) \wedge \|\psi\|^{M_2}(t_2) \leq_i \otimes_{h_1 \in H_1} \oplus_{t_1' \in T_1'} R_1(h_1,t_1') \wedge \otimes_{t_1' \in T_1'} \|\psi\|^{M_1}(t_1')$
  $l \leq_t l$

— $\phi = \Box\psi$: By definition of the $\Box$ modality and the $\oplus$ and $\otimes$ operators, we need to prove:
$\bigwedge_{t_2 \in S_2}(\neg R_2(h_2,t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i \bigwedge_{t_1 \in S_1}(\neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1))$ for $\forall (h_2,h_1) \in (H_2 \times H_1)$
By Proposition 3.4, it is sufficient to prove the following two inequalities:

— $\forall t_1 \in S_1 : \exists l \in L : \exists T_2' \subseteq S_2 : l \leq_i (\neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1)) \wedge \bigwedge_{t_2' \in T_2'}(\neg R_2(h_2,t_2') \vee \|\psi\|^{M_2}(t_2')) \leq_t l$ for all $(h_2,h_1) \in (H_2 \times H_1)$
  Moving in the quantification over $(H_2 \times H_1)$ results in a stronger requirement:
  $\forall t_1 \in S_1 : \exists l \in L : \exists T_2' \subseteq S_2 : l \leq_i \otimes_{h_1 \in H_1}(\neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1)) \wedge \oplus_{h_2 \in H_2} \bigwedge_{t_2' \in T_2'}(\neg R_2(h_2,t_2') \vee \|\psi\|^{M_2}(t_2')) \leq_t l$
  Including Proposition 4.5 allows for two substitutions:
  $\otimes_{h_1 \in H_1}(\neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1)) = \otimes_{h_1 \in H_1} \neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1)$
  $\bigwedge_{t_2' \in T_2'}(\neg R_2(h_2,t_2') \vee \|\psi\|^{M_2}(t_2')) \leq_t \otimes_{t_2' \in T_2'} \neg R_2(h_2,t_2') \vee \oplus_{t_2' \in T_2'} \|\psi\|^{M_2}(t_2'))$
  By item 2 we have two possibilities. If $\mathit{false} \leq_i \otimes_{h_1 \in H_1} R_1(h_1,t_1)$ then we can choose $l = \mathit{true}$ and both inequalities hold:
  $\mathit{true} \leq_i \otimes_{h_1 \in H_1} \neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1)$
  $\oplus_{h_2 \in H_2} \otimes_{t_2' \in T_2'} \neg R_2(h_2,t_2') \vee \oplus_{t_2' \in T_2'} \|\psi\|^{M_2}(t_2')) \leq_t \mathit{true}$
  Otherwise we know $(\{t_1\}, T_2) \in H$ and
  $\oplus_{h_2 \in H_2} \otimes_{t_2 \in T_2} R_2(h_2,t_2) \leq_i \otimes_{h_1 \in H_1} R_1(h_1,t_1)$, meaning we can choose $l = \oplus_{h_2 \in H_2} \otimes_{t_2' \in T_2'} \neg R_2(h_2,t_2') \vee \oplus_{t_2' \in T_2'} \|\psi\|^{M_2}(t_2'))$ and $T_2' = T_2$, ensuring both inequalities hold:
  $\oplus_{h_2 \in H_2} \otimes_{t_2' \in T_2'} \neg R_2(h_2,t_2') \vee \oplus_{t_2' \in T_2'} \|\psi\|^{M_2}(t_2')) \leq_i \otimes_{h_1 \in H_1} \neg R_1(h_1,t_1) \vee \|\psi\|^{M_1}(t_1)$
  $l \leq_t l$

— $\forall t_2 \in S_2 : \exists l \in L : \exists T_1' \subseteq S_1 : (\neg R_2(h_2,t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i l \wedge \bigwedge_{t_1' \in T_1'}(\neg R_1(h_1,t_1') \vee \|\psi\|^{M_1}(t_1')) \leq_t l$ for all $(h_2,h_1) \in (H_2 \times H_1)$
  Moving in the quantification over $(H_2 \times H_1)$ results in a stronger requirement:

$\forall t_2 \in S_2 : \exists l \in L : \exists T_1' \subseteq S_1 : \oplus_{h_2 \in H_2}(\neg R_2(h_2, t_2) \vee \|\psi\|^{M_2}(t_2)) \leq_i$
$l \wedge \otimes_{h_1 \in H_1} \bigwedge_{t_1' \in T_1'}(\neg R_1(h_1, t_1') \vee \|\psi\|^{M_1}(t_1')) \leq_t l$
Including Proposition 4.5 allows for two substitutions:
$\oplus_{h_2 \in H_2}(\neg R_2(h_2, t_2) \vee \|\psi\|^{M_2}(t_2)) = \oplus_{h_2 \in H_2} \neg R_2(h_2, t_2) \vee \|\psi\|^{M_2}(t_2)$
$\bigwedge_{t_1' \in T_1'}(\neg R_1(h_1, t_1') \vee \|\psi\|^{M_1}(t_1')) \leq_t \oplus_{t_1' \in T_1'} \neg R_1(h_1, t_1') \vee \otimes_{t_1' \in T_1'} \|\psi\|^{M_1}(t_1'))$
By item 3 we have two possibilities. If $\oplus_{h_2 \in H_2} R_2(h_2, t_2) \leq_i \mathit{false}$ then we can choose $l = \mathit{true}$ and both inequalities hold:
$\oplus_{h_2 \in H_2} \neg R_2(h_2, t_2) \vee \|\psi\|^{M_2}(t_2) \leq_i \mathit{true}$
$\otimes_{h_1 \in H_1} \oplus_{t_1' \in T_1'} \neg R_1(h_1, t_1') \vee \otimes_{t_1' \in T_1'} \|\psi\|^{M_1}(t_1')) \leq_t \mathit{true}$
Otherwise we know $(T_1, \{t_2\}) \in H$ and
$\oplus_{h_2 \in H_2} R_2(h_2, t_2) \leq_i \otimes_{h_1 \in H_1} \oplus_{t_1 \in T_1} R_1(h_1, t_1)$, meaning we can choose $l = \otimes_{h_1 \in H_1} \oplus_{t_1' \in T_1'} \neg R_1(h_1, t_1') \vee \otimes_{t_1' \in T_1'} \|\psi\|^{M_1}(t_1'))$ and $T_1' = T_1$, ensuring both inequalities hold:
$\oplus_{h_2 \in H_2} \neg R_2(h_2, t_2) \vee \|\psi\|^{M_2}(t_2) \leq_i \otimes_{h_1 \in H_1} \oplus_{t_1' \in T_1'} \neg R_1(h_1, t_1') \vee \otimes_{t_1' \in T_1'} \|\psi\|^{M_1}(t_1'))$
$l \leq_t l$ □

COROLLARY 5.9. *If $(\{s_0^1\}, \{s_0^2\}) \in H$ for some extended mixed simulation H, then M_2 abstracts M_1.*

The result of an extended mixed simulation is shown in Fig. 5e. We can increase precision over the symmetric simulation of Fig. 5d because the most informative value for the first transition is equal to $Tt \oplus tf = Ttf$. This would satisfy item 3 of Def. 5.7, but by choosing Tt for the transition we ensure that the opposite direction described by item 3 is also satisfied.

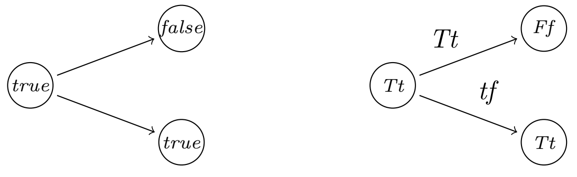
The full extended simulation relation between Fig. 5a and Fig. 5e then becomes: $\{(\{tt\}, \{tt\}), (\{tf\}, \{tf\}), (\{ft\}, \{ft\}), (\{fA\}, \{ft\}), (\{ff\}, \{\})\}$. The propositions p and q are used to identify states as pq. The concrete source states tt and tf are related to their abstract counterparts tt and tf. Concrete states ft and fA are each related to the abstract state ft, which is possible due to the abstract transition to this state being the disjunction $Tt \vee tf = Tt$ instead of the $Tt \oplus tf = Ttf$ mentioned earlier. (Note that $Tt \leq_i Ttf$ by construction.) The state ff can be safely dropped by relating it to the empty set, since $\mathit{false} \leq_i Ftf$ for its transition.

We could add $(\{ft, fA\}, \{ft\})$ to the relation, but this is not necessary since we used disjunction to calculate the transition values. It would express how the abstract state ft is intended as an equivalence class containing the concrete ft and fA. (Note that ff has been explicitly dropped.) This is required when we use $\oplus$ instead of $\vee$ to calculate the transition value in accordance with item 3 of Def. 5.7. In addition it would entail the concrete ft and fA relating to other abstract states for item 2 of Def. 5.7: states that are not available in the current abstraction. We will use this type of construction in section 7 to increase precision.

Determining whether two given models are in an abstraction relation becomes more difficult when using extended mixed simulation due to its weaker requirements. The ability to relate states to sets of states increases the amount of possibilities for finding a relation. Note however that this cost is offset by the ability to relate more models. In the context of creating an abstract model from a given concrete model the weaker requirements are not an issue; on the contrary, they allow for increased precision. We can ensure beforehand that the algorithm creates models that are in a extended mixed simulation with the original without increasing complexity (as evidenced by the algorithm presented in section 7).

6. 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



(a) Boolean logic (b) Steering logic

Fig. 6: Steerability example

precision and reducing the size of the abstraction. The result can be seen in the sequence of abstractions of Fig. 5: 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 take a closer look at the nine-valued bilattice of Fig. 1c. 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 as values indicating the steerability of transitions in a Kripke model.

We first look at the example Kripke model of Fig. 6a. In this model, using classical Boolean logic, we keep track of a proposition p indicated by the value in each state. This proposition should never become *false*. It is clear that this model fails the requirement because there is a path to a state where the proposition is *false*, such that the requirement is *false*. We get this value by evaluating the property $\Box p = (\neg \mathit{true} \vee \mathit{true}) \wedge (\neg \mathit{true} \vee \mathit{false}) = \mathit{false}$.

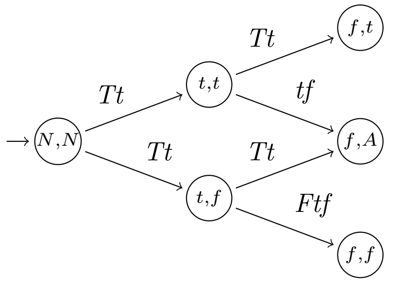
But what if we could control the execution as indicated by the model in Fig. 6b. In this model, using steering logic, we have the same values for the propositions, but a different value for one of the transitions. (In steering logic we have $Tt = \mathit{true}$ and $Ff = \mathit{false}$ by definition.) The transition with the value *tf* indicates that this transition can be steered: it can be turned on or off to influence the execution. If we evaluate the property in this model, we get $\Box p = (\neg Tt \vee Tt) \wedge (\neg tf \vee Ff) = tf$. This indicates that the requirement can be made to either hold or fail by steering the execution.

Truth values of steering logic are effectively subsets of $\{t, f, T, F\}$ with N the empty set, and A 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.

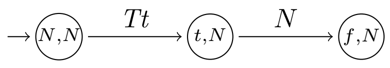
The intuition of the individual values is that T indicates a transition that is enabled by default: during execution it can be non-deterministically chosen to further the execution. A value t indicates a transition that can be enabled when controlling the execution: if we want this transition to be considered,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

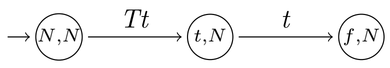
To indicate a steerable transition we use the values tf, Ttf, and Ftf. They respectively indicate a transition that: can be enabled or disabled when controlled; is enabled by default, but can be disabled when controlled; and is disabled by default, but can be enabled when



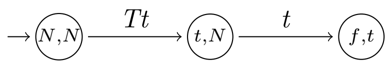
(a) Concrete



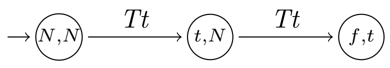
(b) Basic



(c) Asymmetric



(d) Symmetric



(e) Extended

Fig. 7: Steerability abstraction

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 concrete model in Fig. 5a, 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 $\Box(p = t \wedge \Diamond(p = f))$ 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 $(Tt \wedge Tt) \vee (tf \vee Tt) = Tt$ and $(Tt \wedge Tt) \vee (Ftf \vee Tt) = Tt$; then we use these values to calculate the $\Box$ modality and get the final result $(\neg Tt \vee (Tt \wedge Tt)) \wedge (\neg Tt \vee (Tt \wedge Tt)) = Tt$.

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. 5 where all states with $p = f$ are combined, while Fig. 7 shows the result of combining states with $p = t$.

Evaluating the property $\Box(p = t \wedge \Diamond(p = f))$ in these abstract models gives different results based on the abstraction requirements used. In a basic mixed simulation we get $\neg Tt \vee (Tt \wedge (N \wedge Tt)) = N$, indicating we have no information to determine whether the property holds. In the asymmetric and symmetric mixed simulation we get $\neg Tt \vee (Tt \wedge (t \wedge Tt)) = t$, indicating that we can at least steer the execution to ensure the property will hold. Finally, in the extended simulation we get $\neg Tt \vee (Tt \wedge (Tt \wedge Tt)) = Tt$, 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 property.

7. ABSTRACTION RULES

For any two Kripke models, the abstraction requirements of extended mixed simulation can verify whether one model is a correct abstraction of the other. But given only one concrete model, how do we construct an abstract model such that these models are in an extended mixed simulation? In this section two methods are presented for calculating transition values when doing abstraction based on equivalence classes. We want to create an abstract state for each set of concrete states in an equivalence class, such that in the extended mixed simulation $H \subseteq \mathcal{P}(S_1) \times \{S_2\}$ all simulations are between sets of concrete states and singleton abstract states.

7.1. Atomic propositions

Given a concrete Kripke model $M_1 = \left\langle \mathcal{B}, AP, S_1, s_1^0, R_1, \Theta_1 \right\rangle$ with equivalence classes defined over S_1, we want to lift the concrete transition relation R_1 to an abstract transition relation R_2 between the equivalence classes. The equivalence classes will form the states S_2 of the abstract Kripke model $M_2 = \left\langle \mathcal{B}, AP, S_2, s_2^0, R_2, \Theta_2 \right\rangle$. We create an abstract state $h_2 \in S_2$ for each equivalence class $H_1 \subseteq S_1$ of the concrete model. To ensure the requirement in item 1 of Def. 5.7 for extended simulation, the atomic propositions of the concrete states in the equivalence class H_1 are combined using the $\otimes$ operator and define the abstract labelling function: $\Theta_2(a)(h_2) := \otimes_{h_1 \in H_1} \Theta(a)(h_1)$.

7.2. Transitions

When creating the abstract transition relation R_2 we want to ensure that both item 2 and 3 of Def. 5.7 hold. This is done in two parts: we first combine all transitions going to the same target equivalence class, and then we combine the resulting transitions if they are from the same source equivalence class. Stated more briefly, we first combine target states, and then we combine source states. The resulting transitions define the abstract transition relation R_2 of the abstract Kripke model.

Simply following item 3 of Def. 5.7 would suggest that we use the $\oplus$ operator when combining target states, and the $\otimes$ operator when combining source states. With T_1 the equivalence class of the abstract target state, and H_1 the equivalence class of the abstract source state, that would give $R_2(h_2, t_2) := \otimes_{h_1 \in H_1}(\oplus_{t_1 \in T_1} R_1(h_1, t_1) \oplus \mathit{false})$, with h_2 the abstract state for H_1 and t_2 the abstract state for T_1. However, while this fulfils the requirements of item 3, it does not guarantee that item 2 of the definition will hold.

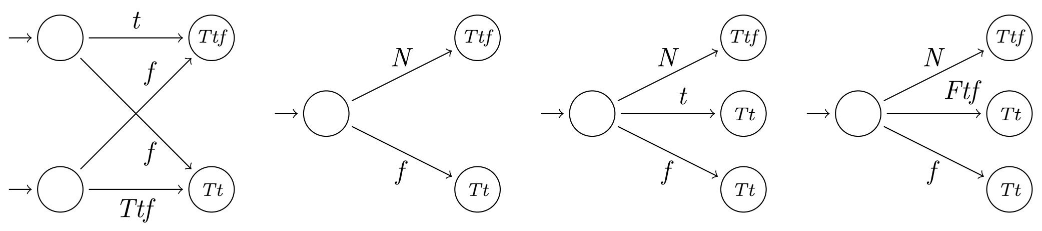
The problem is that the $\oplus$ of a multiset, even when combined with *false* using the $\otimes$ operator, is not necessarily smaller or equal in the information ordering than all elements in the multiset. This is a requirement by item 2 of Def. 5.7 for each concrete transition from a concrete state, and is ideally fulfilled by abstract transitions from the corresponding abstract state. On the contrary, it is possible that the $\oplus$ of the multiset is larger in the information ordering for most if not all elements in the multiset.

It is however the case that the $\vee$ of a multiset, when combined with *false* using the $\otimes$ operator, is always smaller or equal in the information ordering than all elements in the multiset. In addition, the $\vee$ of a multiset is lower or equal in the information ordering than the $\oplus$ of the same multiset, in accordance with item 3 of Def. 5.7. Defining the abstract transition relation as $R_2(h_2, t_2) := \otimes_{h_1 \in H_1}(\bigvee_{t_1 \in T_1} R_1(h_1, t_1))$, with h_2 the abstract state for H_1 and t_2 the abstract state for T_1, satisfies item 2 and 3 of Def. 5.7.

7.3. Increasing precision

The previous subsection suggests that there is no use for the $\oplus$ operator in constructing abstract transitions. This is not true, as it can be used to increase precision. It requires the construction of abstract states for sets of equivalence classes in addition to abstract states for individual equivalence classes. Such a construction is useful to express that while it is unclear whether individual abstract states are reachable, we do know that at least one in a group of abstract states is reachable. In general, we can increase the information on a transition, by lowering the information of the abstract target state.

We start the abstraction using the $\vee$ operator to abstract individual equivalence classes: $R_2(h_2, t_2) := \otimes_{h_1 \in H_1}(\bigvee_{t_1 \in T_1} R_1(h_1, t_1))$. This ensures that item 2 of Def. 5.7 is ful-



(a) Concrete (b) Base (c) $\vee$ Operator (d) $\oplus$ Operator

Fig. 8: Increasing precision

filled: each concrete transition has a matching abstract transition. Then we are free to add abstract transitions to increase precision using the $\oplus$ operator: $R_2(h_2, t_2) := \otimes_{h_1 \in H_1}(\oplus_{t_1 \in T_1} R_1(h_1, t_1) \oplus \mathit{false})$. In both cases item 3 of Def. 5.7 holds by construction.

Note that using the $\oplus$ operator for combining multiple abstract states, always gives an equally or more informative answer than using the $\vee$ operator. This follows from $\bigvee_{t_1 \in T_1} R_1(h_1, t_1) \leq_i \oplus_{t_1 \in T_1} R_1(h_1, t_1)$ holding for any h_1. While this proves that using the $\oplus$ operator will not cost precision, it does not guarantee it can increase precision. In the next section we show an example where $\oplus$ actually improves the precision of the abstraction.

7.4. Example

In Fig. 8a we have an example Kripke model for which we need to combine the two concrete initial states (indicated by small incoming arrows) into a single abstract initial state. Transitions are labeled with their values, and the target states contain the value of the atomic proposition p. As will be shown, combining source states with the $\otimes$ operator can cause a loss of information, but this loss can be minimised by creating more informative transitions for combined abstract target states.

In the base case of Fig. 8b the transitions are simply combined using the $\otimes$ operator, giving $t \otimes f = N$ and $f \otimes \mathit{Ttf} = f$. In the next two figures, we add an extra abstract state, which is the combination of the two original target states. Its value for p is the combination of their values for p: $\mathit{Ttf} \otimes \mathit{Tt} = \mathit{Tt}$. The transition to this extra state in Fig. 8c is calculated using the $\vee$ operator as $(t \vee f) \otimes (f \vee \mathit{Ttf}) = t$, while in Fig. 8d it is calculated using the $\oplus$ operator as $(t \oplus f \oplus \mathit{Ff}) \otimes (f \oplus \mathit{Ttf} \oplus \mathit{Ff}) = \mathit{Ftf}$. Note that the additional Ff when using the $\oplus$ operator corresponds to the additional *false* value present in item 3 of Def. 5.7 and can be omitted when using the $\vee$ operator.

If we now calculate the property $\|\Diamond p\|(s) = \bigvee_{t \in S}(R(s,t) \wedge \|p\|(t))$ at the initial state of each abstract Kripke model, then we will see an increase in precision for each successive abstraction. In Fig. 8b we have $N \wedge \mathit{Ttf} = f$ and $f \wedge \mathit{Tt} = f$ as intermediate truth values. For the additional transition in Fig. 8c and 8d we have respectively $t \wedge \mathit{Tt} = t$ and $\mathit{Ftf} \wedge \mathit{Tt} = \mathit{Ftf}$. This gives $f \vee f = f$ as a final result for the base case, $f \vee f \vee t = t$ for the $\vee$ operator, and $f \vee f \vee \mathit{Ftf} = \mathit{tf}$ for the $\oplus$ operator. Calculating the same property for the concrete model gives $(t \vee \mathit{Ttf}) \vee (f \wedge \mathit{Tt}) = \mathit{tf}$ and $(f \wedge \mathit{Ttf}) \vee (\mathit{Ttf} \wedge \mathit{Tt}) = \mathit{Ttf}$ for the initial states; since both initial states are valid starting points, the property in the concrete model gets the value $\mathit{tf} \vee \mathit{Ttf} = \mathit{Ttf}$. Using the $\oplus$ operator gives the most informative answer, which is closest in information to the concrete result.

7.5. Overview

To end this section we summarise the presented construction in a theorem. The constructed model is in an extended mixed simulation with the original model. We note that this theorem does not hold for the other notions of mixed simulation as discussed in section 5.

THEOREM 7.1. *Given a Kripke model $M_1 = \langle \mathcal{B}, AP, S_1, s_1^0, R_1, \Theta_1 \rangle$, with a partitioning $P = \{P_1, ..., P_n\}$ of S_1 defining the equivalence classes of M_1. We can construct a Kripke model $M_2 = \langle \mathcal{B}, AP, S_2, s_2^0, R_2, \Theta_2 \rangle$, such that M_2 abstracts M_1, using the following rules:*

(1) $S_2 := \mathcal{P}(P)$
(2) $s_2^0 := \{P_k\}$ *with k such that* $s_1^0 \in P_k$
(3) $R_2(s_2, t_2) := \otimes_{s_1 \in \cup s_2}(\bigvee_{t_1 \in \cup t_2} R_1(s_1, t_1))$ *when t_2 is a singleton*
(4) $R_2(s_2, t_2) := \otimes_{s_1 \in \cup s_2}(\oplus_{t_1 \in \cup t_2} R_1(s_1, t_1) \oplus \mathit{false})$ *when t_2 is not a singleton.*
(5) $\Theta_2(a)(s_2) := \otimes_{s_1 \in \cup s_2} \Theta(a)(s_1)$

PROOF. Follows by construction from Def. 5.7. We create an extended mixed simulation that relates each $s_1 \in S_1$ to the singleton $s_2 \in S_2$ such that $s_1 \in \cup s_2$. That is, s_1 must be in the single partition contained by s_2. In addition we relate each state $s_2 \in S_2$ that is not a singleton to the set $\cup s_2$. That is, we take the union of the multiple partitions contained in s_2. Item 1 of Def. 5.7 then holds by item 5 of the theorem, item 2 of Def. 5.7 holds by item 3, and item 3 of Def. 5.7 holds by item 3 of the theorem if $t_2 \in S_2$ is a singleton, or by item 4 of the theorem if $t_2 \in S_2$ is not a singleton. This extended mixed simulation together with item 2 of the theorem ensure that Cor. 5.9 holds and we have that M_2 abstracts M_1. $\square$

This construction uses a power set of the partitioning for the abstract Kripke model, which might suggest that the abstract model has more states than the original concrete model. Note however, that a useful partitioning in equivalence classes compensates for this by ensuring that $|\mathcal{P}(P)| < |S_1|$. The cost of the additional states over the $|P|$ states of a more traditional abstraction is offset by the increase in precision. In addition, any transition from a state $s \in S_2$ to a non-singleton state $T = \{P_j, ..., P_{j+k}\}$ such that $\exists t \in T : (R_2(s, T) \leq_i R_2(s, t)) \wedge (\forall a \in AP : \Theta_2(a)(T) \leq_i \Theta_2(a)(t))$ can be safely removed since it does not add to the precision of the model. This can make useless states unreachable or will at least reduce the amount of transitions in the Kripke model.

When model checking the abstract Kripke model, we can generally be agnostic to the power set construction used to create it. As far as the model checker is concerned it is a plain Kripke model, be it with additional states to increase precision. Alternatively we could modify the model checking algorithm to make it aware of the power set construction. That would allow it to reuse results obtained from singleton target states to calculate results for non singleton target states without having to explore them. This tradeoff between complexity of the algorithm and the size of the search space is left for future research.

8. CONCLUSION

We have shown how the definition of mixed simulation can be modified up to 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 [Meller et al. 2009] with the abstraction technique used in [Gurfinkel and Chechik 2006]. In addition we presented a set of abstraction rules for constructing an abstract Kripke model from a given concrete Kripke model. Following these rules ensures the two models are in an extended mixed simulation.

We conjecture that extended mixed simulation might be the most exact simulation for two models that are in an abstraction relation. I.e., if two multi-valued Kripke models are in an abstraction relation as defined in Def. 2.10, then there exists an extended mixed simulation between these models. We have yet to find a counter-example of this statement, but this might be possible due to the approximations used for the lattice operations over multisets of intermediate truth values.

Since the theory presented in this paper is generic and works for any distributive bilattice, it can be applied to develop new abstraction techniques, based on new multi-valued logics. More specifically, we intend to implement the presented abstraction rules in the SpinJa model checker [de Jonge and Ruys 2010; Vijzelaar et al. 2011]. This would include our recently submitted research on encoding multi-valued temporal properties in so-called Büchi automata [Vijzelaar and Fokkink 2016]. Together with the steering logic mentioned earlier in this paper, this will allow us to investigate execution steering based on abstract models [Vijzelaar et al. 2014].

References

N.D. Belnap. 1977. *Modern Uses of Multiple-Valued Logics.* Reidel, Chapter A Useful Four-Valued Logic, 5–37.

A. Bialynicki-Birula and H. Rasiowa. 1957. On the representation of quasi-Boolean algebras. Bulletin de l'Académie Polonaise des Sciences. (1957).

G. Bruns and P. Godefroid. 1999. Model Checking Partial State Spaces with 3-Valued Temporal Logics. In *CAV (LNCS)*, Vol. 1633. Springer, 274–287.

M. Chechik, B. Devereux, S.M. Easterbrook, and A. Gurfinkel. 2003. Multi-valued symbolic model-checking. *ACM TOSEM* 12, 4 (2003), 371–408.

E.M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. 2000. Counterexample-Guided Abstraction Refinement. In *CAV (LNCS)*, Vol. 1855. Springer, 154–169.

D. Dams, R. Gerth, and O. Grumberg. 1997. Abstract Interpretation of Reactive Systems. *ACM Trans. Program. Lang. Syst.* 19, 2 (1997), 253–291.

M. de Jonge and T.C. Ruys. 2010. The SpinJa Model Checker. In *SPIN (LNCS)*, Vol. 6349. Springer, 124–128.

M. Fitting. 1989. Bilattices and the theory of truth. *Journal of Philosophical Logic* 18 (1989), 225–256. Issue 3.

O. Grumberg. 2005. Abstraction and Refinement in Model Checking. In *FMCO (LNCS)*, Vol. 4111. Springer, 219–242.

A. Gurfinkel and M. Chechik. 2003. Multi-Valued Model Checking via Classical Model Checking. In *CONCUR (LNCS)*, Vol. 2761. Springer, 263–277.

A. Gurfinkel and M. Chechik. 2006. Why Waste a Perfectly Good Abstraction. In *TACAS (LNCS)*, Vol. 3920. Springer, 212–226.

A. Gurfinkel, O. Wei, and M. Chechik. 2006. Yasm: A Software Model-Checker for Verification and Refutation. In *CAV (LNCS)*, Vol. 4144. Springer, 170–174.

M. Huth, R. Jagadeesan, and D.A. Schmidt. 2001. Modal Transition Systems: A Foundation for Three-Valued Program Analysis. In *ESOP (LNCS)*, Vol. 2028. Springer, 155–169.

B. Konikowska and W. Penczek. 2002. Reducing Model Checking from Multi-valued CTL* to CTL*. In *CONCUR (LNCS)*, Vol. 2421. Springer, 226–239.

D. Kozen. 1983. Results on the Propositional mu-Calculus. *Theor. Comput. Sci.* 27 (1983), 333–354.

S A Kripke. 1963. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica* 16 (1963).

Y. Meller, O. Grumberg, and S. Shoham. 2009. A Framework for Compositional Verification of Multi-valued Systems via Abstraction-Refinement. In *ATVA (LNCS)*, Vol. 5799. Springer, 271–288.

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

Y. Shramko, J.M. Dunn, and T. Takenaka. 2001. The Trilattice of Constructive Truth Values. *J. Log. Comput.* 11, 6 (2001), 761–788.

S. Vijzelaar and W. Fokkink. 2015. Multi-valued Abstraction Using Lattice Operations. In *ACSD*. IEEE Computer Society, 70–79.

S. Vijzelaar and W. Fokkink. Submitted 2016. Creating Büchi Automata for Multi-Valued Model Checking.

S. Vijzelaar, K. Verstoep, W. Fokkink, and H. E. Bal. 2011. Distributed MAP in the SpinJa Model Checker. In *PDMC (EPTCS)*, Vol. 72. 84–90.

S. Vijzelaar, K. Verstoep, W. Fokkink, and H. E. Bal. 2014. Bonsai: Cutting Models Down to Size. In *PSI (LNCS)*, Vol. 8974. Springer, 361–375.