Available online at www.sciencedirect.com
SCIENCE DIRECT®

Electronic Notes in Theoretical Computer Science 73 (2004) 45–85

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Consistent Partial Model Checking

Michael Huth[1]

Department of Computing
Imperial College London
London, England

Shekhar Pradhan[2]

Computer Science Department
Vassar College
Poughkeepsie, New York

Abstract

We propose assertion-consistency (AC) semi-lattices as suitable orders for the analysis of partial models. Such orders express semantic entailment, multiple-viewpoint and multiple-*valued* analysis, maintain internal consistency of reasoning, and subsume finite De Morgan lattices. We classify those orders that are finite and distributive and apply them to design an efficient algorithm for multiple-viewpoint checking, where checks are delegated to single-viewpoint models — efficiently driven by the order structure. Instrumentations of this algorithm enable the detection and location of inconsistencies across viewpoint boundaries. To validate the approach, we investigate multiple-valued models and their compositional property semantics over a finite distributive AC lattice. We prove that this semantics is computed by our algorithm above whenever the primes of the AC lattice determine 'projected' single viewpoints and the order between primes is preserved as refinements of single-viewpoint models. As a case study, we discuss a multiple-valued notion of state-machines with first-order logic plus transitive closure.

Keywords: lattices, DeMorgan lattices, model checking, viewpoints, requirements, refinement, verification

1 Introduction

In computer science and computer engineering, models are a popular tool for documenting, specifying or analyzing relevant aspects of a design. As

[1] Email: M.Huth@doc.imperial.ac.uk
[2] Email: pradhan@cs.vasser.edu

doi:10.1016/j.entcs.2004.08.003

representative examples we cite state-machines in UML and class diagrams annotated with constraints written in OCL. Ideally, such models are partial: although some aspects of the model are fully described and have to be realized, other aspects of the model are not fully specified and their realization is up to the implementor. Example ontologies that make this distinction between guaranteed and possible aspects explicit are modal transition systems [26,25] and models written in the executable constraint language Alloy [22]. Conceptually, such models are partial in that they allow a whole set of qualitatively different implementations. In practice, partial models also occur by abstracting an existing implementation — a step necessary due to the complexity of implementations — prior to its analysis. In the ongoing effort of automatic verification of software, guaranteed and possible aspects of the abstract model are computed as `must`- and `may`-transitions (respectively); see e.g. [31].

It is beneficial to distill a reference ontology for the many material ontologies of partial models and their analysis, be they modal transition systems, Alloy programs, partial Kripke structures [5,6], modal shape graphs [32,19], labeled transition systems with a divergence predicate [35] etc. Our reference ontology has a class of partial models $\mathcal{P}$ and a class of total models $\mathcal{T}$ such that every total model may be seen as a partial model

$$\mathcal{T} \subseteq \mathcal{P} \tag{1}$$

where the identification of a total as a partial model may involve some casting. There is an implementation relation

$$\texttt{impl} \subseteq \mathcal{P} \times \mathcal{T} \tag{2}$$

which is reflexive on $\mathcal{T}$ since implementations implement themselves. We have a query language $\mathcal{L}$ that supports negation $\neg$ and conjunction $\wedge$, and has a satisfaction relation

$$\models \ \subseteq \mathcal{T} \times \mathcal{L} \tag{3}$$

over total models such that all total models are consistent:

$$\forall T \in \mathcal{T}\ \forall \phi \in \mathcal{L}: \qquad T \not\models \phi \wedge \neg\phi\,. \tag{4}$$

Since partial models have a, possibly empty, set of total models as implementations, we can lift $\models$ to partial models in two generally distinct ways:

$$P \models^{\mathrm{a}} \phi \qquad \text{iff} \qquad \forall P' \in \mathcal{P}: \qquad (P, P') \in \texttt{impl} \Rightarrow P' \models \phi \tag{5}$$

$$P \models^{\mathrm{c}} \phi \qquad \text{iff} \qquad \exists P' \in \mathcal{P}: \qquad (P, P') \in \texttt{impl}\ \&\ P' \models \phi\,. \tag{6}$$

The relation $\models^{\mathrm{a}}$ views ϕ as an *assertion* and checks its validity with respect to all implementations of a model. Dually, $\models^{\mathrm{c}}$ views ϕ as a *consistency check* and analyzes whether ϕ is consistent with respect to a model's set of implementations. We assume

(7) $T \models \neg\phi$ iff $T \not\models \phi$
(8) $T \models \phi_1 \wedge \phi_2$ iff $T \models \phi_1$ and $T \models \phi_2$

for all $T \in \mathcal{T}$; $\phi, \phi_1, \phi_2 \in \mathcal{L}$. But then

(9) $$P \not\models^{a} \phi \qquad \text{iff} \qquad P \models^{c} \neg\phi$$

follows for all $P \in \mathcal{P}$. The semantics of $\models$ ought to be consistent with those of $\models^{a}$ and $\models^{c}$. Since `impl` is reflexive on T

$$T \models \phi \quad \Rightarrow \quad T \models^{c} \phi$$
$$T \models \neg\phi \quad \Rightarrow \quad T \not\models^{a} \phi$$

shows such consistency. It is desirable that the converse implications hold as well. If so, we compute

(10) $$T \models^{c} \phi \text{ iff } T \models \phi \text{ iff } T \not\models \neg\phi \text{ by (7) iff "not } T \not\models^{a} \phi\text{" iff } T \models^{a} \phi \;.$$

The validity of these converse relationships is secured by the more stringent demand that implementations that implement each other be indistinguishable by the query logic: for all $T, T' \in \mathcal{T}$,

(11) $$(T, T') \in \texttt{impl} \quad \Rightarrow \quad \{\phi \in \mathcal{L} \mid T \models \phi\} = \{\phi \in \mathcal{L} \mid T' \models \phi\}\,.$$

A partial model P is *consistent* iff there is some P' with $(P, P') \in$ `impl`.

We discuss a material ontology for this reference ontology in some detail in the next section. This paper investigates how well understood concepts from total model checking transfer to this partial setting. Specifically, we study

- model theory and semantic entailment,
- abstraction and refinement of models,
- reasoning about models under multiple points of view,
- more efficient multiple-*valued* model checking, and
- the consistency of $\models^{c}$ and $\models^{a}$ at the meta-level

in this setting of partial model checking. To that end, we distill an ordered structure, assertion-consistency (AC) semi-lattices, that allow the redevelopment of the notions above for partial models.

Throughout this paper, we work with the reference ontology

(12) $$(\mathcal{P}, \mathcal{T}, \mathcal{L}, \models, \texttt{impl})$$

satisfying (1-11), unless indicated otherwise. Such an ontology has a refinement relation, a pre-order on $\mathcal{P}$:

(13) $$P \prec_{\mathcal{P}} P' \qquad \text{iff} \qquad \forall \phi \in \mathcal{L}\colon\;\; P' \models^{a} \phi \Rightarrow P \models^{a} \phi\,.$$

We write $\prec$ if $\mathcal{P}$ is determined by context. One may think of $P \prec P'$ as specifying that either P is a refinement of P' or P' is an abstraction of P. If P

is inconsistent, then $\{\phi \in \mathcal{L} \mid P \models^{\mathrm{a}} \phi\} = \mathcal{L}$ and so $P \prec Q$ holds for all models $Q \in \mathcal{P}$. The partial order quotient of the pre-order $\prec$ identifies models that cannot be distinguished with answers to any assertion checks; in particular, it identifies all inconsistent partial models.

As an example of reasoning within this reference ontology, we show that `impl` is contained in the relational inverse of $\prec_{\mathcal{P}}$: for all $P, P' \in \mathcal{P}$,

$$(P, P') \in \texttt{impl} \Rightarrow P' \prec P\,. \tag{14}$$

Proof by contradiction: If there is an $\phi \in \mathcal{L}$ with $P \models^{\mathrm{a}} \phi$ and $P' \not\models^{\mathrm{a}} \phi$, then the latter and (10) ensure $P' \models \neg\phi$ which contradicts $P \models^{\mathrm{a}} \phi$ as $(P, P') \in \texttt{impl}$.

Outline of paper

Section 2 presents the constraint language Alloy and its analyzer as an example ontology of the framework set out in the introduction.

In Section 3, we define AC semi-lattices and discuss important examples of such structures arising from partial model checking.

Section 4 classifies those AC semi-lattices that are finite and distributive; these structures often occur in applications and subsume finite distributive De Morgan lattices. These classifications have computational significance as they allow the efficient implementation of algebraic operations via transformations on the order-generating subset of prime elements.

Embeddings of AC semi-lattices into De Morgan lattices are documented in Section 5.

Using a mixed-power order, an internal consistency between satisfiability and validity denotations within AC semi-lattices is realized for partial model checking in Section 6.

In Section 7, we present a novel semantics for partial model checks under multiple viewpoints; it delegates checks to single-viewpoint models driven by a priority pre-order and synthesizes such checks into sets of obligations. Such sets enable the systematic detection and location of inconsistencies across model boundaries.

Section 8 takes a closer look at the role of negation for checks over partial models that are multiple-*valued* or have multiple viewpoints. In particular, we see that Heyting negations are inadequate in this setting.

Multiple-valued state-machines are discussed in Section 9 as an example application of the technical material and algorithms developed in this paper. In particular, we prove that our algorithm for synthesizing obligation sets also computes a multiple-*valued* compositional semantics whenever the order on primes is preserved as refinement between projected single-view models.

Section 10 states related work and Section 11 summarizes our findings.

2 An Example Ontology

Alloy is a constraint language with typed first-order logic plus transitive closure, supported by an automatic analyzer that restricts models to finite scopes [22]. For sake of simplicity, we ignore issues of scope sizes as well as Alloy's ability to analyze NP and co-NP formulas in this section and merely sketch how Alloy realizes an instance of the ontology (1–11). The set $\mathcal{T}$ consists of all conventional models of first-order logic: types are interpreted as sets, relation/function symbols are interpreted as relations/functions over the sets that interpret their types (respectively), and transitive closure has the usual meaning. For example, $T \in \mathcal{T}$ may consist of a set $X = \{a, b\}$ and a binary relation $R = \{(a, b), (b, b)\} \subseteq X \times X$. For the T above, this could be

```
sig X {
  R: set X
}

fact ( some a,b: X | a != b && X = a + b && R = a->a + a->b }
```

Facts are enforced to hold in an Alloy model. The remaining meaning of this program is self-explanatory given the definition of T. The query logic $\mathcal{L}$ for Alloy consists of sentences over typed first-order logic with transitive closure. The satisfaction relation $\models$ is the standard interpretation of such queries over models in $\mathcal{T}$.

To explain how to write assertion and consistency checks in Alloy let us consider a more elaborate example. In Figure 2, P is a partial model of the internal structure of classes and objects in an object-oriented programming language — specified through the two signatures and the fact in lines 1-11. The fact specifies that, in all implementations of P, all objects' methods are declared in their class.

Alloy uses the keyword `fun` to denote consistency checks; here lines 13-16. For example, `Scenario` asks whether one can generate an implementation of P in which the first object has at least one method and its type has all other classes as subtypes; and where a second object is of a different type; its analysis uses implicit existential quantification of its parameters. The invocation `^subtype` specifies the transitive closure of `subtype`.

Alloy uses the keyword `assert` to denote assertion checks; here lines 18-19. By virtue of (9), the analysis of the assertion `Safety` may be done by checking its negation: "Is there an implementation of P with an object that has a method which is declared private and public in the object's class?" Figure 3 shows an implementation of P that satisfies `Scenario`, so $P \models^{c}$ `Scenario`. In particular, the partial model P is *consistent*: there is some implementation T

```
sig X {
 f: X }

fact { some a,b,c: X | a->b + a->c in f && b != c }
```

Fig. 1. A partial model — written in D. Jackson's Alloy [22] — that is inconsistent as it has no implementations. The source of the inconsistency is subtle. Relations within signatures have the signature type (here `X`) as first type; if only an atomic type is declared (here `X`), the relation is a total function $f: X \to X$ by default. This contradicts the statement expressed in the fact.

of P. Figure 4 depicts a violation to the assertion `Safety` in P, from which we infer $P \not\models^{\mathrm{a}}$`Safety`.

Alloy's implementation relation associates to an Alloy program P all total models T that (1) have the same structure and name space for sets, relations, and functions; and (2) satisfy all facts stated in P. We leave it as an exercise to verify that Alloy is an instance of our ontology.

Since Alloy models are signatures plus finitely many constraints, they can easily be inconsistent — e.g. through the insertion or deletion of multiplicity constraints — and the detection of such inconsistencies is an important analysis activity supported by Alloy's analyzer: run a consistency check of a consistent query [21]; if no implementation is found and the scope of analysis was sufficiently large, the model is probably inconsistent. Figure 1 depicts such an inconsistent model P: there is no T with $(P, T) \in$ `impl`. It is useful to group together those Alloy models that share a common static structure. The static structure of the model in Figure 2, for example, is determined by its two signatures. Queries that refer only to instances of these signatures and their relational components are interpretable in any model of that static structure, as is the case for `Safety` and `Scenario` in Figure 2. Our ontology and its properties remain intact if such a restriction is being imposed. Each static structure corresponds to an application, e.g. one can fix static structures for behavioral models such as labeled transition systems and state machines.

3 AC Semi-lattices

Distinct modes of analysis ('assertion' or 'consistency' checks) result in distinct modes of meaning. Thus, the lattice $\{0 < 1\}$ serves as two meaning spaces — one for assertion checks (L^{a}) and one for consistency checks (L^{c}). The logical duality of satisfiability and validity suggest maps $\neg_{\mathrm{a}}: L^{\mathrm{a}} \to L^{\mathrm{c}}$ and $\neg_{\mathrm{c}}: L^{\mathrm{c}} \to L^{\mathrm{a}}$ which swap 0 and 1, so there seems to be no need to tag the lattices or negation maps with modes. As this paper will hopefully demonstrate, there are good reasons for tagging meanings with mode information — even if all partial models in question were consistent. Each mode $\mathrm{m} \in \{\mathrm{a}, \mathrm{c}\}$ and query

```
sig Class {
 pub, priv : set Method,
 field : set Field,
 subtype: set Class }

sig Object {
 type : Class,
 method : set Method,
 field : set Field }

fact { all o : Object | o.method = o.type.(pub + priv) }

fun Scenario(Object_1, Object_0 : Object) {  -- consistency check
 some Object_1.method
 Object_1.type.^subtype = Class - Object_1.type
 Object_1.type != Object_0.type }

assert Safety{  -- assertion check
 all o:Object | no o.method & o.type.pub & o.type.priv }
```

Fig. 2. A partial model of the internal structure of classes and objects in an object-oriented programming language — written in D. Jackson's Alloy [22].

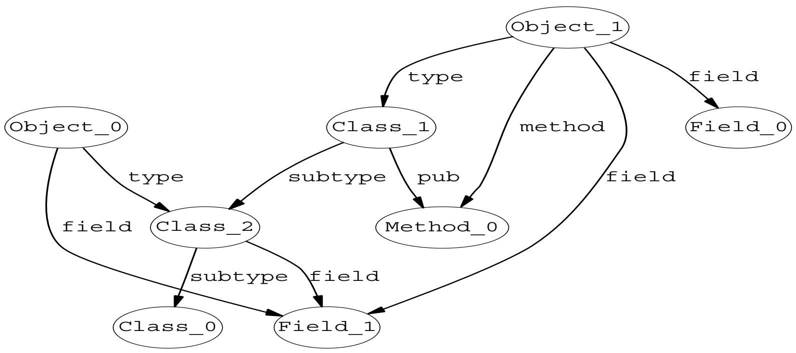

Fig. 3. Alloy's analyzer finds an implementation of the partial model in Figure 2 that satisfies `Scenario`.

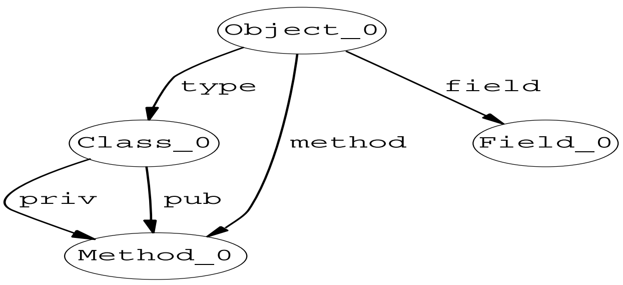

Fig. 4. Alloy's analyzer finds an implementation of the partial model in Figure 2 that violates `Safety`.

$$
\begin{array}{ll}
\neg_a\neg_c\phi = \phi & \neg\neg\phi = \phi \\
\neg_c\neg_a\phi = \phi & \neg(\phi \wedge \psi) = \neg\phi \vee \neg\psi \\
\phi \leq_a \psi \Rightarrow \neg_a\psi \leq_c \neg_a\phi & \neg(\phi \vee \psi) = \neg\phi \wedge \neg\psi \\
\phi \leq_c \psi \Rightarrow \neg_c\psi \leq_a \neg_c\phi & \phi \leq \psi = \neg\psi \leq \neg\phi
\end{array}
$$

Fig. 5. Axioms for AC semi-lattices (left) and De Morgan lattices [10] (right).

ϕ determine a set

$$\llbracket \phi \rrbracket^{\mathrm{m}} \stackrel{\mathrm{def}}{=} \{P \in \mathcal{P} \mid P \models^{\mathrm{m}} \phi\}. \tag{15}$$

The partial model of Figure 2 is in $\llbracket$ `Safety` $\rrbracket^{\mathrm{c}}$ but not in $\llbracket$ `Safety` $\rrbracket^{\mathrm{a}}$ — the latter was established in Figure 4. Evidently, we not only have to be able to compute or determine membership relations of such sets but it is also vital to tag them with the proper mode. Misinterpretations and inconsistent analyzes would follow otherwise.

This paper therefore proposes assertion-consistency semi-lattices — also referred to as AC semi-lattices — as ordered structures for semantics of partial model checks.

Definition 3.1 (i) A partial order $(Q, \leq)$ is an *inf-semi-lattice with one* (*sup-semi-lattice with zero*) iff all its finite subsets — including the empty set — have an infimum (supremum).

(ii) An *AC semi-lattice* is a tuple $(\mathcal{L}_a, \leq_a, \neg_a, \mathcal{L}_c, \leq_c, \neg_c)$, where $(\mathcal{L}_a, \leq_a)$ and $(\mathcal{L}_c, \leq_c)$ are partial orders, $\neg_a \colon \mathcal{L}_a \to \mathcal{L}_c$ and $\neg_c \colon \mathcal{L}_c \to \mathcal{L}_a$ are functions that meet the axioms of Figure 5 (left), and $(\mathcal{L}_a, \leq_a)$ is an inf-semi-lattice with one. (We write $\phi \vee \psi$ as a shorthand for $\neg(\neg\phi \wedge \neg\psi)$.)

(iii) An AC semi-lattice $(\mathcal{L}_a, \leq_a, \neg_a, \mathcal{L}_c, \leq_c, \neg_c)$ is an *AC lattice* iff $(\mathcal{L}_a, \leq_a)$ is also a sup-semi-lattice with zero.

(iv) A *De Morgan lattice* is a triple $(D, \leq, \neg)$ where $(D, \leq)$ is a lattice and $\neg : D \to D$ is a map meeting the axioms of Figure 5 (right).

Since the axioms of Figure 5 (left) spell out a witness to $(\mathcal{L}_a, \leq_a)$ and $(\mathcal{L}_c, \leq_c^{-1})$ being order-isomorphic, we infer that — in every AC semi-lattice — $(\mathcal{L}_c, \leq_c)$ is a sup-semi-lattice with zero. If we interpret $\neg$ in the second and third axiom of Figure 5 (right) as $\neg_a$ and $\neg_c$ (respectively), these axioms also hold in AC semi-lattices. We already discussed the finite distributive AC lattice $L^{\mathrm{a}} = L^{\mathrm{c}} = \{0 < 1\}$ with negations $\neg_a = \neg_c$ swapping 0 and 1. This structure also satisfies the axioms of Figure 5 (right) and is therefore a De Morgan lattice.

Our second example of an AC semi-lattice is

$$\mathcal{L}^{\mathrm{m}} = \{\llbracket\, \phi \,\rrbracket^{\mathrm{m}} \mid \phi \in \mathcal{L}\} \tag{16}$$

with ordering $\subseteq$ and

$$\neg_{\mathrm{m}}\llbracket\, \phi \,\rrbracket^{\mathrm{m}} \stackrel{\mathrm{def}}{=} \llbracket\, \neg\phi \,\rrbracket^{\neg\mathrm{m}}, \tag{17}$$

where $\neg a \stackrel{\mathrm{def}}{=} c$ and $\neg c \stackrel{\mathrm{def}}{=} a$. The ordering on $\mathcal{L}^{\mathrm{m}}$ encodes semantic entailment over the class $\mathcal{P}$ of all partial models and equality in $\mathcal{L}^{\mathrm{m}}$ acts as a quotient operation on queries: queries that cannot yield partial models with different answers in mode m are identified in that mode.

Proposition 3.2 *The tuple* $(\mathcal{L}^{\mathrm{a}}, \subseteq, \neg_{\mathrm{a}}, \mathcal{L}^{\mathrm{c}}, \subseteq \neg_{\mathrm{c}})$ *is an AC semi-lattice.*

All proofs are attached in an appendix. The proof of this proposition reveals and motivates why AC semi-lattices have finite infima for denotations of assertion checks: such checks *universally* quantify over all implementations and conjunction distributes over universal quantification; dually, consistency checks existentially quantify over implementations and disjunctions distribute over existential quantification. This also explains why we cannot work with lattices for partial model checks. In general, the set $\{\llbracket\, \phi \,\rrbracket^{\mathrm{a}}, \llbracket\, \psi \,\rrbracket^{\mathrm{a}}\}$ won't have a suprema in $\mathcal{L}^{\mathrm{a}}$. The maps $\neg_{\mathrm{a}}$ and $\neg_{\mathrm{c}}$ act as set complements

$$\mathcal{M} \mapsto \mathcal{P} \setminus \mathcal{M} \colon \mathbb{P}(\mathcal{P}) \to \mathbb{P}(\mathcal{P}) \tag{18}$$

only if we restrict $\mathcal{P}$ in (15) to the subset of consistent models in $\mathcal{P}$. In that AC semi-lattice, the assertion meaning of ϕ is then a subset of the consistency meaning of ϕ. But, as Proposition 3.2 testifies, AC semi-lattices are definable even in the presence of inconsistent partial models.

Example 3.3 As a third example, consider a topological space (X, τ). Let $L^{\mathrm{a}} = \tau$ be the set of opens and $L^{\mathrm{c}} = \{X \setminus O \mid O \in \tau\}$ be the set of closed sets, both ordered by $\subseteq$. The negation maps are $A \mapsto X \setminus A$. This is an AC lattice but, in general, not a De Morgan lattice; however, L^{a} and L^{c} have a structure-preserving embedding into the distributive De Morgan lattice $(\mathbb{P}(X), \subseteq, \setminus)$.

Given the existence of such embeddings, one may argue that there is no real need for AC semi-lattices at all. Yet partial model checks still have to remember the modes of semi-lattice elements even if these elements are the image points of such embeddings. For example, this is crucial for obtaining more efficient model-checking algorithms for multiple-*valued* analysis, as described in subsequent sections. Finally, we note that De Morgan lattices are special kinds of AC lattices in that reasoning about assertions *is* reasoning about consistencies, as would be the case for total models in (10), but with reversed order.

Proposition 3.4 *Every De Morgan lattice $(D, \leq, \neg)$ determines an AC lattice $(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \neg_{\mathrm{a}}, \mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}, \neg_{\mathrm{c}})$ such that $\mathcal{L}_{\mathrm{a}} = \mathcal{L}_{\mathrm{c}}$, $\leq_{\mathrm{a}}^{-1} = \leq_{\mathrm{c}}$, and $\neg_{\mathrm{a}} = \neg_{\mathrm{c}}$. Conservely, any such AC lattice determines a De Morgan lattice $(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \neg_{\mathrm{a}})$ and the compositions of these constructions are mutually inverse.*

4 Finite Distributive AC and De Morgan Lattices

We prove representation theorems for finite distributive AC semi-lattices and De Morgan lattices. We focus on finite distributive structures since they are important in applications such as multiple-valued and viewpoint-based reasoning. (Representation theorems for the non-distributive case are the subject of Section 5.) The computational contents of these results yield efficient implementations of the algebraic operations of such orders in terms of their prime elements. These results also recognize De Morgan lattices as AC lattices with an internal symmetry. Readers who focus on model-checking issues may safely skip to Section 6 on page 14.

Lemma 4.1 *Every finite AC semi-lattice is an AC lattice.*

Therefore, we may focus on AC lattices in the finite case. Finite distributive lattices are representable as topologies. In our case, this requires an understanding of the lower power-domain functor and a definition of its logical dual.

Let X be a finite set with a pre-order $\leq$ and τ the set of upper subsets U of X: $x \in U$ and $x \leq y$ imply $y \in U$. Closed sets are then lower subsets L: $y \in L$ and $x \leq y$ imply $x \in L$. We write $\mathsf{L}(X, \leq)$ for the set of lower subsets of X. Let

(19) $$\eta_X \colon (X, \leq) \to (\mathsf{L}(X, \leq), \subseteq) \qquad \eta_X(x) \stackrel{\text{def}}{=} {\downarrow} x = \{x' \in X \mid x' \leq x\}.$$

Given a monotone map $f \colon (X, \leq) \to (L, \leq)$ into a complete lattice, we write $\mathrm{ext}^{\mathcal{L}}(f)$ for the unique sup-map satisfying $\mathrm{ext}^{\mathcal{L}}(f) \circ \eta_X = f$. (A function $f \colon L \to L'$ between complete lattices L and L' is a sup-map iff $f(\bigvee A) = \bigvee f(A)$ for all $A \subseteq L$.) This defines the lower power-domain functor [1]

(20) $$\mathsf{L}(g) \stackrel{\text{def}}{=} \mathrm{ext}^{\mathcal{L}}(\eta_Y \circ g)$$

where $g \colon (X, \leq) \to (Y, \sqsubseteq)$ is a monotone map between pre-orders. Dually, we write $\mathsf{U}(X, \leq)$ for the set of upper subsets of X. The maps

$$\backslash_{\mathrm{a}} : (\mathsf{L}(X, \leq), \subseteq) \to (\mathsf{U}(X, \leq), \supseteq) \qquad \backslash_{\mathrm{a}}(A) = X \setminus A$$

(21) $$\backslash_{\mathrm{c}} : (\mathsf{U}(X, \leq), \supseteq) \to (\mathsf{L}(X, \leq), \subseteq) \qquad \backslash_{\mathrm{c}}(A) = X \setminus A$$

are monotone, as is

(22) $$\epsilon_X \stackrel{\text{def}}{=} \backslash_{\mathrm{a}} \circ \eta_X \colon (X, \leq) \to (\mathsf{U}(X, \leq), \supseteq).$$

For f as above, we write $\mathrm{ext}^{\mathcal{U}}(f)$ for the unique sup-map satisfying $\mathrm{ext}^{\mathcal{U}}(f) \circ \epsilon_X = f$. This defines the *complemented lower power-domain functor*

$$\mathsf{U}(g) \stackrel{\mathrm{def}}{=} \mathrm{ext}^{\mathcal{U}}(\epsilon_Y \circ g) \tag{23}$$

for g as above. Finally, let $\mathrm{Aut}(X, \leq)$ be the set of order-isomorphisms of type $(X, \leq) \to (X, \leq)$.

Proposition 4.2 (i) $\mathsf{U}(\cdot)$: **Pre** $\to$ **Lat** *is a monotone functor between the categories of pre-orders with monotone maps and complete lattices with sup-maps.*

(ii) *We have* $\backslash_{\mathrm{a}} = \mathrm{ext}^{\mathcal{L}}(\epsilon_X)$*,* $\backslash_{\mathrm{c}} = \mathrm{ext}^{\mathcal{U}}(\eta_X)$*,* $\eta_X = \backslash_{\mathrm{c}} \circ \epsilon_X$*, and* $\mathsf{U}(f) = \backslash_{\mathrm{a}} \circ \mathsf{L}(f) \circ \backslash_{\mathrm{c}}$.

(iii) *For* partial orders $(X, \leq)$*, the functors* $\mathsf{L}(\cdot)$ *and* $\mathsf{U}(\cdot)$ *induce bijective maps of type* $\mathrm{Aut}(X, \leq) \to \mathrm{Aut}(\mathsf{L}(X, \leq))$ *and* $\mathrm{Aut}(X, \leq) \to \mathrm{Aut}(\mathsf{U}(X, \leq))$ *(respectively).*

We now have all the machinery at our disposal to prove a representation theorem for finite distributive AC lattices.

Theorem 4.3 *Every AC semi-lattice* $(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \neg_{\mathrm{a}}, \mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}, \neg_{\mathrm{c}})$ *whose partial order* $(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}})$ *is a finite distributive lattice can be represented as*

$$(\mathsf{L}(X, \leq), \subseteq, \mathrm{ext}^{\mathcal{L}}(\epsilon_X \circ i), \mathsf{U}(X, \leq), \subseteq, \mathrm{ext}^{\mathcal{U}}(\eta_X \circ i^{-1})) \tag{24}$$

for some partial order $(X, \leq)$ *and some* $i \in \mathrm{Aut}(X, \leq)$*.*

Example 4.4 Consider the finite partial order $(X, \leq)$, where $X = \{x, y, z\}$ and y and z are incomparable but greater than x. There are two elements i in $\mathrm{Aut}(X, \leq)$. For i being the identity id_X, $\neg_{\mathrm{a}} \stackrel{\mathrm{def}}{=} \mathrm{ext}^{\mathcal{L}}(\epsilon_X \circ i)$ and $\neg_{\mathrm{c}} \stackrel{\mathrm{def}}{=} \mathrm{ext}^{\mathcal{U}}(\eta_X \circ i^{-1})$ act as $A \mapsto X \setminus A$. For i being the map that swaps y and z and keeps x fixed, we have $\neg_{\mathrm{a}}(L) = \bigcup\{X \setminus {\downarrow} i(v) \mid v \in L\}$ and $\neg_{\mathrm{c}}(U) = \bigcup\{{\downarrow} i^{-1}(v) \mid v \notin U\}$. For example, $\neg_{\mathrm{c}}(\{z\}) = {\downarrow} x \cup {\downarrow} z = \{x, z\}$ and $\neg_{\mathrm{a}}(\{x, y\}) = (X \setminus {\downarrow} i(x)) \cup (X \setminus {\downarrow} i(y)) = \{y, z\} \cup \{y\} = \{y, z\}$. In particular, $\{z\} \cap \neg_{\mathrm{c}}(\{z\})$ and $\{x, y\} \cap \neg_{\mathrm{a}}(\{x, y\})$ are non-empty.

Given a lower set $L \in \mathsf{L}(X, \leq)$, its set complement $X \setminus L$ is in $\mathsf{U}(X, \leq)$. If $i\colon (X, \leq) \to (X, \leq^{-1})$ is an order isomorphism with $i \circ i = \mathrm{id}_X$, then $L \mapsto \{i(x) \mid x \in X \setminus L\}$ is a map of type $\mathsf{L}(X, \leq) \to \mathsf{L}(X, \leq)$ which makes $\mathsf{L}(X, \leq)$ into a De Morgan lattice. The existence of such an i is therefore a sufficient condition for $\mathsf{L}(X, \leq)$ being a De Morgan lattice; this condition is also necessary for finite distributive De Morgan lattices.

Theorem 4.5 *Every finite distributive De Morgan lattice* $(\mathcal{L}, \leq, \neg)$ *can be represented as* $(\mathsf{L}(X, \leq), \subseteq, \neg)$ *for a finite partial order* $(X, \leq)$ *where*

$$\neg L = \{i(x) \mid x \in X \setminus L\} \tag{25}$$

for some monotone $i\colon (X, \leq) \to (X, \leq^{-1})$ *with* $i \circ i = \mathrm{id}_X$.

It would be of interest to see whether the proof of Theorem 4.5 can be reduced to the one of Theorem 4.3.

Example 4.6 (i) Consider the finite self-dual partial order $X = \{x, y\}$ in the discrete ordering. There are exactly two monotone maps $i\colon (X, \leq) \to (X, \leq^{-1})$ with $i \circ i = \mathrm{id}_X$, the identity id_X and sw which permutes x and y. This gives rise to two different De Morgan negations: $\neg \overset{\mathrm{def}}{=} \mathrm{ext}^{\mathcal{L}}(\epsilon_X \circ \mathrm{id}_X)$ and $\neg' \overset{\mathrm{def}}{=} \mathrm{ext}^{\mathcal{L}}(\epsilon_X \circ \mathrm{sw})$. We have $\neg\{x\} = \{y\}$, and $\neg\{y\} = \{x\}$, $\neg'\{x\} = \{x\}$, and $\neg\{y\} = \{y\}$. Thus, the first negation is the classical one, and the second negation renders Belnap's four-valued logic [3]. By Theorem 4.5, this lattice cannot have any other De Morgan negations.

(ii) For a negative example, consider the finite partial order $(X, \leq)$ of Example 4.4, where $X = \{x, y, z\}$ and y and z are incomparable but greater than x. This partial order is not self-dual; in particular, there cannot be a monotone map $i\colon (X, \leq) \to (X, \leq^{-1})$ with $i \circ i = \mathrm{id}_X$. By Theorem 4.5, $\mathsf{L}(X, \leq)$ cannot have a De Morgan negation. To see this explicitly, the third axiom of Figure 5 (right) renders $\{\} = \neg\{x, y, z\} = \neg(\{x, y\} \cup \{x, z\}) = \neg\{x, y\} \cap \neg\{x, z\}$. Thus, at least one of the sets $\neg\{x, y\}$ and $\neg\{x, z\}$ has to be empty as all non-empty lower sets contain x. But then $\neg$ cannot be a bijection, since $\{\} = \neg\{x, y, z\}$. Of course, $(\mathsf{L}(X, \leq), \subseteq, \setminus_{\mathrm{a}}, \mathsf{U}(X, \leq), \subseteq, \setminus_{\mathrm{c}})$ is an AC lattice.

5 Embeddings

This section merely documents some rather obvious embeddings of AC semi-lattices into AC lattices and De Morgan lattices; the reader may therefore skip to Section 6 on page 14 as these embeddings are not used in the remainder of this paper. AC semi-lattices, AC lattices, and De Morgan lattices have structure-preserving maps.

Definition 5.1 (i) The category AC has all AC semi-lattices as objects. A morphism in AC, $f\colon (\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}) \to (\mathcal{L}'_{\mathrm{a}}, \leq'_{\mathrm{a}}, \mathcal{L}'_{\mathrm{c}}, \leq'_{\mathrm{c}})$, is a pair $(f_{\mathrm{a}}, f_{\mathrm{c}})$ where

- $f_{\mathrm{a}}\colon (\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}) \to (\mathcal{L}'_{\mathrm{a}}, \leq'_{\mathrm{a}})$ preserves all finite infima,
- $f_{\mathrm{c}}\colon (\mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}) \to (\mathcal{L}'_{\mathrm{c}}, \leq_{\mathrm{c}}')$ preserves all finite suprema,
- $f_{\mathrm{a}} \circ \neg_{\mathrm{c}} = \neg'_{\mathrm{c}} \circ f_{\mathrm{c}}$, and
- $f_{\mathrm{c}} \circ \neg_{\mathrm{a}} = \neg'_{\mathrm{a}} \circ f_{\mathrm{a}}$.

(ii) The category ACL has as objects all AC lattices and all morphisms f of *AC* such that f_{a} and f_{c} preserve all finite infima *and* suprema.

(iii) The category DM is a full subcategory of ACL whose objects are all De Morgan lattices, represented in the form of Proposition 3.4.

It is routine to verify that these data give rise to categories. There is an analogy between their morphisms and Galois connections [12] in that f_{a} and f_{c} determine each other uniquely:

(26) $$f_{\mathrm{a}} = \neg'_{\mathrm{c}} \circ f_{\mathrm{c}} \circ \neg_{\mathrm{a}} \qquad \text{and} \qquad f_{\mathrm{c}} = \neg'_{\mathrm{a}} \circ f_{\mathrm{a}} \circ \neg_{\mathrm{c}}\,.$$

We use these categories to formalize an embedding, a "final" construction, of AC lattices into De Morgan lattices. We then conclude this section with formalizing an embedding of AC *semi*-lattices into De Morgan lattices.

5.1 Embedding AC lattices

For every AC lattice $(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \neg_{\mathrm{a}}, \mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}, \neg_{\mathrm{c}})$, consider the partial ordering

(27) $$(Q, \leq) \stackrel{\mathrm{def}}{=} (\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}) \times (\mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}})$$

and the map $\neg\colon (Q, \leq) \to (Q, \leq^{-1})$ defined by

(28) $$\neg(x, y) \stackrel{\mathrm{def}}{=} (\neg_{\mathrm{c}} y, \neg_{\mathrm{a}} x)\,.$$

For $(x, y) \leq (u, v)$ in $(Q, \leq)$, the axioms for AC semi-lattices render $\neg_{\mathrm{a}} u \leq_{\mathrm{c}} \neg_{\mathrm{a}} x$ and $\neg_{\mathrm{c}} v \leq_{\mathrm{a}} \neg_{\mathrm{c}} y$ from which we infer that $\neg$ in (28) is monotone for the type $(Q, \leq) \to (Q, \leq^{-1})$. Similarly, we conclude $\neg \circ \neg = \mathrm{id}_Q$. Therefore, the tuple $(Q, \leq, \neg, Q, \leq, \neg)$ is an object in DM. The projections $\pi_{\mathrm{a}}\colon (Q, \leq) \to (\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}})$ with $\pi_{\mathrm{a}}(x, y) \stackrel{\mathrm{def}}{=} x$ and $\pi_{\mathrm{c}}\colon (Q, \leq) \to (\mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}})$ with $\pi_{\mathrm{c}}(x, y) \stackrel{\mathrm{def}}{=} y$ preserve binary infima *and* suprema such that $\pi_{\mathrm{a}} \circ \neg = \neg_{\mathrm{c}} \circ \pi_{\mathrm{c}}$ and $\pi_{\mathrm{c}} \circ \neg = \neg_{\mathrm{a}} \circ \pi_{\mathrm{a}}$. Thus,

(29) $$\pi\colon (Q, \leq, \neg, Q, \leq, \neg) \to (\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \neg_{\mathrm{a}}, \mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}, \neg_{\mathrm{c}})$$

is a morphism in ACL whose source object is a De Morgan lattice. (If $(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}})$ is distributive, then so is $(Q, \leq)$.) Moreover, for any De Morgan lattice $(D, \sqsubseteq, \neg_D, D, \sqsubseteq, \neg_D)$ and morphism

(30) $$\alpha\colon (D, \sqsubseteq, \neg_D, D, \sqsubseteq, \neg_D) \to (\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \neg_{\mathrm{a}}, \mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}, \neg_{\mathrm{c}})$$

in ACL, there is a unique morphism

(31) $$h\colon (D, \sqsubseteq, \neg_D, D, \sqsubseteq, \neg_D) \to (Q, \leq, \neg, Q, \leq, \neg) \text{ with } h_{\mathrm{a}} = h_{\mathrm{c}}$$

in ACL such that $\alpha_{\mathrm{a}} = \pi_{\mathrm{a}} \circ h_{\mathrm{a}}$ and $\alpha_{\mathrm{c}} = \pi_{\mathrm{c}} \circ h_{\mathrm{c}}$: $h_{\mathrm{m}}(d) \stackrel{\mathrm{def}}{=} (\alpha_{\mathrm{a}}(d), \alpha_{\mathrm{c}}(d))$ defines h_{m} uniquely since $h_{\mathrm{a}} = h_{\mathrm{c}}$. Since α_{a} and α_{c} are lattice morphisms, h_{a} and h_{c} are lattice morphisms as well. Finally, $h_{\mathrm{a}} \circ \neg_D = \neg \circ h_{\mathrm{c}}$ and $h_{\mathrm{c}} \circ \neg_D = \neg \circ h_{\mathrm{a}}$ follow from the definition of h_{m}, $\alpha_{\mathrm{a}} \circ \neg_D = \neg \circ \alpha_{\mathrm{c}}$, and $\alpha_{\mathrm{c}} \circ \neg_D = \neg \circ \alpha_{\mathrm{a}}$.
Thus, AC *lattices* have a universal embedding into a De Morgan lattice. In particular, this theorem applies to all *finite* AC semi-lattices. For general AC semi-lattices, we cannot use $(Q, \leq, \neg)$ of (27) since it will neither be an inf-

semi-lattice nor a sup-semi-lattice in general. Note that the set of fixed points of $\neg$ in (28) is $\{(x, \neg_{\mathrm{a}} x) \mid x \in \mathcal{L}_{\mathrm{a}}\}$, which equals $\{(\neg_{\mathrm{c}} y, y) \mid y \in \mathcal{L}_{\mathrm{c}}\}$.

5.2 Embedding AC semi-lattices

We now turn to a construction that embeds AC semi-lattices into De Morgan lattices that are complete, in fact, completely distributive [12]. Given an AC semi-lattice $(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \neg_{\mathrm{a}}, \mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}, \neg_{\mathrm{c}})$, we define a completely distributive AC lattice:

- $(\mathcal{L}'_{\mathrm{a}}, \leq'_{\mathrm{a}}) \stackrel{\mathrm{def}}{=} (\mathsf{L}(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}), \subseteq)$,
- $(\mathcal{L}'_{\mathrm{c}}, \leq'_{\mathrm{c}}) \stackrel{\mathrm{def}}{=} (\mathsf{U}(\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}), \subseteq)$, and
- its negations $\neg'_{\mathrm{a}}$ and $\neg'_{\mathrm{c}}$ act as $A \mapsto \mathcal{L}_{\mathrm{a}} \setminus A$.

The maps $\epsilon_{\mathrm{a}} \colon (\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}) \to (\mathcal{L}'_{\mathrm{a}}, \leq'_{\mathrm{a}})$ and $\epsilon_{\mathrm{c}} \colon (\mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}) \to (\mathcal{L}'_{\mathrm{c}}, \leq'_{\mathrm{c}})$ are defined as

(32) $\epsilon_{\mathrm{a}}(x) \stackrel{\mathrm{def}}{=} {\downarrow_a} x$

(33) $\epsilon_{\mathrm{c}}(x) \stackrel{\mathrm{def}}{=} \mathcal{L}_{\mathrm{a}} \setminus \epsilon_{\mathrm{a}}(\neg_{\mathrm{c}} x)\,.$

It is routine to show that ϵ_{a} preserves all finite infima, ϵ_{c} preserves all finite suprema, and that both are order-isomorphisms onto their respective images. Since $\neg'_{\mathrm{c}} \circ \epsilon_{\mathrm{c}} = \epsilon_{\mathrm{a}} \circ \neg_{\mathrm{c}}$ and $\neg'_{\mathrm{a}} \circ \epsilon_{\mathrm{a}} = \epsilon_{\mathrm{c}} \circ \neg_{\mathrm{a}}$, we have a morphism

$$\epsilon \colon (\mathcal{L}_{\mathrm{a}}, \leq_{\mathrm{a}}, \neg_{\mathrm{a}}, \mathcal{L}_{\mathrm{c}}, \leq_{\mathrm{c}}, \neg_{\mathrm{c}}) \to (\mathcal{L}'_{\mathrm{a}}, \leq'_{\mathrm{a}}, \neg'_{\mathrm{a}}, \mathcal{L}'_{\mathrm{c}}, \leq'_{\mathrm{c}}, \neg'_{\mathrm{c}}) \qquad (34)$$

in AC such that its co-domain is completely distributive. We did not investigate whether this construction is a left adjoint for the forgetful functor into AC, which forgets that objects are complete and morphisms preserve *all* infima and suprema (respectively).

At the object level, we may now compose the two embeddings discussed in this section to embed an AC semi-lattice into a De Morgan lattice.

6 The Mixed Power-order

The AC semi-lattice of Proposition 3.2 orders queries with respect to the partial models they assert or judge to be consistent. Dually, one may order partial models with respect to their asserted queries, as done in (13).

Definition 6.1 For $\prec$ as in (13) and $\mathcal{S} \subseteq \mathcal{P}$, we write ${\downarrow}\mathcal{S} \stackrel{\mathrm{def}}{=} \{P \in \mathcal{P} \mid \exists Q \in \mathcal{S} \colon P \prec Q\}$ and ${\uparrow}\mathcal{S} \stackrel{\mathrm{def}}{=} \{P \in \mathcal{P} \mid \exists Q \in \mathcal{S} \colon Q \prec P\}$.

The refinement order $\prec$ maintains consistency at the meta-level if models or queries are consistent.

Proposition 6.2 *For every $\phi \in \mathcal{L}$,*

(i) *$\| \phi \|^{\mathrm{a}}$ is a lower set and $\| \phi \|^{\mathrm{c}}$ is an upper set in $(\mathcal{P}, \prec)$;*

(ii) *the equation*

$$\| \phi \|^{\mathrm{a}} = {\downarrow}(\| \phi \|^{\mathrm{a}} \cap \| \phi \|^{\mathrm{c}}) \tag{35}$$

holds iff ϕ is consistent; and

(iii) *if $\mathcal{P}$ is restricted to consistent models only, (35) holds for* all *ϕ, even inconsistent ones.*

Implementations of partial models often need to meet multiple, potentially conflicting, constraints — making inconsistency a real possibility. Queries, however, are checked one at a time, and will rarely ever be inconsistent themselves. Thus, equation (35) will hold in almost all practical situations: a material ontology can violate (35) only by checking an inconsistent query over a set of models of which some are inconsistent. Equation (35) specifies an element of the mixed power-order over $\prec$.

Definition 6.3 The *mixed power-order* [14,15] $\mathsf{M}_{(X,\leq)}$ for a pre-order $(X, \leq)$ is the sublattice of $(\mathsf{L}(X, \leq), \subseteq) \times (\mathsf{U}(X, \leq), \supseteq)$ consisting of all pairs (L, U) that satisfy the consistency condition

$$L = \{x \in X \mid \exists y \in L \cap U \colon x \leq y\}\,. \tag{36}$$

The original definition of the mixed power-domain [14,15] assumes that $\leq$ is a partial order, X a domain, and L and U enjoy topological properties.

Example 6.4 (i) Let $\mathcal{A}_\Sigma$ be a set of Alloy models with static structure Σ. Then $(\mathcal{A}_\Sigma, \prec_\Sigma)$ is a pre-order, $\| \phi \|_\Sigma \overset{\mathrm{def}}{=} (\| \phi \|^{\mathrm{a}}_\Sigma, \| \phi \|^{\mathrm{c}}_\Sigma)$ is an element of $\mathsf{M}_{(\mathcal{A}_\Sigma, \prec_\Sigma)}$ whenever all models in $\mathcal{A}_\Sigma$ are consistent or ϕ is consistent. In that case, (35) is an instance of (36).

(ii) The set-complement negation $(L, U) \to (X \setminus U, X \setminus L) \colon \mathsf{M}_{(X,\leq)} \to \mathsf{M}_{(X,\leq)}$ is well defined iff $\leq$ is flat: $x \leq y$ implies $x = y$. For if $\leq$ is not flat, let $L = X$ and $U = \max(X)$, where $\max(X)$ denotes the set of maximal elements of $(X, \leq)$. Then $(L, U) \in \mathsf{M}_{(X,\leq)}$ but $(X \setminus U, X \setminus L) = (L', \{\})$ with $L' \neq \{\}$ and so $(X \setminus U, X \setminus L) \notin \mathsf{M}_{(X,\leq)}$. If $\leq$ is flat, the negation is well defined.

Although the set-complement negation in $\mathsf{M}_{(\mathcal{P},\prec)}$ is not well defined in general, it is well defined on the subset of its $\mathcal{L}$ denotations if all models in $\mathcal{P}$ are consistent — by Proposition 6.2(iii). Not every element of $\mathsf{M}_{(\mathcal{P},\prec)}$ is of that form, e.g. $(\mathcal{P}, \max(\mathcal{P}))$. In general, mixed power-orders themselves have neither binary infima, suprema [15], nor a De Morgan-like negation operation.

Example 6.5 (i) Consider $(X, \leq) = (\{*\}, (*, *))$, $(L_1, U_1) = (\{*\}, \{*\})$, and $(L_2, U_2) = (\{\}, \{\})$. Both pairs are elements of $\mathsf{M}_{(X,\leq)}$ but $(L_1 \cup L_2, U_1 \cap U_2) = (\{*\}, \{\})$ is not. Any upper bound (A, C) of the former two sets in $\mathsf{M}_{(X,\leq)}$ needs to be such that $* \in A$ and $C \subseteq \{\}$. Hence (L_1, U_1) and (L_2, U_2) cannot have a suprema in $\mathsf{M}_{(X,\leq)}$.

(ii) Let again $X = \{x, y, z\}$ where y and z are two maximal elements and x the sole minimal element of $(X, \leq)$. Consider

$$(L_1, U_2) = (\{x, z\}, \{z\}) \qquad (L_2, U_2) = (\{x, y\}, \{y\})$$
$$(L, U) = (\{\}, \{y, z\}) \qquad (L', U') = (\{x\}, \{x, y, z\})\,.$$

All four pairs are elements of $\mathsf{M}_{(X,\leq)}$. Both (L, U) and (L', U') are lower bounds of $\mathcal{X} = \{(L_1, U_1), (L_2, U_2)\}$. If $\mathcal{X}$ has an infimum (A, C) in $\mathsf{M}_{(X,\leq)}$, then

- $L' \subseteq A$ follows — i.e. $x \in A$ — since (L', U') is a lower bound of $\mathcal{X}$. Since (A, C) is another such lower bound, we get $A \subseteq \{x, z\}$ and $A \subseteq \{x, y\}$. Thus, $A \subseteq \{x, z\} \cap \{x, y\} = \{x\}$ shows $A = \{x\}$ as $x \in A$ was already established;
- $U_1 \subseteq C$ and $U_2 \subseteq C$ — i.e. $\{y, z\} \subseteq C$ since (A, C) is a lower bound of $\mathcal{X}$. Since (A, C) is the greatest such lower bound, we obtain $C \subseteq U$ and $C \subseteq U'$ which implies $C \subseteq \{y, z\}$. Thus, $C = \{y, z\}$.

A contradiction arises since $(A, C) = (\{x\}, \{y, z\})$ is not in $\mathsf{M}_{(X,\leq)}$.

(iii) Given any pre-order $(X, \leq)$, assume the existence of a map $\neg \colon \mathsf{M}_{(X,\leq)} \to \mathsf{M}_{(X,\leq)}$ such that $\neg \circ \neg = \mathrm{id}_{\mathsf{M}_{(X,\leq)}}$ and $(L, U) \leq (L', U')$ implies $\neg(L', U') \leq \neg(L, U)$ in $\mathsf{M}_{(X,\leq)}$. Then $\mathsf{M}_{(X,\leq)}$ is order-isomorphic to its own dual. Since $(\{\}, X)$ is the least element of $\mathsf{M}_{(X,\leq)}$, $\mathsf{M}_{(X,\leq)}$ has a greatest element (A, C). From $(X, X) \in \mathsf{M}_{(X,\leq)}$ we then get $X \subseteq A$, so $A = X$. From $(\{\}, \{\}) \in \mathsf{M}_{(X,\leq)}$ we conclude $C \subseteq \{\}$, so $C = \{\}$. But $(X, \{\})$ is in $\mathsf{M}_{(X,\leq)}$ only when X is empty.

If we change the ordering in $\mathsf{M}_{(X,\leq)}$ such that it be $\subseteq$ in each coordinate, then $(\mathsf{L}((X, \leq)), \subseteq) \times (\mathsf{U}((X, \leq)), \subseteq)$ is simply the De Morgan lattice of Section 5.1 into which we may embed the AC lattice of Example 3.3 for the topology $\tau = \mathsf{U}((X, \leq))$ over $(X, \leq)$. If we restrict Q in (27) to its set of coherent pairs, those pairs that satisfy the mix condition (36) — which does not appeal to the ordering on $\mathsf{M}_{(X,\leq)}$ — we have a least element $(\{\}, \{\})$ and a greatest element (X, X); alas, Examples 6.4(ii) and 9.5 (on page 26) still stand. Nonetheless, it is still conceivable that another map makes this coherent subset of Q into a De Morgan lattice. We did not investigate this any further.

7 Multiple Viewpoints

AC semi-lattices are also useful for reasoning in situations where multiple partial models mean to describe the same underlying reality.

For example, distributed agents or systems need to store and share incomplete information. "Group membership" and "fault detection" are mechanisms used in conflict-resolution in fault-tolerant distributed systems [33]. Groups need to maintain a consistent view of which services are available at which hosts, whereas fault detection specifies information flow regarding the change of such information. Inconsistencies result in scheduling services on hosts that no longer exist or no longer offer such services. Records of such information are partial models since the relevant information is expressed as guarantees ("This host offers that service.") or possibilities ("It is consistent with what we know that this host offers that service.").

Another application of multiple partial models is in the context of requirements engineering and requirements elicitation, where behavioral requirements are encoded as partial models. Since requirements solicitation typically means that different stake-holders come up with different observations (and omissions of those), there is the need to detect and locate inconsistencies across model boundaries [30,29]. Although one could lump together all stake-holders' constraints into one master model which would then be checked for consistency, this may not be desirable for two reasons. First, stake-holders may use different formalisms for specifying partial models — with slight variations of static structures and possibly very different name spaces. Second, even if this obstacle can be overcome, one often wants to *prioritize* requirements [37] in order to aggressively drive the specification and implementation process. This is especially true in goal-oriented approaches to requirements engineering [34], where goals will have implicit or explicit comparative rankings.

Therefore, we propose to work with a partial order of priorities between models of the same underlying artifact. This will naturally lead us to consider a generalization of finite AC semi-lattices and, perhaps surprisingly, to more efficient algorithms for multiple-*valued* model checking. Multiple-valued models and multiple-valued logics have a long-standing tradition (see e.g. [13]). But few works have pushed multiple-valued semantics into the realm of partial models, perhaps as one may think of partial models as three-valued models with Kleene's strong semantics [24]; e.g. as done successfully by Bruns and Godefroid in [5] for Kripke structures.

In a case study in Section 9, we show that the framework for multiple-viewpoint partial models of this section specializes to a multiple-valued compositional semantics whenever priorities between partial models are honored *as refinements.*

The consistency condition (36) for partial checks reduces to $L \subseteq U$ whenever the pre-order $\prec$ is flat: $x {\prec} y$ implies $x = y$, but our semantics in (35) was based on a non-flat pre-order of refinement or abstraction. Such non-flat applications of (36) also occur within the framework of abstract interpretation [9]. Multiple-valued partial models and their analysis are another such non-flat application. In a simplified scenario, each element x of a finite partial order $(X, \leq)$ has a partial model $P_x \in \mathcal{P}$ — a different view of a software artifact. The relation $x \leq y$ expresses that P_y has higher or equal priority than P_x if we think of these models as implementation goals, demands on existing software, state descriptions etc. The interpretation of $\leq$ suggests that assertions validated at a viewpoint are then *obliged* to hold at viewpoints of lesser priority. That is, we *pretend* that $\leq$ be contained in the refinement relation $\prec$ and explore the consequences of this containment. In light of Proposition 6.2(i), queries consistent at a viewpoint are then *obliged* to be consistent in viewpoints of higher priorities. A semantics [20] collects these obligations of validity $\{\!|X{:}\phi|\!\}^{\mathrm{a}}$ and consistency $\{\!|X{:}\phi|\!\}^{\mathrm{c}}$

$$
\begin{aligned}
&\{\!|X{:}\phi|\!\}^{\mathrm{a}} \stackrel{\mathrm{def}}{=} \{x \in X \mid \exists y \in X{:}\, x \leq y,\ P_y \models^{\mathrm{a}} \phi\} \\
&\{\!|X{:}\phi|\!\}^{\mathrm{c}} \stackrel{\mathrm{def}}{=} \{x \in X \mid \exists y \in X{:}\, y \leq x,\ P_y \models^{\mathrm{c}} \phi\} \\
(37)\quad &\{\!|X{:}\phi|\!\} \stackrel{\mathrm{def}}{=} (\{\!|X{:}\phi|\!\}^{\mathrm{a}}, \{\!|X{:}\phi|\!\}^{\mathrm{c}})\,.
\end{aligned}
$$

The set $\{\!|X{:}\phi|\!\}^{\mathrm{a}}$ computes the *assertion-cone of influence* of $\leq$ for the query ϕ: if we pretend that $\leq$ is contained in $\prec$, which models assert ϕ? Similarly, the set $\{\!|X{:}\phi|\!\}^{\mathrm{c}}$ computes the *consistency-cone of influence*, those models for which ϕ is consistent, if we pretend that $\leq$ is contained in $\prec$. These cones of influence are first-order in that consistencies or validities derived from $\leq$ don't interfere with the computation of checks $P_y \models^{\mathrm{a}} \phi$ and $P_y \models^{\mathrm{c}} \phi$. This guarantees efficient computation of the sets in (37) but may not be desirable in approaches where the analysis uses a proof theory, in which obligations may function as additional assumptions. For example, labeled deduction systems can be used to that end [16].

If $\leq$ happens to be contained in $\prec$ — a fact we secure for reductions of multiple-valued model checking in subsequent sections — then we can remove all existential quantifiers in (37) as one can always choose x as a witness for y. This results in non-trivial efficiency gains in multiple-valued model checking, as shown in subsequent sections.

A query ϕ's cone of influence $\{\!|X{:}\phi|\!\}$ can be computed efficiently only if model checks $P_x \models^{\mathrm{m}} \phi$ can be decided efficiently. The algorithm for computing $\{\!|X{:}\phi|\!\}^{\mathrm{m}}$ is depicted in Figure 6. The number of iterations of its while-statement is bounded by the size of X, but will be significantly less than

```
if (m == a) { leq = <= } % <= for mode a
   else     { leq = >= } % inverse of <= for mode c
U = X;
L = emptyset;
while (U != emptyset) {
% invariants: L contained in { X : phi }^m,
%   { X : phi }^m contained in L union U
  for all x in max(U,leq) {
    if (check(P_x,phi,m)) {
      let A = { y | y leq x } in {
        L = L union A
        U = U \ A
      }
    } else {
      U = U \ { x };
    }
  }
} % post: L = { X : phi }^m
```

Fig. 6. Algorithm for computing $\{\!|X{:}\phi|\!\}^{\mathrm{m}}$. The order `leq` equals $\leq$ for mode a and $\leq^{-1}$ for mode c. The method `check(P_x,phi,m)` decides $P_x \models^{\mathrm{m}} \phi$, potentially abstracting undecided checks to appropriate boolean values. The method `max(U,leq)` returns the set of maximal elements of the partial order `(U,leq)`.

this for priority orders occurring in practice. We emphasize that only two things are needed for these computations: a model-checking engine for deciding $P_x \models^{\mathrm{m}} \phi$ and an implementation of the partial order $(X, \leq)$. The former is achievable for certain static structures — e.g. partial Kripke structures [5] and CTL have efficient model checks — or by putting bounds on the size of models — as done in the tool Alloy.

Given a finite set Φ of queries, one may use the cone-of-influence semantics of (37) for instrumented checks. For example, $\cap_{\phi\in\Phi}\{\!|X{:}\phi|\!\}^{\mathrm{a}}$ identifies those viewpoints that are obliged to assert every query in Φ; and $\cap_{\phi\in\Phi}\{\!|X{:}\phi|\!\}^{\mathrm{c}}$ singles out those viewpoints that are obliged to hold the conjunction $\bigwedge\Phi$ to be consistent. If $\Phi = \{\phi, \neg\phi\}$ or, more generally, if Φ is inconsistent, this intersection can detect and locate the source of such inconsistency in the multiple-viewpoint setting. Locating the sources of such conflicts simply amounts to choosing all maximal (for m = a) or minimal (for m = c) witnesses for the existential quantifier that determines membership in each $\{\!|X{:}\phi|\!\}^{\mathrm{m}}$ being considered.

Example 7.1 Consider the partial order $(X, \leq)$ with $X = \{v, x, y, z\}$ of Figure 7, where for each $u \in X$ the model P_u is a partial interpretation of the

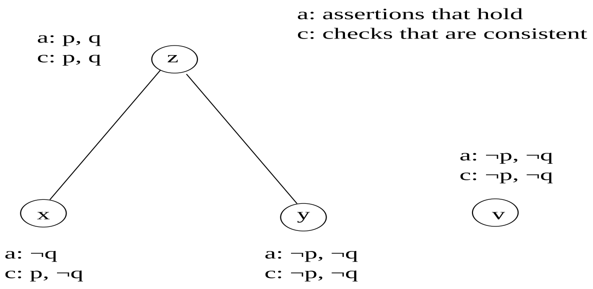

Fig. 7. A multiple-viewpoint scenario with consistent viewpoints which illustrates (i) that inconsistencies can be effectively detected and located (Example 7.1); and (ii) that the negation in (39) does not reverse inclusions (Example 8.2).

propositions p and q. For example, in P_x we have that p and $\neg q$ are consistent and $\neg q$ is a valid assertion. The conjunction $p \wedge q \wedge \neg p \wedge \neg q$ of all those literals that are consistent in some model P_u, the "master model" alluded to earlier, is clearly inconsistent and this could be tested for with a standard SAT solver. Priorities, however, provide a means for locating potential sources of such inconsistencies. We compute $\{\!|X{:}p|\!\}^{\mathrm{c}} = \{x, z\}$ and $\{\!|X{:}\neg p|\!\}^{\mathrm{c}} = \{v, y, z\}$. The intersection $\{\!|X{:}p|\!\}^{\mathrm{c}} \cap \{\!|X{:}\neg p|\!\}^{\mathrm{c}}$ contains all points (only z) that have an inconsistency based on $\Phi = \{p, \neg p\}$. In this case, x is the unique minimal (as $\mathrm{m} = \mathrm{c}$) u such that P_u causes $z \in \{\!|X{:}p|\!\}^{\mathrm{c}}$ and y is the unique minimal v such that P_v causes $z \in \{\!|X{:}\neg p|\!\}^{\mathrm{c}}$. Thus, conflict resolution should focus on P_x and P_y.

7.1 Abstract interpretation

In applications, checks $P \models^{\mathrm{m}} \phi$ may not be decidable. In that case, undecidable instances of `check(P_x,phi,m)` in Figure 6 need to be interpreted as booleans such that the resulting cone of influence be a conservative approximation of the one specified in (37). If (A, C) is the resulting approximation of $\{\!|X{:}\phi|\!\}$, its soundness is specified as $(A, C) \leq \{\!|X{:}\phi|\!\}$ in $(\mathsf{L}((X, \leq)), \subseteq) \times (\mathsf{U}((X, \leq)), \supseteq)$ — the pair (A, C) may not be in $\mathsf{M}_{(X,\leq)}$. Soundness can be secured if undecided checks are interpreted as `ff` and `tt` in mode a and c (respectively). For mode a, whenever $P_x \models^{\mathrm{a}} \phi$ holds but is undecided the algorithm then chooses the else-branch and L will not change. Conversely, any if-branch execution will add elements from $\{\!|X{:}\phi|\!\}^{\mathrm{a}}$ only. Thus, $A \subseteq \{\!|X{:}\phi|\!\}^{\mathrm{a}}$ follows. For mode c, whenever $P_x \models^{\mathrm{c}} \phi$ holds but is undecided the algorithm then chooses the if-branch and L will add elements that may not be in $\{\!|X{:}\phi|\!\}^{\mathrm{c}}$. Since only if-branch executions change L and since the else-branch is only executed when the check is decided to be `ff`, we infer $\{\!|X{:}\phi|\!\}^{\mathrm{c}} \subseteq C$ as required.

We note that this approximation applies directly to theorem provers if

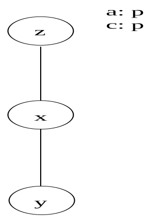

Fig. 8. A multiple-viewpoint scenario with consistent viewpoints which illustrates (Example 7.3) that the assertion-cone of influence is not contained in the consistency-cone of influence in general.

instances of (5) and (6) are representable as theorems. However, it does not match at all Alloy's reduction of checks to SAT instances. In Alloy, checks $P {\models}^{\mathrm{a}} \phi$ only return with `ff` or `probably_true`, whereas checks $P {\models}^{\mathrm{c}} \phi$ only return with `tt` or `probably_false`. Thus, the interpretation above would always reply with the same answer, `ff` in mode a and `tt` in mode c, and (A, C) would equal $(\{\}, X)$, the least (informative) element of $\mathsf{M}_{(X,\leq)}$. Alternatively, one may use Alloy's small-scope hypothesis [22] to obtain an unsound but potentially useful approximation of $\{\!|X{:}\phi|\!\}$.

7.2 Mixed power-order

Denotations of multiple-viewpoint checks are elements of the mixed power-order.

Lemma 7.2 *Let $(X, \leq)$ be a pre-order and all models P_x be consistent.*

(i) *For every $\phi \in \mathcal{L}$, the element $\{\!|X{:}\phi|\!\}$ is in $\mathsf{M}_{(X,\leq)}$.*

(ii) *If $\leq$ is contained in $\prec$, then $\{\!|X{:}\phi|\!\}^{\mathrm{a}} \subseteq \{\!|X{:}\phi|\!\}^{\mathrm{c}}$.*

Example 7.3 In general, $\{\!|X{:}p|\!\}^{\mathrm{a}}$ won't be a subset of $\{\!|X{:}p|\!\}^{\mathrm{c}}$ even if all P_x are consistent. For $y < x < z$, $P_z {\models}^{\mathrm{a}} p$, $P_x \not\models^{\mathrm{c}} p$, and $P_y \not\models^{\mathrm{c}} p$ (see Figure 8) we derive $x \in \{\!|X{:}p|\!\}^{\mathrm{a}} \setminus \{\!|X{:}p|\!\}^{\mathrm{c}}$. In that case, neither $y \leq z$ nor $x \leq z$ are instances of $\prec$.

7.3 Semantic entailment

Semantic entailment is preserved under a multiple-viewpoint interpretation.

Proposition 7.4 *For every partial order $(X, \leq)$, the map*

$$\Phi_X(\|\ \phi\ \|) \stackrel{\mathrm{def}}{=} \{\!|X{:}\phi|\!\} \colon (\mathcal{L}^{\mathrm{a}}, \subseteq) \times (\mathcal{L}^{\mathrm{c}}, \supseteq) \to \mathsf{M}_{(X,\leq)} \tag{38}$$

is monotone, where $\|\ \phi\ \| = (\|\ \phi\ \|^{\mathrm{a}}, \|\ \phi\ \|^{\mathrm{c}})$.

The monotonicity of Φ_X is significant. If $\llbracket \phi \rrbracket \leq \llbracket \psi \rrbracket$ models semantic entailment or if $\mathcal{P}$ is enriched with a theory of such entailments, then this theory will also hold in the multiple-viewpoint obligation semantics for $(X, \leq)$.

7.4 Refinement

The information collected in cones of influence of queries remains valid under refinement of viewpoints. Let $(X, \leq)$ be a finite partial order such that for each $x \in X$ there are two partial models P_x and Q_x with $Q_x \prec P_x$. We write $\{\!|X_P{:}\phi|\!\}$ and $\{\!|X_Q{:}\phi|\!\}$ for the cones-of-influence semantics over all P_x and Q_x, respectively.

Proposition 7.5 *For all ϕ, $\{\!|X_P{:}\phi|\!\} \leq \{\!|X_Q{:}\phi|\!\}$ in $\mathsf{M}_{(X,\leq)}$.*

Therefore, the detection and location of inconsistencies, based on $\{\!|X_P{:}\phi|\!\}$, remains to be a valid analysis and diagnostics if single-viewpoint models P_x get refined or implemented by some Q_x. Naturally, this remains true whenever the semantics in (37) serves as a reduction for a compositional multiple-*valued* semantics in subsequent sections.

7.5 Consistent viewpoints

There are cost-effective tools for guaranteeing the consistency of models P_x. We give two examples. Alloy's constraint analyzer may validate the consistency of complex partial models. Alternatively, the static structure may be such that its models are consistent by construction, e.g. the partial Kripke structures of Bruns & Godefroid [5]. Therefore, one may often assume that all models P_x are consistent. What is then at issue are inconsistencies that arise from conflicts across models P_x and our cone-of-influence semantics should prove a valuable tool.

8 Negation Over Multiple Viewpoints

We now study possible interpretations of negation on queries in the setting of partial models over multiple viewpoints. We see that the mode-sensitive negation of AC lattices, unlike Heyting negations, harmonizes well with the internal consistency of assertion and consistency checks. By Lemma 7.2(i),

$$\neg\{\!|X{:}\phi|\!\} \stackrel{\text{def}}{=} \{\!|X{:}\neg\phi|\!\} \tag{39}$$

has type $\mathrm{im}(\Phi_X) \to \mathsf{M}_{(\mathcal{P},\prec)}$ with component maps $\neg^{\mathrm{m}}_{\mathrm{X}}\{\!|X{:}\phi|\!\}^{\mathrm{m}} = \{\!|X{:}\neg\phi|\!\}^{\neg\mathrm{m}}$, $\mathrm{m} \in \{\mathrm{a}, \mathrm{c}\}$. We define

$$\mathcal{L}^{\mathrm{m}}_X \stackrel{\text{def}}{=} \{\{\!|X{:}\phi|\!\}^{\mathrm{m}} \mid \phi \in \mathcal{L}\} \qquad \mathrm{m} \in \{\mathrm{a}, \mathrm{c}\}\,. \tag{40}$$

Proposition 8.1 *Let $\leq$ be contained in $\prec$ and all P_x consistent.*

(i) *The negation in (39) acts as $(L, U) \mapsto (X \setminus U, X \setminus L)$.*

(ii) *The tuple $(\mathcal{L}^{\mathrm{a}}_X, \subseteq, \neg^{\mathrm{a}}_{\mathrm{X}}, \mathcal{L}^{\mathrm{c}}_X, \subseteq, \neg^{\mathrm{c}}_{\mathrm{X}})$ is an AC semi-lattice.*

In particular if $\leq$ is contained in $\prec$, then $\{\!|X{:}\phi|\!\}^{\mathrm{m}} \subseteq \{\!|X{:}\psi|\!\}^{\mathrm{m}}$ implies that $\{\!|X{:}\neg\psi|\!\}^{\neg\mathrm{m}}$ is contained in $\{\!|X{:}\neg\phi|\!\}^{\neg\mathrm{m}}$. This is no longer true without that assumption.

Example 8.2 Consider the partial order depicted in Figure 7. It assumes that all properties $p, q, \neg p, \neg q$ are distinct and states which properties hold in which mode. Properties don't hold in a mode if they are not stated. E.g. we have $P_x \models^{\mathrm{a}} \neg q$ but we don't have $P_x \models^{\mathrm{a}} p$. We compute $\{\!|X{:}p|\!\}^{\mathrm{a}} = X \setminus \{v\} = \{\!|X{:}q|\!\}^{\mathrm{a}}$, but $\{\!|X{:}\neg q|\!\}^{\mathrm{a}} = X \setminus \{z\}$ is not a subset of $\{\!|X{:}\neg p|\!\}^{\mathrm{a}} = X \setminus \{x, z\}$. Note that all models P_x, P_y, P_z, and P_v are consistent.

What structure does $\mathcal{L}^{\mathrm{m}}_X$ have for general partial orders $\leq$? With $\neg$ as in (39) we retain the first two axioms for AC semi-lattices. If $\leq$ is not contained in $\prec$, the remaining ones are lost — as seen in Example 8.2. Without the last two axioms, we have no guarantee that $(\mathcal{L}^{\mathrm{a}}_X, \subseteq)$ has finite infima, despite the fact that $\mathcal{L}^{\mathrm{c}}_X$ has finite suprema. On the bright side, $\{\!|X{:}\phi|\!\} \in \mathsf{M}_{(X,\leq)}$ ensures the internal consistency of our cone-of-influence semantics.

Each partial order $(\mathcal{L}^{\mathrm{m}}_X, \subseteq)$ sits within a finite distributive lattice of lower, respectively, upper sets. If $\leq$ is not contained in $\prec$, one may be tempted to seek an alternative semantics of negation by applying the corresponding Heyting negations of those lattices — $\neg_{\mathrm{A}} \colon \mathsf{L}((X, \leq)) \to \mathsf{L}((X, \leq))$ and $\neg_{\mathrm{C}} \colon \mathsf{U}((X, \leq)) \to \mathsf{U}((X, \leq))$ — to denotations:

$$\neg_{\mathrm{A}} L \stackrel{\mathrm{def}}{=} \bigcup \{L' \in \mathsf{L}((X, \leq)) \mid L \cap L' = \{\}\}$$

$$\neg_{\mathrm{C}} U \stackrel{\mathrm{def}}{=} \bigcap \{U' \in \mathsf{U}((X, \leq)) \mid U \cup U' = X\}\,. \tag{41}$$

This will restore the last two axioms of AC semi-lattices, but brake the first two axioms. Even worse, such a course of action abandons the consistency condition (36) altogether.

Example 8.3 Let $X = \{x, y, z\}$ be as in Figure 9. Consider $L = \{x, y\}$ and $U = \{y, z\}$. Then (L, U) is in $\mathsf{M}_{(X,\leq)}$ and equals $\{\!|X{:}p|\!\}$ over consistent models since $P_y \models^{\mathrm{a}} p$, $P_y \models^{\mathrm{c}} p$, and $P_z \models^{\mathrm{c}} p$ are the only holding instances of checks for p. We compute $\neg_{\mathrm{A}} L = \{z\}$ since any larger lower set needs to intersect L. Similarly, $\neg_{\mathrm{C}} U$ equals $\{x, y\}$ since any U' with $U \cup U' = X$ has to contain x and therefore y as well and since $\{x, y\}$ is such a U'. The pair $(\neg_{\mathrm{A}} L, \neg_{\mathrm{C}} U)$ is *not* in $\mathsf{M}_{(X,\leq)}$ since its sets are non-empty but have empty intersection.

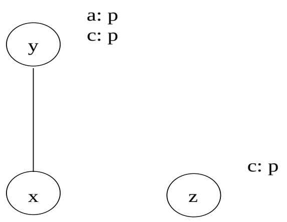

Fig. 9. A multiple-viewpoint scenario with consistent viewpoints which illustrates (Example 8.3) that a negation based on a pairwise Heyting negation won't map into $\mathsf{M}_{(X,\leq)}$ in general.

One can further show qualified inclusions and equalities between $\neg_{\mathrm{M}}$, $\neg^{\mathrm{m}}_{\mathrm{X}}$ of (39), and $\backslash$ and we refer the interested reader to the technical report [20].

9 Multiple-valued Semantics for Partial Models

We put to use the insights and techniques developed in previous sections by showing that our algorithm of Figure 6 computes a multiple-*valued* property semantics whenever single-viewpoint models are determined by the primes of the underlying AC lattice and the order between primes is preserved as refinement between their corresponding single-viewpoint models. These techniques apply to a wide range of static structures and logics. For example, in [20] we showed this for multiple-valued modal transition systems and a temporal logic with fixed points. Here we choose a case study of multiple-valued state-machines with first-order logic plus transitive closure.

Definition 9.1 Let $(X, \leq)$ be a finite partial order.

(i) We define $\mathsf{K}_{(X,\leq)}$ to be the set of all those elements (L, U) of $\mathsf{M}_{(X,\leq)}$ with $L \subseteq U$, in the ordering induced by $\mathsf{M}_{(X,\leq)}$.

(ii) For a set of propositions $\mathtt{AP}$, and a set of states S, a *modal multiple-valued state-machine* (mmv state-machine) P is a tuple

$$(S, I\colon S \to \mathsf{K}_{(X,\leq)}, R\colon S \times S \to \mathsf{K}_{(X,\leq)}, \mathrm{Lb}\colon \mathtt{AP} \times S \to \mathsf{K}_{(X,\leq)}) \tag{42}$$

where I identifies initial states, R is a transition function, and Lb a labeling function. We call $(\mathtt{AP}, (X, \leq))$ the *type* of P and write $P = (S, I, R, \mathrm{Lb})$ if that type is determined by context.

(iii) For each $s, s' \in S$ and $p \in \mathtt{AP}$ the sets $I(s)$, $R(s, s')$, and $\mathrm{Lb}(p, s)$ are of the form (L, U). For those bindings of L and U, we write $I(s)^{\mathrm{m}}$, $R(s, s')^{\mathrm{m}}$, and $\mathrm{Lb}(p, s)^{\mathrm{m}}$ for the set L (m being a) and U (m being c), respectively.

Since $I(s)$, $R(s, s')$, and $\mathrm{Lb}(p, s)$ are of the form $(L, U) \in \mathsf{K}_{(X,\leq)}$, we understand that L measures the degree of validity and U the degree of consistency of the statements "State s is initial," "There is a transition from s to s'," and "Proposition p holds at s" (respectively). Internal consistency is enforced since all witnesses $x \in L$ to validity are also witnesses to consistency ($x \in U$).

Example 9.2 If $X = \{*\}$, then $\mathsf{K}_{(X,\leq)}$ and $\mathsf{M}_{(X,\leq)}$ are isomorphic to the partial order $1/2 < 0, 1$ where 0 and 1 are distinct maximal elements [19]. Thus, mmv state-machines with type $(\mathtt{AP}, (\{*\}, \{(*,*)\}))$ correspond to state-based versions of modal transition systems [26]. If I is omitted and R never maps to 1/2, mmv state-machines with that type are the partial Kripke structures of [5,6].

We assume a first-order logic with transitive closure, unary relation symbols $\mathtt{p}$ for each $p \in \mathtt{AP}$, and a sole binary relation symbol $\mathtt{R}$ for R — where $+\mathtt{R}$ denotes the transitive closure of $\mathtt{R}$. With $\mathtt{u}$, $\mathtt{v}$ etc ranging over a set $\mathtt{Var}$ of variables for states, the grammar of the query logic is

$$\phi ::= \bot \mid \mathtt{p}(\mathtt{u}) \mid \mathtt{R}(\mathtt{u},\mathtt{u}) \mid +\mathtt{R}(\mathtt{u},\mathtt{u}) \mid \neg\phi \mid \phi \vee \phi \mid \exists \mathtt{u}\, \phi. \tag{43}$$

In particular, this logic is expressive enough to encode most practical temporal logics, e.g. LTL and CTL. We write $\phi \wedge \psi$ as a shorthand for $\neg(\neg\phi \vee \neg\psi)$. (In applications, the predicate $I(s)$ may be seen as a special case of $\mathtt{p}(\mathtt{u})$, whence we did not include it in (43).) Environments $\rho\colon \mathtt{Var} \to S$ bind variables to states. For such environments, we define a *compositional* query semantics $[\![\phi]\!]^{\mathrm{P}}_{\rho}$ as the meaning of ϕ in the mmv state-machine P under environment ρ. The demands on this semantics are that it

(i) is compositional;

(ii) has a formal notion of refinement such that refining models are consistent with but more specified than their refinements;

(iii) is sound with respect to this refinement;

(iv) has a clean interpretation of negation; and

(v) can be computed by projecting models down to single-valued ones and then using the efficient algorithm of Figure 6.

We expect that meanings $[\![\phi]\!]^{\mathrm{P}}_{\rho}$ are elements of $\mathsf{K}_{(X,\leq)}$, establishing their internal consistency. The interpretation of atomic formulas will be consistent due to the consistency of models. The interpretations of disjunction and existential quantification rely on the fact that the mixed power-order has a well defined union operation [14,15]:

$$(L,U) \sqcup (L',U') \stackrel{\text{def}}{=} (L \cup L', U \cup U')\colon \mathsf{M}_{(X,\leq)} \times \mathsf{M}_{(X,\leq)} \to \mathsf{M}_{(X,\leq)} \tag{44}$$

which is easily seen to restrict to $\mathsf{K}_{(X,\leq)}$. Since $\sqcup$ is commutative and associative, we write $\bigsqcup$ for the application of $\sqcup$ to a, possibly infinite, subset of $\mathsf{M}_{(X,\leq)}$. We now interpret $+\mathtt{R}$.

Definition 9.3 For mmv state-machines, we interpret `+R` as a function

$$R^+ \colon S \times S \to \mathsf{K}_{(X,\leq)}. \tag{45}$$

For each $s, s' \in S$, $R^+(s, s')$ equals (L, U), where U is defined as all those y for which we can find a finite "consistent" path from s to s' in P: there is some $n \geq 0$ and states $s = s_0$, s_1, …, $s_n = s'$ such that $y \in \bigcap_{i=0}^{n-1} R^{\mathrm{c}}(s_i, s_{i+1})$. (Note that U is an upper subset.) Let L be the set of all those $x \in X$ which have an "asserted" path from s to s': there is some $m \geq 0$ and states $s = t_0$, t_1, …, $t_m = s'$ such that $x \in \bigcap_{i=0}^{m-1} R^{\mathrm{a}}(t_i, t_{i+1})$. Since $R^{\mathrm{a}}(s, s') \subseteq R^{\mathrm{c}}(s, s')$ for all $s, s' \in S$, we infer $L \subseteq U$. Thus, the interpretation of `+R` is well defined.

The compositional query semantics $\llbracket\, \phi \,\rrbracket^{\mathrm{P}}_{\rho}$ is depicted in Figure 10. Below, we show that such meanings are elements of $\mathsf{K}_{(X,\leq)}$ and prove that $\llbracket\, \phi \,\rrbracket^{\mathrm{P}}_{\rho}$ equals $\{\!|X{:}\phi|\!\}_{\rho}$, where P_x is a state-machine with type $(\mathtt{AP}, (\{*\}, \{(*, *)\}))$ for each $x \in X$. In particular, such meanings can be efficiently computed with the algorithm in Figure 6.

Lemma 9.4 *Let P be a mmv state-machine. Then $\llbracket\, \phi \,\rrbracket^{\mathrm{P_a}}_{\rho} \subseteq \llbracket\, \phi \,\rrbracket^{\mathrm{P_c}}_{\rho}$ for all ϕ and ρ.*

Somewhat surprisingly, our case study would fail for meanings of type $\mathsf{M}_{(X,\leq)}$.

Example 9.5 According to the semantics in Figure 10, conjunction — defined through $\vee$ and $\neg$ — is interpreted as $(L, U) \sqcap (L', U') \stackrel{\mathrm{def}}{=} (L \cap L', U \cap U')$. This map is not well defined in $\mathsf{M}_{(X,\leq)}$, justifying our move from $\mathsf{M}_{(X,\leq)}$ to $\mathsf{K}_{(X,\leq)}$. Consider the $(X, \leq)$ of Example 4.4 on page 11. The pairs $(L, U) = (\{x, z\}, \{z\})$ and $(L', U') = (\{x, y\}, \{y\})$ are in $\mathsf{M}_{(X,\leq)}$ but the pair $(L \cap L', U \cap U') = (\{x\}, \{\})$ is not.

The pair (L, U) is the denotation of p and (L', U') is the denotation of q if x asserts p and q; y asserts q and deems q to be consistent; and z asserts p and deems q to be consistent.

One could circumvent this problem by saturating models as follows: If (L, U) is the meaning of an observable of the model, one can saturate that pair to an observable

$$(L, {\uparrow}L \cup U) \tag{46}$$

so in our example the pair (L, U) would turn into (L, X) and (L', U') turns into (L', X); thus, the saturated conjunction renders $(\{x\}, X) \in \mathsf{M}_{(X,\leq)}$.

Definition 9.6 Let P be a mmv state-machine as in (42).

(i) For each $x \in X$, the map

$$\pi_x \colon \mathsf{K}_{(X,\leq)} \to \mathsf{K}_{(\{*\},\{(*,*)\})} \tag{47}$$

$$\begin{array}{ll} \llbracket \bot \rrbracket^{\mathrm{P}}_{\rho} = (\{\}, \{\}) & \llbracket \mathtt{p}(\mathtt{u}) \rrbracket^{\mathrm{P}}_{\rho} = \mathrm{Lb}(p, \rho(\mathtt{u})) \\ \llbracket \mathtt{R}(\mathtt{u},\mathtt{v}) \rrbracket^{\mathrm{P}}_{\rho} = R(\rho(\mathtt{u}), \rho(\mathtt{v})) & \llbracket +\mathtt{R}(\mathtt{u},\mathtt{v}) \rrbracket^{\mathrm{P}}_{\rho} = R^{+}(\rho(\mathtt{u}), \rho(\mathtt{v})) \\ \llbracket \neg\phi \rrbracket^{\mathrm{P}}_{\rho} = (X \setminus \llbracket \phi \rrbracket^{\mathrm{P_c}}_{\rho}, X \setminus \llbracket \phi \rrbracket^{\mathrm{P_a}}_{\rho}) & \llbracket \phi \vee \psi \rrbracket^{\mathrm{P}}_{\rho} = \llbracket \phi \rrbracket^{\mathrm{P}}_{\rho} \sqcup \llbracket \psi \rrbracket^{\mathrm{P}}_{\rho} \\ \llbracket \exists \mathtt{u}\, \phi \rrbracket^{\mathrm{P}}_{\rho} = \bigsqcup_{s \in S} \llbracket \phi \rrbracket^{\mathrm{P}}_{\rho[\mathtt{u} \mapsto s]}. & \end{array}$$

Fig. 10. Compositional query semantics for mmv state-machines P under environment ρ. We write $\llbracket \phi \rrbracket^{\mathrm{P_a}}_{\rho}$ and $\llbracket \phi \rrbracket^{\mathrm{P_c}}_{\rho}$ for the first and second element of $\llbracket \phi \rrbracket^{\mathrm{P}}_{\rho}$, respectively.

is defined by $\pi_x(L, U) \stackrel{\text{def}}{=} (\{* \mid x \in L\}, \{* \mid x \in U\})$.

(ii) For each $x \in X$, we define the mmv state-machine

$$P_x \stackrel{\text{def}}{=} (S, \pi_x \circ I, \pi_x \circ R, \pi_x \circ L). \tag{48}$$

Also, $R^+_x \stackrel{\text{def}}{=} \pi_x \circ R^+$.

It is elementary to check that the structure P_x in (48) is a mmv state-machine of type $(\mathtt{AP}, (\{*\}, \{(*, *)\}))$, for $L \subseteq U$ implies $\{* \mid x \in L\} \subseteq \{* \mid x \in U\}$. Throughout this section, P_x refers to the mmv state-machine in (48). We specify the semantics for assertion checks $\models^{\mathrm{a}}$ and consistency checks $\models^{\mathrm{c}}$ for these models P_x, where $\mathrm{m} \in \{\mathrm{a}, \mathrm{c}\}$:

- $P_x \not\models^{\mathrm{m}}_{\rho} \bot$;
- $P_x \models^{\mathrm{m}}_{\rho} \mathtt{p}(\mathtt{u})$ iff $* \in \pi_x(\mathrm{Lb}^{\mathrm{m}}(p, \rho(\mathtt{u})))$;
- $P_x \models^{\mathrm{m}}_{\rho} \mathtt{R}(\mathtt{u},\mathtt{v})$ iff $* \in \pi_x(R^{\mathrm{m}}(\rho(\mathtt{u}), \rho(\mathtt{v})))$;
- $P_x \models^{\mathrm{m}}_{\rho} +\mathtt{R}(\mathtt{u},\mathtt{v})$ iff $* \in \pi_x((R^+)^{\mathrm{m}}(\rho(\mathtt{u}), \rho(\mathtt{v})))$;
- $P_x \models^{\mathrm{m}}_{\rho} \neg\phi$ iff $P_x \not\models^{\neg\mathrm{m}}_{\rho} \phi$;
- $P_x \models^{\mathrm{m}}_{\rho} \phi \vee \psi$ iff $(P_x \models^{\mathrm{m}}_{\rho} \phi$ or $P_x \models^{\mathrm{m}}_{\rho} \psi)$;
- $P_x \models^{\mathrm{m}}_{\rho} \exists \mathtt{u}\, \phi$ iff for some $s \in S$, $P_x \models^{\mathrm{m}}_{\rho[\mathtt{u} \mapsto s]} \phi$.

From this definition, we readily obtain $P_x \models^{\mathrm{m}}_{\rho} \phi \wedge \psi$ iff $(P_x \models^{\mathrm{m}}_{\rho} \phi$ and $P_x \models^{\mathrm{m}}_{\rho} \psi)$. We record important facts about these partial models and their compositional semantics $\models^{\mathrm{m}}$.

Proposition 9.7 (i) *For all ϕ, ρ, and x we have that $P_x \models^{\mathrm{a}}_{\rho} \phi$ implies $P_x \models^{\mathrm{c}}_{\rho} \phi$.*
(ii) *Whenever $x \leq y$, then $P_x \prec P_y$.*

Given item (ii) above and the mode-sensitive treatment of negation in AC lattices, we can now show that $\llbracket \phi \rrbracket^{\mathrm{P}}_{\rho}$ equals $\{\!|X{:}\phi|\!\}_{\rho}$ — where the latter is defined as in (37) with $\models^{\mathrm{m}}_{\rho}$ instead of $\models^{\mathrm{m}}$. In particular, we can compute the former semantics by using the more efficient algorithm of Figure 6 and computing the single-view models P_x whenever needed in its control flow. We first prove that $\{\!|X{:}\phi|\!\}_\rho$, although computed by a non-compositional synthesis of obligations in (37), is compositional.

Lemma 9.8 *Let $(P_x)_{x\in X}$ be as in (48).*

(i) *All models P_x are consistent.*

(ii) *For all ρ we have $\{\!|X{:}\bot|\!\}_\rho = (\{\}, \{\})$.*

(iii) *For all $\mathtt{p}(\mathtt{u})$ and ρ we have $\{\!|X{:}\mathtt{p}(\mathtt{u})|\!\}_\rho = \mathrm{Lb}(p, \rho(\mathtt{u}))$.*

(iv) *For all $\mathtt{R}(\mathtt{u},\mathtt{v})$ and ρ we have $\{\!|X{:}\mathtt{R}(\mathtt{u},\mathtt{v})|\!\}_\rho = R(\rho(\mathtt{u}), \rho(\mathtt{v}))$.*

(v) *For all ϕ, ρ, and $\mathrm{m} \in \{\mathrm{a}, \mathrm{c}\}$ we have $\{\!|X{:}\neg\phi|\!\}^{\mathrm{m}}_\rho = X \setminus \{\!|X{:}\phi|\!\}^{\neg\mathrm{m}}_\rho$.*

(vi) *For all ϕ, ψ, and ρ we have $\{\!|X{:}\phi \vee \psi|\!\}_\rho = \{\!|X{:}\phi|\!\}_\rho \sqcup \{\!|X{:}\psi|\!\}_\rho$.*

(vii) *For all $\exists\mathtt{u}\,\phi$ and ρ we have $\{\!|X{:}\exists\mathtt{u}\,\phi|\!\}_\rho = \bigsqcup_{t\in S} \{\!|X{:}\phi|\!\}_{\rho[\mathtt{u}\mapsto t]}$.*

With this lemma at hand, we can prove the main result of this section.

Theorem 9.9 *For all mmv state-machines P, formulas ϕ, and environments ρ we have $\|\ \phi\ \|^{\mathrm{P}}_\rho = \{\!|X{:}\phi|\!\}_\rho$.*

Example 9.10 Consider the mmv state-machine depicted in Figure 11. We compute $\|\ \exists\mathtt{v}\,(+\mathtt{R}(\mathtt{v},\mathtt{v}) \wedge \neg\mathtt{p}(\mathtt{v}))\ \|^{\mathrm{P}}$. This query asks whether there is a state that is on a cycle and does not satisfy p. From Figure 10, we infer that this query has meaning $(L_0 \cup L_1 \cup L_2, U_0 \cup U_1 \cup U_2)$, where

$$L_i = \|\ +\mathtt{R}(\mathtt{v},\mathtt{v})\ \|^{\mathrm{P_a}}_{[v\mapsto s_i]} \cap (X \setminus \|\ \mathtt{p}(\mathtt{v})\ \|^{\mathrm{P_c}}_{[v\mapsto s_i]})$$

$$U_i = \|\ +\mathtt{R}(\mathtt{v},\mathtt{v})\ \|^{\mathrm{P_c}}_{[v\mapsto s_i]} \cap (X \setminus \|\ \mathtt{p}(\mathtt{v})\ \|^{\mathrm{P_a}}_{[v\mapsto s_i]})\,. \tag{49}$$

We have $L_0 = \{x\} \cap (X \setminus \{u, v, x\}) = \{\}$ since only x asserts a path from s_0 to s_0 and only y thinks that p is inconsistent at that state. Similarly, we compute $L_1 = \{x, y\} \cap (X \setminus X) = \{\}$ and $L_2 = \{x\} \cap (X \setminus \{u, v, y\}) = \{x\}$.

As for the consistency semantics, we obtain $U_0 = \{u, v, x\} \cap (X \setminus \{x\}) = \{u, v\}$ since only u, v, and x think that a loop from s_0 to s_0 is consistent and only x asserts p at that state. Similarly, $U_1 = X \cap (X \setminus \{x, y\}) = \{u, v\}$ and $U_2 = X \cap (X \setminus \{y\}) = \{u, v, x\}$. Thus, $\|\ \exists\mathtt{v}\,(+\mathtt{R}(\mathtt{v},\mathtt{v}) \wedge \neg\mathtt{p}(\mathtt{v}))\ \|^{\mathrm{P}} = (\{x\}, \{u, v, x\})$. This means that x thinks the query is valid; whereas u, v, and x deem the query to be consistent.

9.1 Degree of partiality

Our query semantics comes with a measurement of uncertainty. The map $\mu_P\colon \mathcal{L} \times \mathrm{Env} \to \{0, 1, \ldots, |X|\}$, defined by

$$\mu_P(\phi, \rho) \stackrel{\mathrm{def}}{=} |\,\|\ \phi\ \|^{\mathrm{P_c}}_\rho \setminus \|\ \phi\ \|^{\mathrm{P_a}}_\rho\,| \tag{50}$$

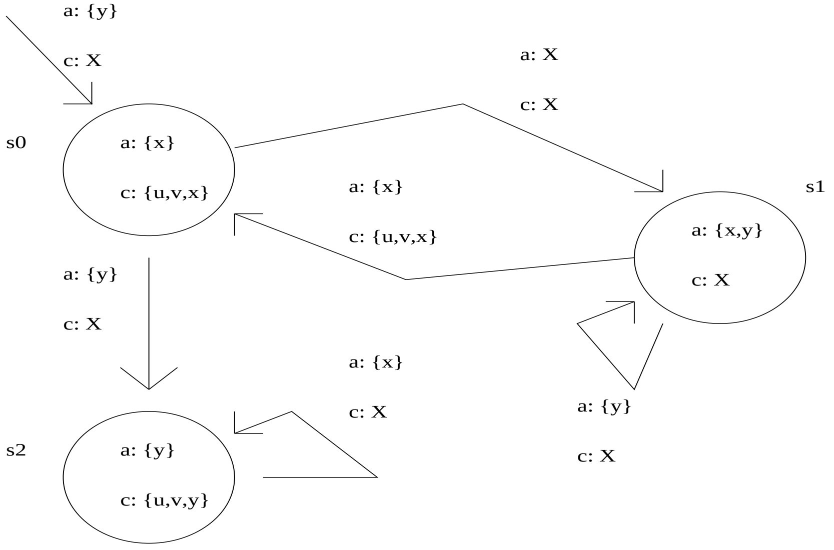

Fig. 11. A mmv state-machine with $\mathtt{AP} = \{p\}$ and $(X, \leq)$ as in Figure 12. The labeling function Lb is specified within states; e.g. $\mathrm{Lb}(p, s_0) = (\{x\}, \{u, v, x\})$. Modeling information of the form $(\{\}, \{\})$, be it initial states or transitions, is not shown. The transition into s_0 without any source specifies $I(s_0)$.

measures the degree of partiality of property ϕ over the mmv state-machine P in environment ρ. For the check from Example 9.10, this computes to $3 - 1 = 2$. For a systematic theory of measurements see [28,27,36].

9.2 Refinement

Since $\| \phi \|^{\mathrm{P}}_{\rho}$ equals $\{\!|X{:}\phi|\!\}_{\rho}$, Proposition 7.5 ensures soundness of our semantics if P changes to Q such that $Q_x \prec P_x$ for all $x \in X$. It is routine to define co-inductive relations for mmv state-machines — similar to the ones for modal transition systems [26] or partial Kripke structures [5] — such that all instances of such relations represent refinements between all projected models Q_x and P_x.

9.3 Total models

A mmv state-machine P with type $(\mathtt{AP}, (\{*\}, \{(*, *)\}))$ is *total* iff its functions Lb and R map into the diagonal $\{(\{\}, \{\}), (\{*\}, \{*\})\}$. For total models of that type, $\models^{\mathrm{a}}$ equals $\models^{\mathrm{c}}$ and is the usual semantics for Kripke structures.

Example 9.11 Totality cannot be characterized in this way for mvv state-machines whose type contains a non-flat order. Consider the mmv state-machine with one state, one unary predicate p, and no transitions. Figure 12

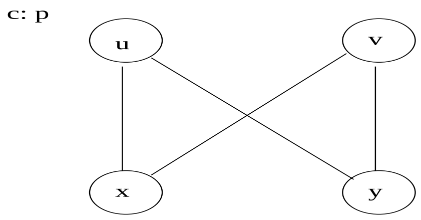

Fig. 12. Assertion checks are different from consistency checks for total mmv state-machines in general (Example 9.11).

shows $(X, \leq)$ and the labeling function for that state. The pair $(L, U) = (\{x\}, \{x, u, v\})$ is maximal in $\mathsf{K}_{(X,\leq)}$, but $L \neq U$.

9.4 Strong semantics

In comparison to the semantics of Section 2, our specification of checks over P_x, put forward in this section, is an abstraction: $P_x \models^{\mathrm{a}}_{\rho} \phi \vee \psi$ certainly holds whenever ($P_x \models^{\mathrm{a}}_{\rho} \phi$ or $P_x \models^{\mathrm{a}}_{\rho} \psi$), but our compositional semantics makes the latter also a necessary condition. This results in a loss of precision. Stronger versions of this semantics exist — e.g. the focus operation of Ball et al. [2] for program analysis and Bruns and Godefroid's generalized model checking [6] for partial Kripke structures. One can extend generalized model checking to mmv state-machines P through a strong version of the cone-of-influence semantics. Projected models P_x are then checked with respect to the strong semantics, which is an instance of our ontology. (However, it is unclear whether this semantics has a meaningful interpretation at the level of mmv state-machines.)

Definition 9.12 Let $\mathcal{P}$ be the set of all mmv state-machines of the same type $(\mathtt{AP}, (X, \leq))$ and state space S. For $P, Q \in \mathcal{P}$,

(i) $P \sqsubseteq Q$ iff $I_P \leq I_Q$, $R_P \leq R_Q$, and $\mathrm{Lb}_P \leq \mathrm{Lb}_Q$ in the point-wise ordering of $\mathsf{K}_{(X,\leq)}$;

(ii) Q is *total* iff Q is maximal with respect to $\sqsubseteq$: for all $K \in \mathcal{P}$, $Q \sqsubseteq K$ implies $K \sqsubseteq Q$; and

(iii) the implementation relationship is defined as

$$\mathtt{impl} \stackrel{\mathrm{def}}{=} \{(P, Q) \mid P \sqsubseteq Q,\ Q \text{ total}\}. \tag{51}$$

It is immediate that P is total iff all its component functions I_P, R_P, and Lb_P map into the set of maximal elements of $\mathsf{K}_{(X,\leq)}$. From Example 9.11 we learn that $L = U$ does not necessarily hold for maximal elements (L, U) in $\mathsf{K}_{(X,\leq)}$.

Example 9.13 If P has type $(\mathtt{AP}, (\{*\}, \{(*, *)\}))$, then P is a three-valued Kripke structure and $T_P \stackrel{\mathrm{def}}{=} \{Q \mid (P, Q) \in \mathtt{impl}\}$ forms a complete cover [6] of

Kripke structures: every $Q \in T_P$ refines P and every Kripke structure which refines P is refined by some element of T_P, where refinement is defined similar to the completeness order in [6]. Elements Q of T_P resolve *all* modeling information $(\{\}, \{*\})$ (truth unknown) of P into either $(\{*\}, \{*\})$ (true) or $(\{\}, \{\})$ (false).

Definition 9.14 (i) We define a strong semantics $\models^{\mathrm{m(s)}}$ for mmv state-machines P of type $(\mathtt{AP}, (\{*\}, \{(*, *)\}))$ as in (5–6), where $\models$ turns into $\models_\rho$, the environment-based standard semantics over Kripke structures.

(ii) The strong cone-of-influence semantics $\{\!|X{:}\phi|\!\}^{\mathrm{s}}_\rho \stackrel{\mathrm{def}}{=} (\{\!|X{:}\phi|\!\}^{\mathrm{a(s)}}_\rho, \{\!|X{:}\phi|\!\}^{\mathrm{c(s)}}_\rho)$ is defined as in (37), expect that we replace all $\models^{\mathrm{m}}$ with $\models^{\mathrm{m(s)}}_\rho$.

The relations $\models^{\mathrm{m(s)}}$ represent the strong semantics $\models^{\mathrm{s}}$ of [6]. Soundness of our compositional semantics $\models^{\mathrm{m}}$ with respect to the strong semantics holds: $P \models^{\mathrm{a}}_\rho \phi$ implies $P \models^{\mathrm{a(s)}}_\rho \phi$ and $P \models^{\mathrm{c(s)}}_\rho \phi$ implies $P \models^{\mathrm{c}}_\rho \phi$. Therefore, $\{\!|X{:}\phi|\!\}_\rho \leq \{\!|X{:}\phi|\!\}^{\mathrm{s}}_\rho$ for all ϕ and ρ. Since $\models^{\mathrm{m(s)}}_\rho$ is non-compositional, there is probably no strong compositional analogue of $[\![\, \phi \,]\!]^{\mathrm{P}}_\rho$.

10 Related Work

Fitting studies multiple-valued Kripke structures and a multiple-valued proof theory in [11], where a partial order of experts enforces consistency of their assertions about the truth and falsity of transitions and state observables.

Easterbrook and Chechik use Fitting's models and a De Morgan lattice 'negotiated' among experts to derive a multiple-valued version of the temporal logic CTL, its models, and its model checking algorithm (e.g. [8]).

Bruns and Godefroid [7] devise a query checker for temporal logic that, given a Kripke structure and a query with a hole in it as input, returns a (strongest) formula of propositional logic that, when placed into the query's hole, makes the query true for that Kripke structure. Among other things, this technique is of use in the reverse engineering of systems.

The handbook article by Blamey [4] nicely surveys extant work on partial logics and their semantics, emphasizing linguistics and philosophical logic as applications.

A wide range of multiple-valued logics employ a lattice of truth values, which is often finite and distributive. Semi-bilattices [13] take this further by having two orderings: an information order which captures uncertainty, and a logical order which captures degree of truth.

11 Conclusions

In this paper we developed proper foundations for partial model checking that subsume work on multiple-viewpoint and multiple-valued checking. We proposed AC semi-lattices as a generalization of De Morgan lattices in the finite distributive case. These structures lead to an efficient algorithm for detecting and locating inconsistencies across model boundaries, where models are partially ordered by priorities, which specify the poset of primes for the underlying AC lattice. We also saw that the same algorithm provides a, generally more efficient, implementation of compositional multiple-*valued* model checking whenever the order on primes leads to refinement between their corresponding 'projected' single-viewpoint models. We also showed that such proper semantic foundations are unlikely to exist if negation is implemented via Heyting negations.

Although our work is completely inspired by and drawn from the work of Fitting in [11] — and programmatically anticipated in the position paper [17] — our contributions complement his approach in that we

(i) allow experts to *under-specify* their models,

(ii) allow the priority ('dominance' in [11]) order *not* to be reflected as refinement, turning this mathematical weakness into a practically relevant detection and diagnostics tool,

(iii) *don't* develop multiple-valued proof theories, but synthesize and instrument obligation sets as *ordinary* model checks, and

(iv) make a case for not using Heyting negations.

Item (iii) is central to our approach. For one thing, it allows the use of whatever techniques available to specify and check single-viewpoint models (modular description languages, abstraction, compact data structures etc). For another, it integrates well with design-review techniques, where checks of single-viewpoint queries may be answered by real-life experts whose consistency and assertion rulings are based on their fallible domain-specific expertise. The algorithm in Figure 6 and its instrumentations discussed in this paper apply equally well in this setting.

Acknowledgments

We gratefully acknowledge Glenn Bruns and Marsha Chechik for valuable discussions and most useful comments regarding this material. Glenn Bruns and, independently, Gordon Plotkin suggested to consider the possibility of embedding AC lattices into De Morgan lattices.

References

[1] S. Abramsky and A. Jung. Domain theory. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 3, pages 1–168. Oxford Univ. Press, 1994.

[2] T. Ball, A. Podelski, and S. K. Rajamani. Boolean and Cartesian Abstraction for Model Checking C Programs. In T. Margaria and W. Yi, editors, *Proceedings of TACAS'2001*, volume 2031 of *LNCS*, pages 268–283, Genova, Italy, April 2001. Springer Verlag.

[3] N. Belnap. A useful four-valued logic. In J. M. Dunn, editor, *Modern Uses of Many-Valued Logic*, pages 8–37. Reidel, 1977.

[4] S. Blamey. Partial logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, Volume 3: *Alternatives to Classical Logic*, pages 1–70, 1986.

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

[6] G. Bruns and P. Godefroid. Generalized Model Checking: Reasoning about Partial State Spaces. In *Proceedings of CONCUR'2000 (11th International Conference on Concurrency Theory)*, volume 1877 of *Lecture Notes in Computer Science*, pages 168–182. Springer Verlag, August 2000.

[7] G. Bruns and P. Godefroid. Temporal Logic Query Checking. In *Proceedings of the 16th Annual IEEE Symposium on Logic in Computer Science*, pages 409–417, Boston, Massachusetts, 16-19 June 2001. IEEE Computer Society Press.

[8] M. Chechik, B. Devereux, and S. M. Easterbrook. Implementing a Multi-Valued Symbolic Model Checker. In *Proceedings of the Fourth European Joint Conferences on Theory and Practice of Software (ETAPS2001) - Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2001)*, Lecture Notes in Computer Science, pages 404–419, Genova, Italy, 2-6 April 2001. Springer Verlag.

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

[10] J. M. Dunn. *What Is Negation?*, chapter *A Comparative Study of Various Model-Theoretic Treatments of Negation: A History of Formal Negation.* Kluwer Academic Publishers, 1999.

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

[12] G. Gierz, K. H. Hofmann, K. Keimel, J. D. Lawson, M. Mislove, and D. S. Scott. *A Compendium of Continuous Lattices.* Springer Verlag, 1980.

[13] M. L. Ginsberg. Multivalued Logics. In *Proceedings of the 5th International Conference on Artificial Intelligence (AAAI'86)*, pages 243–247. Morgan-Kaufmann, 1986.

[14] C. Gunter. The mixed power domain. *Theoretical Computer Science*, 103:311–334, 1992.

[15] R. Heckmann. Power domains and second order predicates. *Theoretical Computer Science*, 111:59–88, 1993.

[16] A. Hunter and B. Nuseibeh. Managing Inconsistent Specifications: Reasoning, Analysis, and Action. *ACM Transactions on Software Engineering and Methodology*, 7(4):335–367, October 1998.

[17] M. Huth and S. Pradhan. Abstraction and refinement for model checking inconsistent systems. Position paper presented at *The Second International Workshop on Living with Inconsistencies*, 13 May 2001, Toronto, Canada. URL: `www.cis.ksu.edu/~huth/lwi2.ps`

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

[19] M. Huth, R. Jagadeesan, and D. A. Schmidt. A domain equation for refinement of partial systems. Accepted for publication in the journal *Mathematical Structures in Computer Science*, January 2003.

[20] M. Huth and S. Pradhan. Lifting assertion and consistency checkers from single to multiple viewpoints. Technical Report TR 2002/11, Imperial College London, Department of Computing, London, UK, March 2002.

[21] D. Jackson. Alloy: A Lightweight Object Modelling Language. Technical Report TR-797, Laboratory of Computer Science, Massachusetts Institute of Technology, 28 July 2000.

[22] D. Jackson, I. Shlyakhter, and M. Sridharan. A Micromodularity Mechanism. In *Proceedings of the ACM SIGSOFT Conference on the Foundations of Software Engineering/European Software Engineering Conference (FSE/ESEC'01)*, September 2001.

[23] P. T. Johnstone. *Stone spaces*. Number 3 in Cambridge studies in advanced mathemathics. Cambridge University Press, 1982.

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

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

[26] K. G. Larsen and B. Thomsen. A Modal Process Logic. In *Third Annual Symposium on Logic in Computer Science*, pages 203–210. IEEE Computer Society Press, 1988.

[27] K. Martin. Nonclassical techniques for models of computation. In *Topology Proceedings*, volume 24, 1999.

[28] K. Martin. *A Foundation for Computation*. PhD thesis, Department of Mathematics, Tulane University of Louisiana, New Orleans, LA 70118, 2000.

[29] B. Nuseibeh and S. M. Easterbrook. The Process of Inconsistency Management: A framework for understanding. In *Proceedings of the First International Workshop on the Requirements Engineering Process (REP'99)*, 2-3 September 1999.

[30] B. Nuseibeh, J. Kramer, and A. Finkelstein. A Framework for Expressing the Relationships Between Multiple Views in Requirements Specification. *IEEE Transactions on Software Engineering*, 20(10):760–773, October 1994.

[31] C. S. Pasareanu, M. B. Dwyer, and W. Visser. Finding Feasible Counter-examples when Model Checking Abstracted Java Programs. In T. Margaria and W. Yi, editors, *In: Proc. of the 7th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'01)*, volume 2031 of *LNCS*, pages 284–298, Genova, Italy, April 2-4 2001. Springer Verlag.

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

[33] A. Schiper. Failure Detection vs Group Membership. In H. Hermanns and R. Segala, editors, *Process Algebra and Probabilistic Methods, Performance Modeling and Verification, Second Joint International Workshop PAPM-PROBMIV 2002*, volume 2399 of *Lecture Notes in Computer Science*, pages 1–15, Copenhagen, Denmark, July 25-26 2002. Springer.

[34] A. van Lamsweerde. Goal-Oriented Requirements Engineering. Invited talk at the BSC-RESG One-Day Symposium on "Using Formal Models to Understand Requirements Better." Imperial College London, 6 November 2002.

[35] D. J. Walker. Bisimulation and divergence. *Information and Computation*, 85(2):202–241, 1990.

[36] P. Waszkiewicz. *Quantitative Continuous Domains*. PhD thesis, School of Computer Science, University of Birmingham, Birmingham, B15 2TT, United Kingdom, August 2002.

[37] K. Wiegers. First Things First: Prioritizing Requirements. *Software Development Online*, September 1999.

Proofs

Proof. [of Proposition 3.2] Verification of the first two axioms in Figure 5 (left) follows from the fact that $\neg\neg\phi$ and ϕ are semantically equivalent for $\models$; e.g. $P \models^{c} \neg\neg\phi$ iff (there is some P' with $(P, P') \in \texttt{impl}$ and $P' \models \neg\neg\phi$) iff (there is some P' with $(P, P') \in \texttt{impl}$ and $P' \models \phi$) iff $P \models^{c} \phi$. For the third axiom, let

$$\| \phi \|^{a} \subseteq \| \psi \|^{a}. \tag{.1}$$

Assume $P \in \neg_{a} \| \psi \|^{a} (= \| \neg\psi \|^{c})$. We claim that $P \in \neg_{a} \| \phi \|^{a} (= \| \neg\phi \|^{c})$. Otherwise, $P \notin \| \neg\phi \|^{c}$, i.e. $P \not\models^{c} \neg\phi$. Then (7) renders $P \models^{a} \phi$. By (.1), we obtain $P \models^{a} \psi$. By assumption, $P \models^{c} \neg\psi$ which means that there is a P' with $(P, P') \in \texttt{impl}$ and $P' \models \neg\psi$. From $P \models^{a} \psi$ and $(P, P') \in \texttt{impl}$ we infer $P' \models \psi$. But then $P' \models \psi \wedge \neg\psi$ follows, contradicting (4).

For the fourth axiom, we reason in a similar style. Let

$$\| \phi \|^{c} \subseteq \| \psi \|^{c}. \tag{.2}$$

Assume $P \in \neg_{c} \| \psi \|^{c} (= \| \neg\psi \|^{a})$. We claim that $P \in \neg_{c} \| \phi \|^{c} (= \| \neg\phi \|^{a})$. Proof by contradiction: otherwise, $P \notin \| \neg\phi \|^{a}$, i.e. $P \not\models^{a} \neg\phi$. But then (5) implies the existence of some P' with $(P, P') \in \texttt{impl}$ and $P' \not\models \neg\phi$, i.e. $P' \models \phi$. Then (6) renders $P \models^{c} \phi$. By (.2), $P \models^{c} \psi$ as well. But by assumption, $P \models^{a} \neg\psi$ — a contradiction now reasoned as for the third axiom.

Finally, $\| \phi \|^{a} \cap \| \psi \|^{a} = \| \phi \wedge \psi \|^{a}$ and $\| \phi \vee \neg\phi \|^{a} = \mathcal{P}$ is the greatest element of $(\mathcal{L}^{a}, \subseteq)$. Therefore, $(\mathcal{L}^{a}, \subseteq)$ has binary infima and the infimum of the empty set; thus, it is an inf-semi-lattice with one.

□

Proof. [of Proposition 3.4] Given $(D, \leq, \neg)$, the tuple $(D, \leq, \neg, D, \leq^{-1}, \neg)$ is such an AC lattice. Conversely, any such AC lattice determines a De Morgan lattice $(\mathcal{L}_{a}, \leq_{a}, \neg_{a})$. □

Proof. [of Lemma 4.1] Finite partial orders $(\mathcal{L}_{a}, \leq_{a})$ with finite infima have finite suprema; $x \vee y$ is the infimum of the finite set $\{l \in \mathcal{L}_{a} \mid x, y \leq_{a} l\}$ and the supremum of the empty set is the infimum of all elements in $\mathcal{L}_{a}$. □

Proof. [of Proposition 4.2]

(i) Using $U = \bigcap_{x \notin U} \epsilon_X(x)$ for each $U \in \mathsf{U}(X, \leq)$, $\mathrm{ext}^{\mathcal{U}}(f)(U)$ has to equal $\bigvee\{f(x) \mid x \notin U\}$. We then can show that the functor $\mathsf{U}(\cdot)$ is monotone.

(ii) The first three equations are immediate. Finally, $\backslash_{a} \circ \mathsf{L}(f) \circ \backslash_{c}$ is a sup-map so it suffices to show $(\backslash_{a} \circ \mathsf{L}(f) \circ \backslash_{c}) \circ \epsilon_X = \epsilon_Y \circ f$. But $(\backslash_{a} \circ \mathsf{L}(f) \circ \backslash_{c}) \circ \epsilon_X = \backslash_{a} \circ (\mathsf{L}(f) \circ \eta_X) = \backslash_{a} \circ (\eta_Y \circ f) = \epsilon_Y \circ f$.

(iii) Since automorphisms are lifted by functors, $f \mapsto \mathsf{L}(f) \colon \mathrm{Aut}(X, \leq) \rightarrow$

$\mathrm{Aut}(\mathsf{L}(X,\leq))$ is well defined; it is an injection as the functor is faithful over *partial orders*. Conversely, given $g \in \mathrm{Aut}(\mathsf{L}(X,\leq))$, the map $g \circ \eta_X\colon (X,\leq) \to \mathsf{L}(X,\leq)$ is monotone and so $\mathrm{ext}^{\mathcal{L}}(g\circ\eta_X)$ is well defined, but g equals $\mathrm{ext}^{\mathcal{L}}(g\circ\eta_X)$ since g is also a sup-map ρ with $\rho\circ\eta_X = g\circ\eta_X$. The proof for $\mathsf{U}(\cdot)$ follows from this and $\mathsf{U}(f) = \backslash_{\mathrm{a}} \circ \mathsf{L}(f) \circ \backslash_{\mathrm{c}}$.

□

Proof. [of Theorem 4.3] By Stone duality — which in this simple setting amounts to Birkhoff's Representation Theorem for finite distributive lattices — there exists an order-isomorphism $\phi\colon (\mathcal{L}_{\mathrm{a}},\leq_{\mathrm{a}}) \cong (\mathsf{L}(X,\leq),\subseteq)$ [1] for some finite partial order $(X,\leq)$ since $(\mathcal{L}_{\mathrm{a}},\leq_{\mathrm{a}})$ is a finite distributive lattice. But then $\backslash_{\mathrm{c}}\circ\phi\circ\neg_{\mathrm{c}}\colon (\mathcal{L}_{\mathrm{c}},\leq_{\mathrm{c}}) \to (\mathsf{U}(X,\leq),\subseteq)$ is an order-isomorphism as well. Thus, we may assume $(\mathcal{L}_{\mathrm{a}},\leq_{\mathrm{a}}) = (\mathsf{L}(X,\leq),\subseteq)$ and $(\mathcal{L}_{\mathrm{c}},\leq_{\mathrm{c}}) = (\mathsf{U}(X,\leq),\subseteq)$. Since $\neg_{\mathrm{a}}\colon (\mathsf{L}(X,\leq),\subseteq) \to (\mathsf{U}(X,\leq),\supseteq)$ is an order-isomorphism, it restricts to an order-isomorphism of the respective sets of complete primes. (An element p in a complete lattice L is a complete prime iff for all $A \subseteq L$, $p = \bigvee A$ implies $p \in A$.) The set of complete primes of $(\mathsf{L}(X,\leq),\subseteq)$ is the image of η_X; the set of complete primes of $(\mathsf{U}(X,\leq),\supseteq)$ is the image of ϵ_X. Thus, for every $x \in X$ there is a unique $i(x) \in X$ with $\neg_{\mathrm{a}}(\eta_X(x)) = \epsilon_X(i(x))$. Since the map $\epsilon_X\colon (X,\leq) \to (\mathsf{U}(X,\leq),\supseteq)$ is an order-isomorphism onto its image, $i\colon (X,\leq) \to (X,\leq)$ is monotone. Thus, $\neg_{\mathrm{a}}\circ\eta_X = \epsilon_X\circ i$ implies $\neg_{\mathrm{a}} = \mathrm{ext}^{\mathcal{L}}(\epsilon_X\circ i)$.

Similarly, we infer the existence of a monotone map $j\colon (X,\leq) \to (X,\leq)$ with $\neg_{\mathrm{c}}\circ\epsilon_X = \eta_X\circ j$, and so $\neg_{\mathrm{c}} = \mathrm{ext}^{\mathcal{U}}(\eta_X\circ j)$. Thus, it suffices to show $j = i^{-1}$. But $\epsilon_X\circ i = \neg_{\mathrm{a}}\circ\eta_X = (\neg_{\mathrm{a}}\circ\backslash_{\mathrm{c}})\circ\epsilon_X$ implies $\neg_{\mathrm{a}}\circ\backslash_{\mathrm{c}} = \mathsf{U}(i)$. Dually, $\eta_X\circ j = \neg_{\mathrm{c}}\circ\epsilon_X = (\neg_{\mathrm{c}}\circ\backslash_{\mathrm{a}})\circ\eta_X$ implies $\neg_{\mathrm{c}}\circ\backslash_{\mathrm{a}} = \mathsf{L}(j)$. Thus, $\mathsf{L}(i) = \backslash_{\mathrm{c}}\circ\mathsf{U}(i)\circ\backslash_{\mathrm{a}} = \backslash_{\mathrm{c}}\circ(\neg_{\mathrm{a}}\circ\backslash_{\mathrm{c}})\circ\backslash_{\mathrm{a}} = \backslash_{\mathrm{c}}\circ\neg_{\mathrm{a}}$. From this we immediately get that $\mathsf{L}(i)$ is the inverse of $\mathsf{L}(j)$. Thus, $j = i^{-1}$ as the functor $\mathsf{L}(\cdot)$ is faithful over partial orders. □

Proof. [of Theorem 4.5] Stone duality again ensures an order-isomorphism $\phi\colon (\mathcal{L},\leq) \to (\mathsf{L}(X,\leq),\subseteq)$ for some finite partial order $(X,\leq)$. But then the map

$$\mathrm{neg} \stackrel{\mathrm{def}}{=} \phi\circ\neg\circ\phi^{-1} \tag{.3}$$

is monotone for the types $(\mathsf{L}(X,\leq),\subseteq) \to (\mathsf{L}(X,\leq),\supseteq)$ and $(\mathsf{L}(X,\leq),\supseteq) \to (\mathsf{L}(X,\leq),\subseteq)$. Since $\mathrm{neg}\circ\mathrm{neg} = \mathrm{id}_{\mathsf{L}(X,\leq)}$, neg is an order-isomorphism and, therefore, maps complete primes of $(\mathsf{L}(X,\leq),\subseteq)$ — all elements of the form $\downarrow x$ — to complete primes of $(\mathsf{L}(X,\leq),\supseteq)$ — all elements of the form $X \setminus \uparrow x$. Hence, $\mathrm{neg}(\downarrow x) = X \setminus \uparrow i(x)$ for a unique $i(x)$ in X. If $x \leq y$, then $\mathrm{neg}(\downarrow y) \subseteq \mathrm{neg}(\downarrow x)$ follows, from which we infer $\uparrow i(x) \subseteq \uparrow i(y)$, i.e. $i(y) \leq i(x)$. Therefore, $x \mapsto i(x)\colon (X,\leq) \to (X,\leq^{-1})$ is monotone. Similarly, neg maps complete primes of $(\mathsf{L}(X,\leq),\supseteq)$ to complete primes of $(\mathsf{L}(X,\leq),\subseteq)$. Thus, $\mathrm{neg}(X \setminus {\uparrow}x) = {\downarrow}j(x)$ for a unique $j(x)$ in X. Similarly, $x \mapsto j(x)\colon (X,\leq) \to (X,\leq^{-1})$ is shown to be monotone. The equation ${\downarrow}x = \mathrm{neg}(\mathrm{neg}({\downarrow}x))$ renders $j(i(x)) = x$ for all $x \in X$. Dually, equation $X \setminus {\uparrow}x = \mathrm{neg}(\mathrm{neg}(X \setminus {\uparrow}x))$ yields $i(j(x)) = x$ for all $x \in X$, so $i\colon (X,\leq) \to (X,\leq^{-1})$ is an order isomorphism with $j\colon (X,\leq^{-1}) \to (X,\leq)$ as inverse.

We claim that $j = i$. Since inverses are unique, it suffices to show that $i \circ i = \mathrm{id}_X$. We compute ${\downarrow}x = \mathrm{neg}(\mathrm{neg}({\downarrow}x)) = \mathrm{neg}(X \setminus {\uparrow}i(x)) = \mathrm{neg}(\bigcup\{{\downarrow}y \mid y \in V \setminus {\uparrow}i(x)\}) = \bigcap\{\mathrm{neg}({\downarrow}y) \mid i(x) \not\leq y\} = \bigcap\{V \setminus {\uparrow}i(y) \mid i(x) \not\leq y\}$. In particular, x is an element of the latter intersection. Therefore we have

$$\forall y \in X,\ i(x) \not\leq y \text{ implies } i(y) \not\leq x. \tag{.4}$$

For $y \stackrel{\mathrm{def}}{=} j(x)$, $i(y) = i(j(x)) = x$ therefore implies $i(y) \leq x$ and so (.4) renders $i(x) \leq y = j(x)$. Since $i\colon (X,\leq) \to (X,\leq^{-1})$ is monotone, this results in $x = i(j(x)) \leq i(i(x))$. Conversely, suppose that it is not the case that $i(i(x)) \leq x$. Then $x \in X \setminus {\uparrow}i(i(x))$, i.e. ${\downarrow}x \subseteq X \setminus {\uparrow}i(i(x)) = \mathrm{neg}({\downarrow}i(x))$. But then ${\downarrow}i(x) = \mathrm{neg}(\mathrm{neg}({\downarrow}i(x))) \subseteq \mathrm{neg}({\downarrow}x) = X \setminus {\uparrow}i(x)$ implies $i(x) \in X \setminus {\uparrow}i(x)$, a contradiction.

Finally, we show that neg is defined as in (25). Given $L \in \mathsf{L}(X,\leq)$, we have

$$\mathrm{neg}(L) = \mathrm{neg}(\bigcup_{x' \in L} {\downarrow}x') = \bigcap_{x' \in L} \mathrm{neg}({\downarrow}x') = \bigcap_{x' \in L} X \setminus {\uparrow}i(x') \stackrel{\mathrm{def}}{=} A. \tag{.5}$$

We claim that A equals $B \stackrel{\mathrm{def}}{=} \{i(x) \mid x \in X \setminus L\}$:

- If $y \in A$, then $x' \in L$ implies $i(x') \not\leq y$, i.e. $i(y) \not\leq x'$ since $i\colon (X,\leq) \to (X,\leq^{-1})$ is an order-isomorphism with i as its inverse. Thus, $i(y) \in X \setminus L$ since $x' \in L$ was arbitrary. Therefore, $y = i(i(y)) \in B$.
- If $y \in B$, then $y = i(x)$ for some $x \in X \setminus L$. Given $x' \in L$, we have $x \not\leq x'$ since $x \in X \setminus L$ and $L \in \mathsf{L}(X,\leq)$. Thus, $i(x') \not\leq i(x) = y$ shows $y \in X \setminus {\uparrow}i(x')$, i.e. $y \in A$.

□

Proof. [of Proposition 6.2]

(i) Let $Q \prec P$.
- $P \in \|\ \phi\ \|^{\mathrm{a}}$ means $P \models^{\mathrm{a}} \phi$ and so $Q \models^{\mathrm{a}} \phi$ follows from $Q \prec P$. Therefore, $Q \in \|\ \phi\ \|^{\mathrm{a}}$ and $\|\ \phi\ \|^{\mathrm{a}}$ is a lower set.
- $Q \in \|\ \phi\ \|^{\mathrm{c}}$ means $Q \models^{\mathrm{c}} \phi$. Proof by contradiction: if $P \not\models^{\mathrm{c}} \phi$, then (5) and (6) render $P \models^{\mathrm{a}} \neg\phi$. From $Q \prec P$ we then infer $Q \models^{\mathrm{a}} \neg\phi$ which contradicts $Q \models^{\mathrm{c}} \phi$ since all total models are consistent. Thus, $\|\ \phi\ \|^{\mathrm{c}}$ is an upper set.

(ii) The right-hand side of (35) is always contained in the left-hand side by item (i). Thus, we focus on the other containment only.

- Let ϕ be consistent and $P \in \| \phi \|^{\mathrm{a}}$.
  - If P is consistent, then there is some P' with $(P, P') \in \texttt{impl}$. From $P \in \| \phi \|^{\mathrm{a}}$ we then infer $P' \models \phi$ and so $P \in \| \phi \|^{\mathrm{a}} \cap \| \phi \|^{\mathrm{c}}$.
  - If P is inconsistent, then $P \prec Q$ for all partial models Q. Since ϕ is consistent, there is some $T \in \mathcal{T}$ with $T \models \phi$ and so $T \models^{\mathrm{a}} \phi$ by (10). Thus, $T \in \| \phi \|^{\mathrm{a}} \cap \| \phi \|^{\mathrm{c}}$ and $P \prec T$ prove that P is contained in the right-hand side.
- If ϕ is inconsistent, then $\| \phi \|^{\mathrm{a}} = \{P \in \mathcal{P} \mid P \models^{\mathrm{a}} \phi\} = \{P \in \mathcal{P} \mid P \text{ is inconsistent}\}$ and $\| \phi \|^{\mathrm{c}} = \{P \in \mathcal{P} \mid P \models^{\mathrm{c}} \phi\} = \{\}$ show that (35) cannot hold.

(iii) For inconsistent ϕ, if $\mathcal{P}$ is restricted to consistent models only, the left-hand side of (35) now computes to $\{\}$, so it satisfies the mix condition (35). For consistent ϕ, we argue as in the previous item.

□

Proof. [of Lemma 7.2]

(i) Given $x \in \{|X{:}\phi|\}^{\mathrm{a}}$, there is some $y \in X$ with $x \leq y$ and $P_y \models^{\mathrm{a}} \phi$. Since P_y is consistent, $P_y \models^{\mathrm{a}} \phi$ implies $P_y \models^{\mathrm{c}} \phi$. Since $y \leq y$ we infer $x \leq y \in \{|X{:}\phi|\}^{\mathrm{a}} \cap \{|X{:}\phi|\}^{\mathrm{c}}$.

(ii) Let $x \notin \{|X{:}\phi|\}^{\mathrm{c}}$. So $y \leq x$ implies $P_y \not\models^{\mathrm{c}} \phi$, i.e. $P_y \models^{\mathrm{a}} \neg\phi$. In particular, $P_x \models^{\mathrm{a}} \neg\phi$ since $x \leq x$. Proof by contradiction: if $x \in \{|X{:}\phi|\}^{\mathrm{a}}$, there is some y with $x \leq y$ and $P_y \models^{\mathrm{a}} \phi$. Since $\leq$ is contained in $\prec$ this implies $P_x \models^{\mathrm{a}} \phi$. Thus, $P_x \models^{\mathrm{a}} \phi \wedge \neg\phi$ contradicts that P_x is consistent: there is some P' with $(P, P') \in \texttt{impl}$ and so $P' \models \phi \wedge \neg\phi$ would follow for the consistent P'.

□

Proof. [of Proposition 7.4] Let $\| \phi \| \leq \| \psi \|$ in $(\mathcal{L}^{\mathrm{a}}, \subseteq) \times (\mathcal{L}^{\mathrm{c}}, \supseteq)$. For $x \in \{|X{:}\phi|\}^{\mathrm{a}}$ there is some $y \in X$ with $x \leq y$ and $P_y \models^{\mathrm{a}} \phi$. Since $\| \phi \|^{\mathrm{a}} \subseteq \| \psi \|^{\mathrm{a}}$, we infer $P_y \models^{\mathrm{a}} \psi$ which renders $x \in \{|X{:}\psi|\}^{\mathrm{a}}$. For $x' \in \{|X{:}\psi|\}^{\mathrm{c}}$ there is some $y' \in X$ with $y' \leq x'$ and $P_{y'} \models^{\mathrm{c}} \psi$. Since $\| \psi \|^{\mathrm{c}} \subseteq \| \phi \|^{\mathrm{c}}$, we infer $P_{y'} \models^{\mathrm{c}} \phi$ which renders $x' \in \{|X{:}\phi|\}^{\mathrm{c}}$.

□

Proof. [of Proposition 7.5] Given $x \in \{|X_P{:}\phi|\}^{\mathrm{a}}$, there is some $y \in X$ with $x \leq y$ and $P_y \models^{\mathrm{a}} \phi$. From $Q_y \prec P_y$ we infer $Q_y \models^{\mathrm{a}} \phi$, and so $x \in \{|X_Q{:}\phi|\}^{\mathrm{a}}$. Given $u \in \{|X_Q{:}\phi|\}^{\mathrm{c}}$, there is some $v \in X$ such that $v \leq u$ and $Q_v \models^{\mathrm{c}} \phi$. Together with $Q_v \prec P_v$, Proposition 6.2(i) renders $P_u \models^{\mathrm{c}} \phi$ from which we obtain $u \in \{|X_P{:}\phi|\}^{\mathrm{c}}$.

□

Proof. [of Proposition 8.1]

(i) For every $\phi \in \mathcal{L}$, $x \in X \setminus \{\!|X{:}\phi|\!\}^{\mathrm{c}}$ iff $\downarrow x \subseteq \{y \in X \mid P_y \not\models^{\mathrm{c}} \phi\}$. Since $\leq$ is contained in $\prec$, this is equivalent to $P_x \not\models^{\mathrm{c}} \phi$ which, by (9), is equivalent to $P_x \models^{\mathrm{a}} \neg\phi$. But the latter is equivalent to $x \in \{\!|X{:}\neg\phi|\!\}^{\mathrm{a}}$ since $\leq$ is contained in $\prec$. Similarly, $x' \in X \setminus \{\!|X{:}\phi|\!\}^{\mathrm{a}}$ iff $\uparrow x' \subseteq \{y \in X \mid P_y \not\models^{\mathrm{a}} \phi\}$. Since $\leq$ is contained in $\prec$, this inclusion holds iff $P_x \not\models^{\mathrm{a}} \phi$, which is equivalent to $P_x \models^{\mathrm{c}} \neg\phi$ by (9), i.e. $x \in \{\!|X{:}\neg\phi|\!\}^{\mathrm{c}}$ as $\leq$ is contained in $\prec$.

(ii) As for the sup-semi-lattice with zero, the partial order $(\mathcal{L}^{\mathrm{c}}_X, \subseteq)$ does have binary suprema and a zero since $\{\!|X{:}\phi|\!\}^{\mathrm{c}} \cup \{\!|X{:}\psi|\!\}^{\mathrm{c}} = \{\!|X{:}\phi \vee \psi|\!\}^{\mathrm{c}}$ and $\{\} = \{\!|X{:}\phi \wedge \neg\phi|\!\}^{\mathrm{c}}$. Thus, it suffices to show that $\neg^{\mathrm{a}}_{\mathrm{X}}$ and $\neg^{\mathrm{c}}_{\mathrm{X}}$ are order-inverses. But this follows from the first item.

□

Proof. [of Lemma 9.4] We prove this by structural induction on ϕ. For $\perp$, both sides equal $\{\}$. For $\mathtt{p}(\mathtt{u})$ and $\mathtt{R}(\mathtt{u},\mathtt{v})$ this follows from the fact that P's transition and labeling functions map into $\mathsf{K}_{(X,\leq)}$. For $+\mathtt{R}(\mathtt{u},\mathtt{v})$, let $x \in \llbracket +\mathtt{R}(\mathtt{u},\mathtt{v}) \rrbracket^{\mathrm{P_a}}_{\rho}$. Then there is a path $\rho(\mathtt{u}) = s_0, s_1, \ldots, s_n = \rho(\mathtt{v})$ such that $x \in R^{\mathrm{a}}(s_i, s_{i+1})$ for all such $0 \leq i < n$. But $R^{\mathrm{a}}(s_i, s_{i+1}) \subseteq R^{\mathrm{c}}(s_i, s_{i+1})$ then shows $x \in \llbracket +\mathtt{R}(\mathtt{u},\mathtt{v}) \rrbracket^{\mathrm{P_c}}_{\rho}$ through that same path. The cases $\neg\phi$, $\phi \vee \psi$, and $\exists \mathtt{u}\, \phi$ follow since their interpretations are monotone with respect to inclusion in both coordinates, so they map into $\mathsf{K}_{(X,\leq)}$; e.g. for the existential quantifier, if $(L_s, U_s) \in \mathsf{K}_{(X,\leq)}$ for all $s \in S$, then $\bigcup_{s\in S} L_s \subseteq \bigcup_{s \in S} U_s$. □

Proof. [of Proposition 9.7]

(i) This is a routine argument (similar to the one in [18]) and omitted.

(ii) Given $x \leq y$, we show $P_x \prec P_y$ by showing the stronger statement

(.6) "$P_y \models^{\mathrm{a}}_{\rho} \phi$ implies $P_x \models^{\mathrm{a}}_{\rho} \phi$; and $P_x \models^{\mathrm{c}}_{\rho} \phi$ implies $P_y \models^{\mathrm{c}}_{\rho} \phi$."

for all ϕ and ρ by structural induction on ϕ.

- For $\perp$, (.6) holds trivially.
- For $\mathtt{p}(\mathtt{u})$, this follows from the fact that the first component of $L(p, \rho(\mathtt{u}))$ is a lower set and the second component is an upper set.
- Similarly, for $\mathtt{R}(\mathtt{u},\mathtt{v})$ this follows from the fact that the first component of $R(\rho(\mathtt{u}), \rho(\mathtt{v}))$ is a lower set and the second component an upper set.
- For $+\mathtt{R}(\mathtt{u},\mathtt{v})$,
  · let $P_y \models^{\mathrm{a}}_{\rho} +\mathtt{R}(\mathtt{u},\mathtt{v})$. Then there are $m \geq 0$ and a set $\{t_i \in S \mid 0 \leq i < m\}$ with $\rho(\mathtt{u}) = t_0$ and $\rho(\mathtt{v}) = t_m$ such that y is in the lower set $\bigcap_i (R^+)^{\mathrm{a}}(t_i, t_{i+1})$; but $x \leq y$ then implies $x \in \bigcap_i (R^+)^{\mathrm{a}}(t_i, t_{i+1})$, so there is an R^{a}-path in P_x from t_0 to t_m, i.e. $P_x \models^{\mathrm{a}}_{\rho} +\mathtt{R}(\mathtt{u},\mathtt{v})$;
  · dually, let $P_x \models^{\mathrm{c}}_{\rho} +\mathtt{R}(\mathtt{u},\mathtt{v})$. Then there are $k \geq 0$ and a set $\{s_i \in S \mid 0 \leq i < k\}$ with $\rho(\mathtt{u}) = s_0$ and $\rho(\mathtt{v}) = s_k$ such that x is in the

upper set $\bigcap_i (R^+)^{\mathrm{c}}(s_i, s_{i+1})$; since $x \leq y$, we have $y \in \bigcap_i (R^+)^{\mathrm{c}}(s_i, s_{i+1})$ from which we infer that there is an R^{c}-path from s_0 to s_k in P_y, i.e. $P_y \models^{\mathrm{c}}_{\rho} +\mathtt{R}(\mathtt{u}, \mathtt{v})$.

- For $\neg\phi$,
  - let $P_y \models^{\mathrm{a}}_{\rho} \neg\phi$. Then it is not the case that $P_y \models^{\mathrm{c}}_{\rho} \phi$. Using induction on ϕ, we also don't have $P_x \models^{\mathrm{c}}_{\rho} \phi$, from which we get $P_x \models^{\mathrm{a}}_{\rho} \neg\phi$
  - let $P_x \models^{\mathrm{c}}_{\rho} \neg\phi$. If $P_y \models^{\mathrm{c}}_{\rho} \neg\phi$ is not the case, then $P_y \models^{\mathrm{a}}_{\rho} \neg\neg\phi$, i.e. $P_y \models^{\mathrm{a}}_{\rho} \phi$. By induction, this yields $P_x \models^{\mathrm{a}}_{\rho} \phi$ — contradicting $P_x \models^{\mathrm{c}}_{\rho} \neg\phi$ by the definition of $\models^{\mathrm{m}}_{\rho}$.
- The proofs for $\phi \vee \psi$ are straightforward and omitted.
- For $\exists \mathtt{u}\, \phi$, this is easily shown by induction since (.6) quantifies over all environments, so it applies to all $\rho[\mathtt{u} \mapsto s]$ with $s \in S$.

□

Proof. [of Lemma 9.8]

(i) We claim that $P_x \models^{\mathrm{c}}_{\rho} \mathtt{p}(\mathtt{u}) \vee \neg\mathtt{p}(\mathtt{u})$. Let (L, U) be $L(p, \rho(\mathtt{u}))$. Assume that $P_x \models^{\mathrm{c}}_{\rho} \mathtt{p}(\mathtt{u})$ is not the case. Then x is not an element of U. Since $L \subseteq U$, $x \notin L$, too. But then we don't have $P_x \models^{\mathrm{a}}_{\rho} \mathtt{p}(\mathtt{u})$ from which we infer $P_x \models^{\mathrm{c}}_{\rho} \neg\mathtt{p}(\mathtt{u})$.

(ii) This is clear.

(iii) Let (L, U) be $L(p, \rho(\mathtt{u}))$.
- let $x \in \{\!|X{:}\mathtt{p}(\mathtt{u})|\!\}^{\mathrm{c}}_{\rho}$. Then there is some $y \leq x$ with $P_y \models^{\mathrm{c}}_{\rho} \mathtt{p}(\mathtt{u})$. Since $y \leq x$, Proposition 9.7(ii) ensures $P_x \models^{\mathrm{c}}_{\rho} \mathtt{p}(\mathtt{u})$ which implies $x \in U$;
- conversely, let $x \in U$. Then $P_x \models^{\mathrm{c}}_{\rho} \mathtt{p}(\mathtt{u})$. Since $x \leq x$, this renders $x \in \{\!|X{:}\mathtt{p}(\mathtt{u})|\!\}^{\mathrm{c}}_{\rho}$;
- let $x \in \{\!|X{:}\mathtt{p}(\mathtt{u})|\!\}^{\mathrm{a}}_{\rho}$. Then there is some $x \leq y$ with $P_y \models^{\mathrm{a}}_{\rho} \mathtt{p}(\mathtt{u})$. Since $x \leq y$, Proposition 9.7(ii) ensures $P_x \models^{\mathrm{a}}_{\rho} \mathtt{p}(\mathtt{u})$ which implies $x \in L$;
- conversely, let $x \in L$. Then $P_x \models^{\mathrm{a}}_{\rho} \mathtt{p}(\mathtt{u})$. Since $x \leq x$, this renders $x \in \{\!|X{:}\mathtt{p}(\mathtt{u})|\!\}^{\mathrm{a}}_{\rho}$.

(iv) For $\mathtt{R}(\mathtt{u}, \mathtt{v})$, we reason in the same style as for $\mathtt{p}(\mathtt{u})$.

(v) For $\neg\phi$,
- let $x \in \{\!|X{:}\neg\phi|\!\}^{\mathrm{c}}_{\rho}$. Then there is some $y \leq x$ with $P_y \models^{\mathrm{c}} \neg\phi$. By Proposition 9.7(ii), this implies $P_z \models^{\mathrm{c}} \neg\phi$ for all z with $x \leq z$ as $\leq$ is transitive. Thus, for any such z it not the case that $P_z \models^{\mathrm{a}} \phi$. Therefore $x \in X \setminus \{\!|X{:}\phi|\!\}^{\mathrm{a}}_{\rho}$. Conversely, let $x' \in X \setminus \{\!|X{:}\phi|\!\}^{\mathrm{a}}_{\rho}$. Then for all z' with $x' \leq z'$ it is not the case that $P_{z'} \models^{\mathrm{a}} \phi$. In particular, for $z' = x'$ this implies $P_{x'} \models^{\mathrm{c}} \neg\phi$. Since $x' \leq x'$, this shows $x' \in \{\!|X{:}\neg\phi|\!\}^{\mathrm{c}}_{\rho}$;
- the proof for the other mode is dual and omitted;

(vi) For $\phi \vee \psi$,

- let $x \in \{\!|X{:}\phi \vee \psi|\!\}^{\mathrm{a}}_{\rho}$. Then there is some y with $x \leq y$ and $P_y {\models}^{\mathrm{a}}_{\rho} \phi \vee \psi$. Without loss of generality (since ${\models}^{\mathrm{a}}_{\rho}$ interprets $\vee$ compositionally), $P_y {\models}^{\mathrm{a}}_{\rho} \psi$ which secures $x \in \{\!|X{:}\psi|\!\}^{\mathrm{a}}_{\rho} \subseteq \{\!|X{:}\phi|\!\}^{\mathrm{a}}_{\rho} \cup \{\!|X{:}\psi|\!\}^{\mathrm{a}}_{\rho}$;
- conversely, let $x' \in \{\!|X{:}\phi|\!\}^{\mathrm{a}}_{\rho} \cup \{\!|X{:}\psi|\!\}^{\mathrm{a}}_{\rho}$. Without loss of generality $x' \in \{\!|X{:}\phi|\!\}^{\mathrm{a}}_{\rho}$ so there is some z with $x' \leq z$ and $P_z {\models}^{\mathrm{a}}_{\rho} \phi$ which implies $P_z {\models}^{\mathrm{a}}_{\rho} \phi \vee \psi$, from which we infer $x' \in \{\!|X{:}\phi \vee \psi|\!\}^{\mathrm{a}}_{\rho}$; and
- the proof for the other mode is dual and omitted.

(vii) For $\exists \mathtt{u}\, \phi$,

- let $x \in \{\!|X{:}\exists \mathtt{u}\, \phi|\!\}^{\mathrm{c}}_{\rho}$; then there is some $y \leq x$ with $P_y {\models}^{\mathrm{c}}_{\rho} \exists \mathtt{u}\, \phi$. Thus, there exists some $t' \in S$ such that $P_y {\models}^{\mathrm{c}}_{\rho[\mathtt{u} \mapsto \mathtt{t'}]} \phi$. Since $y \leq x$, this implies $x \in \{\!|X{:}\phi|\!\}^{\mathrm{c}}_{\rho[\mathtt{u} \mapsto \mathtt{t'}]}$ which is contained in $\bigcup_{t \in S} \{\!|X{:}\phi|\!\}^{\mathrm{c}}_{\rho[\mathtt{u} \mapsto \mathtt{t}]}$;
- let $x' \in \bigcup_{t \in S} \{\!|X{:}\phi|\!\}^{\mathrm{c}}_{\rho[\mathtt{u} \mapsto \mathtt{t}]}$. Then there exists some $t'' \in S$ such that $x' \in \{\!|X{:}\phi|\!\}^{\mathrm{c}}_{\rho[\mathtt{u} \mapsto \mathtt{t''}]}$. Therefore, there is some $y' \leq x'$ such that $P_{y'} {\models}^{\mathrm{c}}_{\rho[\mathtt{u} \mapsto \mathtt{t''}]}$. From the latter, we obtain $P_{y'} {\models}^{\mathrm{c}}_{\rho} \exists \mathtt{u}\, \phi$. Since $y' \leq x'$, this renders $x' \in \{\!|X{:}\exists \mathtt{u}\, \phi|\!\}^{\mathrm{c}}_{\rho}$, and
- the two cases for mode a are dual and omitted.

□

Proof. [of Theorem 9.9] We prove this by structural induction over all ϕ and ρ. For all cases but $+\mathtt{R}(\mathtt{u}, \mathtt{v})$, this follows directly from Lemma 9.8. For $+\mathtt{R}(\mathtt{u}, \mathtt{v})$,

- let $x \in \{\!|X{:}+\mathtt{R}(\mathtt{u}, \mathtt{v})|\!\}^{\mathrm{a}}_{\rho}$. Then there is some $x \leq y$ with $P_y {\models}^{\mathrm{a}}_{\rho} +\mathtt{R}(\mathtt{u}, \mathtt{v})$. So there is a finite path $\rho(\mathtt{u}) = t_0, t_1, \ldots, t_m = \rho(\mathtt{v})$ with $y \in \bigcap_i R^{\mathrm{a}}(t_i, t_{i+1})$. Since the latter is a lower set, $x \leq y$ renders $x \in \bigcap_i R^{\mathrm{a}}(t_i, t_{i+1})$. Therefore, $x \in [\![\, +\mathtt{R}(\mathtt{u}, \mathtt{v}) \,]\!]^{\mathrm{P_a}}_{\rho}$.
- let $x \in [\![\, +\mathtt{R}(\mathtt{u}, \mathtt{v}) \,]\!]^{\mathrm{P_a}}_{\rho}$. Then there is a finite path $\rho(\mathtt{u}) = s_0, s_1, \ldots, s_n = \rho(\mathtt{v})$ with $x \in \bigcap_i R^{\mathrm{a}}(s_i, s_{i+1})$. From that we conclude $P_x {\models}^{\mathrm{a}}_{\rho} \mathtt{R}(\mathtt{u}, \mathtt{v})$. Since $x \leq x$, this implies $x \in \{\!|X{:}+\mathtt{R}(\mathtt{u}, \mathtt{v})|\!\}^{\mathrm{a}}_{\rho}$; and
- the cases for mode c are dual and omitted.

□